ВІСНИК

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ
"Київський політехнічний інститут"

Інформатика, управління та обчислювальна техніка

58

2013

Міністерство освіти і науки України
Національний технічний університет України «КПІ»

ВІСНИК

НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ «КПІ»

Інформатика, управління та обчислювальна техніка

Заснований у 1964 р.
Випуск 58

Київ “ВЕК+” 2013

УДК 004

Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.: Век+, – 2013. – № 58. – 166 с.

Рекомендований до друку Вченою радою факультету інформатики та обчислювальної техніки, протокол № 3 від 28.10.2013

Головний редактор: Луцький Г.М., д.т.н., проф.

Заст. гол. ред.: *Стіренко С.Г., к.т.н., доц.,*
Пустоваров В.І., к.т.н., доц.

Відповідальний секретар: *Поспішний О.С.*

Редакційна колегія: *Павлов О.А., д.т.н., проф.,*
Теленик С.Ф., д.т.н., проф.,
Бузовський О.В., д.т.н., проф.,
Симоненко В.П., д.т.н., проф.,
Жабін В.І., д.т.н., проф.,
Кулаков Ю.О., д.т.н., проф.,
Марковський О.П., к.т.н., доц.,
Стенін Н.А., д.т.н., проф.,
Томашевський В.М., д.т.н., проф.
Ляшевський С.Е., д.т.н., проф.

Описано результати дослідження і створення компонентів обчислювальних й інформаційних систем і комплексів, пристроїв автоматики та передавання даних, систем автоматизації програмування, контролю й діагностики, штучного інтелекту тощо.

Для аспірантів, студентів, фахівців з обчислювальної техніки, систем керування, автоматизації програмування, штучного інтелекту та інших інформаційно-обчислювальних систем.

Журнал індексується наступними наукометричними базами даних: Російський індекс наукового цитування (РІНЦ), Directory of Open Access Journals (DOAJ), Google Scholar.

Журнал входить до переліку наукових фахових видань МОН України.

ISSN 2310-3620

Свідоцтво про державну реєстрацію № 16949-5719 Р від 17.06.2010

Збірник наукових праць українською, англійською та російською мовами

Web-ресурс – http://it-visnyk.kpi.ua

Підп. до друку 28.10.2013. Формат 60×84 1/16. Гарнітура Times. Папір офсетний № 1.
Наклад 150 прим.
Надруковано в ЗАТ “ВІПОЛ”, 03151 м.Київ, вул. Волинська, 60.

© Національний технічний університет України “КПІ”, 2013

ЗМІСТ

Згуровский М.З., Болдак А.А., Ефремов К.В., Сергеева Н.А., Забаринская Л.П., Шестопалов И.П., Нисилевич М.В. Применение методов интеллектуального анализа данных для эмпирических исследований взаимосвязи гелио- и геофизических процессов……. 4

Павлов А.А., Мисюра Е.Б., Лисецкий Т.Н., Сперкач М.О., Халус Е.А. Четырехуровневая модель планирования, принятия решений и оперативного управления в сетевых системах с ограниченными ресурсами…….. 11

Павлов А.А., Мисюра Е.Б., Лисецкий Т.Н. Составление расписания выполнения независимых заданий идентичными параллельными приборами, моменты запуска которых меньше общего директивного срока……. 24

Теленик С.Ф., Амонс О.А., Єфремов К.В., Жук С.В. Семантична інтеграція різнорідних інформаційних ресурсів……. 29

Anna Kyselova Context Data Analysis For Microgrid Control System…… 46

Дьяконова С.В., Зайченко Ю.П. Подход к решению задачи автоматизированного обнаружения зданий на спутниковых зображениях…….. 51

Надеран Э., Зайченко Ю.П. Подход к разработке системы распознавания рукописных математических выражений вводимых в эвм в режиме реального времени……. 56

Молчановський О.І., Знахуренко В.П. метод індуктивного навчання в основі рекомендаційної системи подарунків……. 61

Кулаков Ю. А., Воротников В. В., Гуменюк И. В. Анализ структуры телекоммуникационной сети путем представления ее топологии предфрактальным графом……. 68

Кулаков Ю.А., Коган А.В. Определение оптимального числа граничных маршрутизаторов при организации безопасной многопутевой маршрутизации в мобильной компьютерной сети большой размерности… 73

Ролик А.И. Декомпозиционно-компенсационный подход к управлению уровнем услуг в корпоративных ит-инфраструктурах……. 78

Шовгун Н.В. Підвищення ефективності роботи нечітких нейронних мереж в задачі оцінки кредитоспроможності позичальника…… 89

Полторак В.П., Голков В.Б. Порівняльний аналіз систем електронного цифрового підпису та особливості їх реалізації на еліптичних кривих……. 95

Пустоваров В.И., Бойко В.В., Лемешко В.А.. Функциональные расширения промышленных автоматизированных систем контроля и управления……. 104

Прошин С.Ю. Оцінка матриць переходів енергонавантажень на базі періодичного ланцюга маркова……. 109

Аракелян Г.А., Сергієнко А.М. Визначення зольності вугілля за допомогою модельованого відпалювання 115

Мельник Т.П. Удосконалення технології динамічного моделювання стохастичної системи визначення зон затоплення із врахуванням дамб обвалування…….. 119

Приймак А.В. Оптимизация вычислений на CUDA при моделировании неустойчивости ленгмюровских волн в плазме ……..... 125

Федотова-Пивень И.Н., Пивень О.Б. Программное моделирование совмещенного во времени сложения двадцати целых положительных чисел в избыточной рекуррентной системе счисления третьего порядка. 131

Минухин С.В., Знахур С.В., Коровин А.В. Координированные стратегии планирования заданий на основе обобщенного статистического анализ их характеристик в распределенных вычислительных системах…. 137

Луцкий Г.М. Мухин В.Е. Методы маршрутизации в компьютерных системах с многоканальными связями 148

Стіренко С.Г. Модель технології програмування паралельних систем ..……. 158

Відомості про авторів…… ……. 165

Summary…… 166

УДК 504.052+004.62+523.98+550.34

ЗГУРОВСКИЙ М.З.,
БОЛДАК А.А.,
ЕФРЕМОВ К.В.,
СЕРГЕЕВА Н.А.,
ЗАБАРИНСКАЯ Л.П.,
ШЕСТОПАЛОВ И.П.,
НИСИЛЕВИЧ М.В.

ПРИМЕНЕНИЕ МЕТОДОВ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДАННЫХ ДЛЯ ЭМПИРИЧЕСКИХ ИССЛЕДОВАНИЙ ВЗАИМОСВЯЗИ ГЕЛИО- И ГЕОФИЗИЧЕСКИХ ПРОЦЕССОВ

В работе представлены результаты экспериментального исследования взаимосвязи гелио- и геофизических процессов с использованием методов интеллектуального анализа данных, а именно предложенного авторами метода вейвлет-разложения сигналов в неортогональных базисах.

The paper presents the results of an experimental study on the relationship of helio- and geophysical processes using data mining methods, namely the proposed method on the wavelet decomposition of signals in non-orthogonal bases.

Введение

Проблемы исследования сложных природных систем связаны с отсутствием их формальных описаний, сложностью наблюдения за ней и управления такой системой вследствие присутствия большого количества взаимодействующих второстепенных процессов, нестационарности исследуемых процессов и эволюции в целом исследуемых систем, невозможности проведения активных экспериментов [1].

Одной из таких задач является междисциплинарная задача исследования взаимосвязи гелио- и геофизических процессов, в частности активности Солнца и сейсмической активности Земли.

Существует достаточно обширный ряд публикаций, авторы которых устанавливают наличие связи между геоэффективными солнечными событиями и сейсмичностью Земли.

В ряде работ на простом эмпирическом анализе исторических данных или статистическими методами найдено неслучайное соотношение между солнечно-земным взаимодействием и возникновением землетрясений. Так установлено, что в период 11-летнего цикла солнечной активности наиболее высокая сейсмическая активность соответствует эпохам максимума и минимума этого цикла, когда уровень солнечной активности быстро и резко меняется [2-4]. В работах [5-7] сделан вывод о наличии корреляционной связи между вариациями сейсмической активности и солнечной активностью. Исследования с применением спектрального анализа временных рядов сильных и слабых землетрясений и чисел Вольфа, характеризующих солнечную активность, методом максимальной энтропии показали, что в периоды повышенных 11-летних циклов солнечной активности повышается активность землетрясений в зонах сжатия Земли, а в периоды пониженной солнечной активности повышается активность землетрясений в зонах растяжения Земли [8].

Однако в работе [9] авторы, используя для статистической проверки гипотезы о существовании связи между солнечной и сейсмической активностью Хи-квадрат критерий и t-критерий Стьюдента, не нашли статистически значимых различий в распределениях глобального числа сильных землетрясений ($M \geq 7{,}5$) и числа групп солнечных пятен. То есть нельзя уверенно сказать о статистически значимых отношениях между геоэффективными событиями на Солнце и возникновением землетрясений. Но и отвергнуть их существование также нельзя.

Задача данного прикладного исследования состоит в проверке описанной выше гипотезы на основе применения методов интеллектуального анализа к данным, описывающим активность гелио- и геофизических процессов. В этой работе предлагается метод анализа причинно-следственных связей между процессами, основанный на неортогональном дискретном вейвлет-преобразовании.

Постановка задачи эмпирического исследования причинно-следственных связей между гелио- и геофизическими процессами

В качестве характеристики солнечной активности обычно используется ряд чисел Вольфа. Число Вольфа W вычисляется по формуле:

$$W = k(f + 10g),$$

где f – число пятен на солнечном диске; g – число групп, объединяющих эти пятна; k – нормировочный коэффициент, характеризующий условия наблюдения и применяемую аппаратуру.

Для настоящего исследования в качестве данных, характеризующих солнечную активность, был выбран ряд среднегодовых значений чисел Вольфа за период с 1890 г. по 2012 г [10].

Сейсмическую активность можно характеризовать суммарной сейсмической энергией, высвобожденной при крупных землетрясениях. Энергия землетрясений определяется (в эрг) по формулам [11]:

$$lgE = 11{,}8 + 1{,}5M_s$$

для гипоцентров на глубине меньше 100 км и

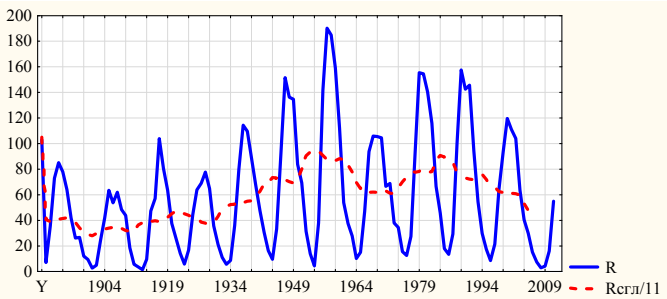
$$lgE = 5{,}8 + 2{,}4m_b$$

для гипоцентров на глубине больше 100 км, где E – энергия землетрясения, M_s – магнитуда по поверхностным волнам, m_b – магнитуда по объемным волнам.

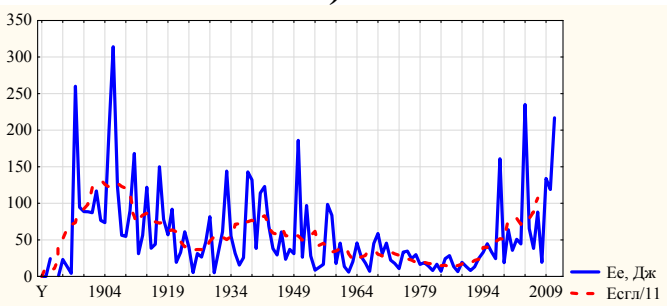
Для расчета суммарных годовых значений сейсмической энергии землетрясений за период 1890 – 2012 гг. на основе каталогов Международного сейсмологического центра [12] и Национального центра информации о землетрясениях Геологической службы США [13] был составлен каталог землетрясений с магнитудами $M_s \geq 6$ и $m_b \geq 5{,}5$.

Для получения однородного ряда данных было проведено их сглаживание в окне шириной 7 лет по трем значениям. Аналогичная процедура предварительной обработки данных была применена к ряду чисел Вольфа. На рис.1 приведены исходные и сглаженные данные о солнечной активности в виде ряда чисел Вольфа R и сейсмической активности в виде выделившейся энергии землетрясений E.

Ранее при изучении взаимосвязи сейсмической активности Земли и активности Солнца исследователями ГЦ РАН были выявлены вековые циклы солнечной и сейсмической активности, продолжительностью около 100 лет [14-18].



а)



б)

Рис.1. Данные, используемые в исследовании взаимосвязи гелио- и геофизических процессов: а) среднегодовые и сглаженные R_{sgl} значения чисел Вольфа: б) суммарные годовые E_e сглаженные $E_{сгл}$ значения сейсмической энергии.

Цель этого исследования состоит в проверке гипотезы о причинно-следственной связи между гелио- и геофизическими процессами на основе интеллектуального анализа охарактеризованных выше данных.

Метод анализа причинно-следственных связей на основе разложения сигналов в неортогональных базисах

Континуальный многомасштабный вейвлет-анализ [19] представляет собой довольно гибкий и, в то же время, весьма мощный инструмент временного и спектрального анализа, целью которого является построение частотно-временных характеристик сигналов, которые во многом проясняют природу не только сигналов в целом, но также и их составляющих.

Вейвлет-преобразование для непрерывного сигнала $x(t)$ относительно вейвлет-функции ψ определяется следующим образом [20]:

$$T(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t)\psi^*\left(\frac{t-b}{a}\right)dt,$$

где ψ^* означает комплексное сопряжение к функции ψ, параметр $b \in R$ соответствует временному сдвигу и называется параметром положения, параметр $a > 0$ задает масштабирование и называется параметром растягивания.

Если определить нормированную функцию:

$$\psi_{a,\mathrm{b}} = \frac{1}{\sqrt{a}}\psi(\frac{t-b}{a}),$$

то вейвлет-преобразование можно записать в виде:

$$T(a,b) = \int_{-\infty}^{\infty} x(t)\psi^*_{a,\mathrm{b}}dt.$$

Гибкость вейвлет-анализа достигается за счет применения различных функций ψ, и, следовательно, различных процедур синтеза вейвлет-спектра сигнала. В соответствии с тем, каким образом формируется базис разложения сигнала (система функций ψ), выделяют неортогональные, ортогональные и биортогональные вейвлет-преобразования. В зависимости от способа представления анализируемых сигналов принято выделять непрерывное и дискретное вейвлет-преобразования.

Предлагаемый метод анализа причинно-следственных связей между процессами основан на неортогональном дискретном вейвлет-преобразовании.

Модель сигналов. Пусть задан набор временных последовательностей $X = \{x_{1..n}\}$ и выдвигается гипотеза о существовании подмножества $X_E = \{x_m \mid m \subset 1..n\}$ последовательностей («причин»), формирующих исходную последовательность («следствие»), которая также входит в X. Необходимо разработать метод проверки такой гипотезы, учитывая то, что входящие последовательности могут присутствовать в исходной последовательности с произвольным сдвигом и масштабом. Также необходимо учитывать тот факт, что в причинно-следственной связи «следствие» не может предшествовать «причине» по времени.

Аддитивную модель сигнала, соответствующую проверяемой гипотезе, можно представить следующим образом:

$$y(t) = \sum_{i=1}^{n} w_i e_i(a,b,t), \quad (1)$$

где w_i – соответствует степени влияния сигнала $e_i(a,b,t)$ на результат, а сигналы $e_i(a,b,t)$ представляют собой масштабированные и смещенные сигналы $x_i \in X_E$:

$$e_i(a_i,b_i,t) = x_i(\frac{t-b_i}{a_i}).$$

Модель (1) является двухуровневой: внутренняя модель описывает преобразования каждого отдельного сигнала x_i, а внешняя – формирование их аддитивной смеси.

Таким образом, для разложения сигнала-«следствия» в базисе сигналов-«причин» необходимо найти такие значения сдвигов b_i и масштабов a_i сигналов x_i, которые максимизируют w_i для всех i.

Определение параметров масштаба и сдвига. Рассмотрим сначала случай, когда сигналы заданы непрерывной моделью.

Пусть задана некоторая функция $f(z)$, которая формирует последовательность значений выборки x, тогда функция $F(a,b,z)$, которая задает масштабирование и сдвиг по z, принимает следующий вид:

$$F(z) = f(\frac{z-b}{a})$$

Учитывая, что результатом вейвлет-анализа является поверхность, точки которой указывают на «сходство» между вейвлет-функцией и сигналом в заданном интервале, можно использовать в качестве вейвлет-функции одну из тех, которые описывают входящие последовательности.

Применим формулу вейвлет-преобразования, заменив дискретные последовательности соответствующими им сигналами с помощью одного из видов интерполяции:

$$\begin{aligned} f(a_k,b_k) &= \sum_{i=1}^{N}\int_{-\infty}^{\infty} w_i x_i\left(\frac{t-b_i}{a_i}\right)x_k\left(\frac{t-b_k}{a_k}\right)dt = \\ &= w_k\int_{-\infty}^{\infty} w_i x_k(\frac{t-\bar{b}_k}{\bar{a}_k})x_k(\frac{t-b_k}{a_k})dt + \\ &+ w_i\sum_{\substack{i=1\\ i\neq k}}^{N}\int_{-\infty}^{\infty} x_i(\frac{t-b_i}{a_i})x_k(\frac{t-b_k}{a_k})dt \end{aligned} \quad (2)$$

В формуле (2) значения $\bar{a},\bar{b}$ соответствуют тем значениям сдвига и масштаба, с которыми последовательность x_k входит в модель (1).

Первое слагаемое в формуле (2) является автокорреляционной функцией [21], значение которой будет максимальным при условии максимальной «похожести» сигналов $e_k(a_k,b_k,t)$ и $e_k(\bar{a}_k,\bar{b}_k,t)$:

$$\max_k(f(a_k,b_k)) = f(\bar{a}_k,\bar{b}_k).$$

Значения функции (2) можно приближенно вычислить с помощью одной из формул численного интегрирования [22]. В случае применения формулы прямоугольников имеем:

$$f(a_k,b_k) = \sum_{i=1}^{N}\int_{-\infty}^{\infty} w_i x_i(\frac{t-b_i}{a_i}) x_k(\frac{t-b_k}{a_k})dt \approx$$

$$\approx \sum_{i=1}^{N}\sum_{t=t_{\min}}^{t_{\max}} w_i x_i(\frac{t-b_i}{a_i}) x_k(\frac{t-b_k}{a_k})\Delta t \qquad (3)$$

где $(t_{\min};t_{\max})$ – пределы интегрирования, соответствующие области, на которой заданы значения последовательности x_k; Δt – шаг интегрирования.

Для приведения значений функции f к диапазону (-1; 1) необходимо нормировать значения, вычисленные по формуле (3), разделив их на произведение энергий функций y_i и x_k:

$$f(a_k,b_k)_{norm} = \frac{\sum_{t=t_{\min}}^{t_{\max}} (y_i x_k(\frac{t-b_k}{a_k})\Delta t)}{\sum_{t=t_{\min}}^{t_{\max}} (y_i^2\Delta t)\sum_{t=t_{\min}}^{t_{\max}} ((x_k(\frac{t-b_k}{a_k}))^2\Delta t)} \qquad (4)$$

Задача вычисления значений w_k, определяющих степень влияния сигналов x_k на результат y, может быть решена с помощью реализации последовательной схемы поиска компонент.

На первом шаге по формуле (4) для каждого сигнала $x_k \in X_E$ осуществляется поиск параметров сдвига и масштаба, максимизирующих w_k. На втором шаге выполняется проверка достоверности значений сдвига и масштаба, которая заключается в нахождении коэффициента w_k с помощью линейной регрессии. На третьем шаге осуществляется вычитание сигнала $e_k(a_k,b_k,t)$ из y. При этом x_k остается в списке исходных последовательностей-«причин», благодаря чему можно обнаружить такие эффекты влияния как «эхо».

Применение предложенного метода для анализа периодических сигналов требует уточнения, связанного с выбором и согласованием размеров временных окон, в пределах которых исследуются сигналы. Окном будем называть отрезок входного сигнала, который включает в себя ровно один период. Теперь задача сравнения последовательностей будет заключаться в сравнении окна входного сигнала с применением к нему соответствующих преобразований с окном выходного сигнала.

Для определения размера окна могут быть применены средства спектрального анализа на основе преобразования Фурье [23]. Однако, в том случае, когда сигнал плохо аппроксимируется гармоническими функциями, такой способ определения размера окна может давать ошибочный результат. Решение задачи определения размера окна для таких сигналов может основываться на анализе их вейвлет-разложения в ортонормированном базисе вейвлет-функций.

Результаты экспериментальных исследований

Предложенный метод реализован в системе обеспечения жизненного цикла данных для научных исследований, разрабатываемой Мировым центром данных «Геоинформатика и устойчивое развитие» [24]. На рис.2 представлено описание этого сценария на языке [25], используемом в этой системе. Такой сценарий представляет собой совокупность взаимосвязанных процессов обработки данных, связь между которыми осуществляется с помощью портов. В представленном сценарии используются компоненты, предназначенные для импорта исходных данных, для экспорта результатов анализа причинно-следственных связей, для выполнения вейвлет-преобразований и преобразований Фурье, для интерполяции и сглаживания данных, а также компонента для анализа взаимосвязи двух исходных сигналов.

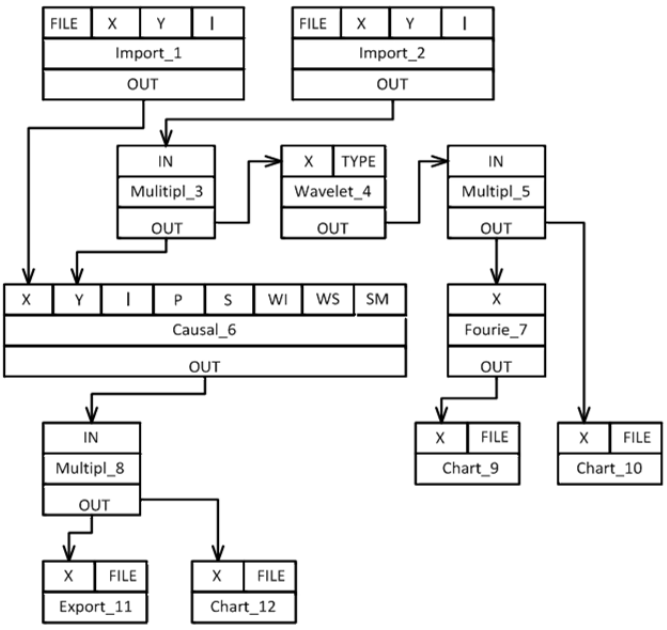
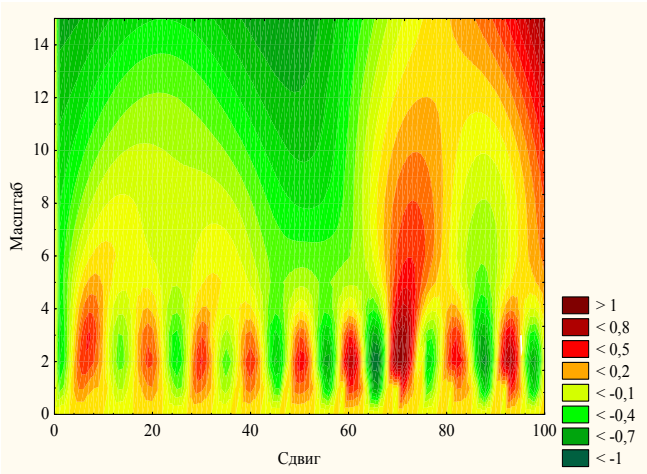


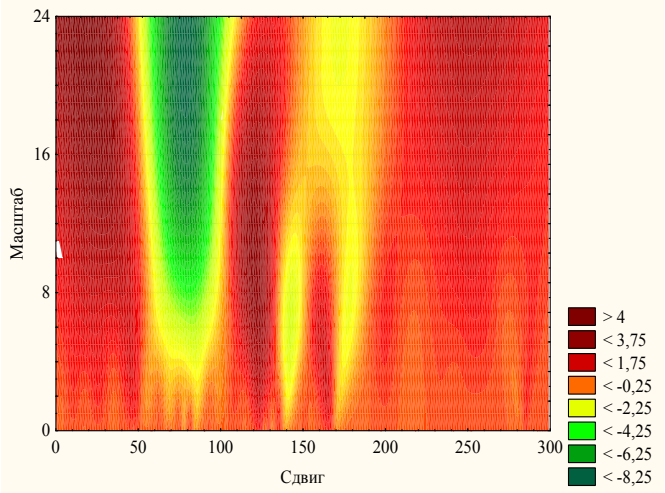
Рис.2. Описание сценария обработки данных для решения задачи анализа взаимосвязи процессов.

Результаты, полученные с помощью разработанных программных средств, представлены ниже.

Как видно из рис.3,а, на котором отображены результаты вейвлет-анализа для достоверного ряда чисел Вольфа, отчетливо прослеживаются известные 11-летние солнечные циклы.



а)



б)

Рис.3. Анализ ряда солнечной активности на периодичность: а) достоверный ряд чисел Вольфа за период 1890-2013 гг.; б) ряд чисел Вольфа за период 1705-2011 гг.

На рис.3,б можно также наблюдать периодичность и на больших масштабах, в частности, циклы длительностью 70-100 лет, что соответствует вековому «циклу Гляйсберга».

Как видно из рис.4, аналогично солнечной активности, для сейсмической активности характерна цикличность. Расстояние между двумя первыми максимумами равно около 100 лет. Третий максимум менее достоверный, потому, что третий период еще не закончился и вейвлет-функция не полностью пересекается с этим периодом, вследствие чего возникает краевой эффект.

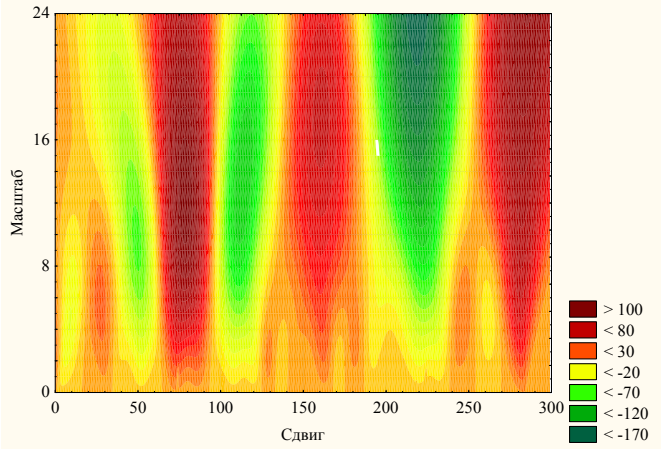


Рис.4. Результат вейвлет-анализа ряда сейсмической активности 1705-2013 гг. на периодичность.

В таблице 1 приведены результаты анализа взаимосвязи гелио- и геофизических процессов. Проанализирована связь между 11-летними циклами солнечной активности и сейсмической активности для достоверных рядов (1890-2013 гг.). Размер окна равен 11 годам, диапазон смещений принимает значение в пределах от 0 до 20 лет с шагом в один год, диапазон масштабов – от 1 до 3 с шагом 0,1.

Таблица 1.

Результаты анализа взаимосвязи процессов

Максимальное сходство	Масштаб	Сдвиг
0,85	1	5
0,804	1,1	0
0,792	1	0
0,813	2,4	4
0,888	1,3	1
0,779	1	3
0,874	1	0
0,926	1,7	0
0,85	1,9	19
0,764	1,3	8

Используя данные таблицы 1, можно дать графическую интерпретацию полученных результатов, изображенную на рис.6.

Как видно из рис.5, во многих случаях максимумам солнечной активности соответствуют максимумы сейсмической активности. Однако также существует много несоответствий.

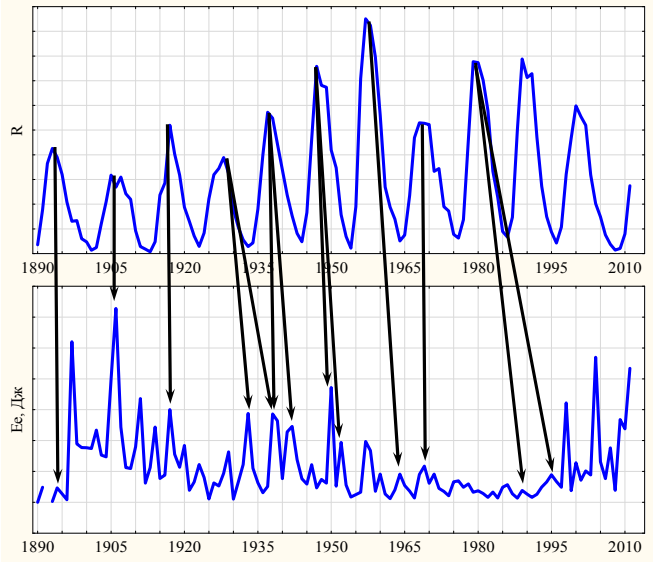


Рис.5. Графическая интерпретация результатов анализа.

Кроме 11-летних циклов также были проанализированы сглаженные по 7 значениям ряды солнечной и сейсмической активности за 1705-2011 гг. Обнаружены похожие фрагменты при значениях сдвига в 20 лет и со значением масштаба 2.2. Эти фрагменты приведены на рис.7:

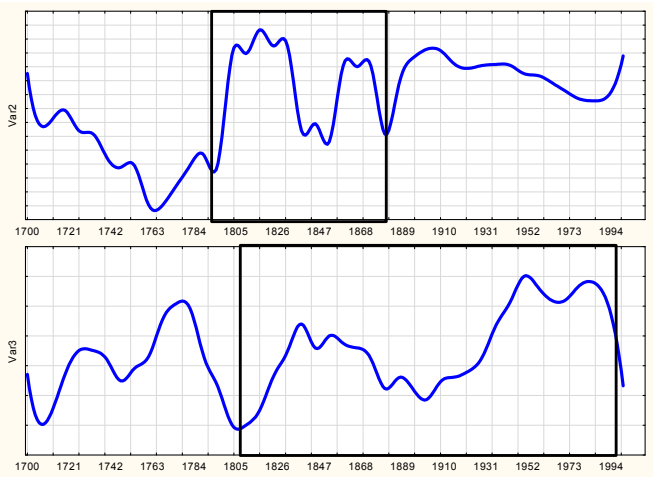


Рис.7. Динамика выделения сейсмической энергии и развития солнечной активности.

Несмотря на то, что замеченное сходство достаточно интересно, оно не позволяет делать однозначные выводы о наличии причинно-следственной связи. В то же время это может свидетельствовать о каких-то общих чертах процессов, которые лежат в основе исследуемых явлений.

Выводы

В результате применения разработанного авторами метода анализа причинно-следственных связей между процессами, основанного на неортогональном дискретном вейвлет-преобразовании, для проверки гипотезы о наличии связи между геоэффективными солнечными событиями и сейсмичностью Земли можно сделать следующие заключения:

Предложенный метод позволил выделить 11-летние циклы солнечной активности на достоверном ряде чисел Вольфа за период 1890-2013 гг. и на ряде чисел Вольфа за период 1705-2011 гг. На ряде чисел Вольфа за период 1705-2011 гг. выделены периоды длительностью 70-100 лет, что соответствует вековому «циклу Гляйсберга».

Анализ показывает, что нельзя говорить об 11-летней периодичности сейсмичности Земли на ряде значений выделившейся сейсмической энергии за период 1890-2013 гг. Установлено, что для сейсмической активности за период 1705-2011 гг. также характерна цикличность с периодом около 100 лет и по времени не совпадающая с цикличностью активности Солнца.

При анализе сглаженных по 7 значениям рядов солнечной и сейсмической активности за 1705-2011 гг. обнаружены похожие фрагменты при значениях сдвига в 20 лет и со значением масштаба 2.2. Но выявленное сходство не позволяет делать однозначные выводы о наличии причинно-следственной связи между геоэффективными солнечными событиями и сейсмической активностью Земли.

Работа выполнена при финансовой поддержке Национальной академии наук Украины (2268/12, 2268/13) и Российского фонда фундаментальных исследований (грант № 12-01-90418-Укр_а).

Список литературы

1. Растригин Л. А. Адаптация сложных систем. — Рига: Зинатне, 1981. — 375 с.
2. Сытинский А.Д. Зависимость сейсмичности Земли от процессов на Солнце, в межпланетной среде и в атмосфере. В кн. Атлас временных вариаций природных антропогенных и социальных процессов. Том 2. М.: Научный Мир, 1998, с.70-72.
3. Одинцов С.Д., Иванов-Холодный Г.С., Георгиева К. Солнечная активность и глобальная сейсмичность Земли // Известия РАН. Серия физическая, 2007. Т. 71. № 4. С. 608-610.

4. Simpson I.F. Solar activity as a triggering mechanism for earthquakes // Earth and Planet. Sci. letter, 1968. V. 3. № 5. P. 417-425.
5. Ананьин И.В., Фаддеев А.О. О возможных причинах корреляций между изменениями величин сейсмической активности и средними годовыми температурами на поверхности Земли. В кн. Атлас Временных вариаций природных, антропогенных и социальных процессов. Том 3. М., Янус-К, 2002, с.222-224.
6. Шестопалов И.П., Рогожин Ю.А. Корреляция между микробиологической и сейсмической активностью с учетом взаимосвязей «Солнце-Земля» и генерации нейтронных потоков // Авиакосмическая и экологическая медицина. 2005. Т. 39. № 3.С. 20-26.
7. Gousheva, M.N., Georgiva, K.Y., Kirov, B.B., Atanssov, D. On the relation between solar activity and seismicity // Proceedings of International Conference on Recent Advances in Space Technologies, 20-22 November 2003.
8. Хаин В.Е., Халилов Э.Н. Пространственно-временные закономерности сейсмической
9. и вулканической активности. Burgas, SWB, 2008. 304 с. ISBN 978-9952-451-00-9.
10. SIDC – Solar Influences Data Analysis Center [Электронный ресурс] – 2013 –Режим доступа: http://sidc.oma.be/.
11. Касахара К. Механика землетрясений. Пер. с англ. Москва. Мир. 1985. 264 с.
12. International Seismological Centre [Электронный ресурс] – 2013 –Режим доступа: http://www.isc.uk.
13. National Earthquake Information Center – NEIC [Электронный ресурс] – 2013 –Режим доступа: http://neic.usgs.gov.
14. Соболев Г.А., Шестопалов И.П., Харин Е.П. Геоэффективные солнечные вспышки и сейсмическая активность Земли // Физика Земли. 1998. № 7. С. 85-89.
15. Шестопалов И.П., Харин Е.П. Изменчивость во времени связей сейсмичности Земли с циклами солнечной активности различной длительности // Геофизический журнал. 2006. Т. 28. № 4. С. 59-70.
16. Шестопалов И.П., Харин Е.П. Сейсмическая активность Земли и процессы в межпланетной среде // IV Международная конференция “Солнечно-земные связи и предвестники землетрясений”: с. Паратунка Камч. обл. 14-17 авг. 2007 г., сборник докладов. Петропавловск-Камч.: ИКИР ДВО РАН, 2007. С.476-481.
17. Белов С.В., Шестопалов И.П., Харин Е.П. О взаимосвязях эндогенной активности Земли с солнечной и геомагнитной активностью // Доклады Академии наук. 2009. Т. 428. № 1. С. 104-108.
18. Харин Е.П., Белов С.В., Шестопалов И.П. Пространственно-временные изменения сейсмичности Земли и солнечная активность // V Международная конференция “Солнечно-земные связи и физика предвестников землетрясений”, с. Паратунка, Камчатский край. 2-7 авг. 2010 г., сборник докладов. Петропавловск-Камчатский: ИКИР ДВО РАН, 2010. С. 470-473.
19. B. Boashash. Time-Frequency Signal Analysis and Processing – A Comprehensive Reference [Text] / B. Boashash. Oxford, UK: Elsevier Science, 2003.
20. Raghuveer M. Wavelet Transforms: Introduction to Theory & Applications [Text] / Raghuveer M. Rao, Ajit S. Bopardikar. Upper Saddle River, New Jersey, US: Prentice Hall PTR, 2008. – 336 p.
21. Орлов А.И. Прикладная статистика [Текст] / Орлов А.И. – М.: Издательство «Экзамен», 2011. – 425 с.
22. Каханер Д. Численные методы и программное обеспечение [Text] / Каханер Д., Моулер К., Нэш С. – М.: Мир, 2011. – 575 с. – ISBN 5-03-003392-0.
23. М. А. Павлейно. Спектральные преобразования в MatLab. [Text] / М. А. Павлейно, В. М. Ромаданов. – СПб., 2007. – 160 с. – ISBN 978-5-983-40121-1.
24. World Data Center for Geoinformatics and Sustainable Development [Электронный ресурс] – 2013 –Режим доступа: http://wdc.org.ua.
25. Болдак А.О. , Єфремов К.В. Предметно-орієнтована мова аналітичної обробки даних[Текст] // Вісник НТУУ "КПІ". Інформатика, управління та обчислювальна техніка: Зб. наук. пр. - К.: Век+, - 2012. - № 55. - с.46-54.

УДК 519.854.2

ПАВЛОВ А.А.,
МИСЮРА Е.Б.,
ЛИСЕЦКИЙ Т.Н.,
СПЕРКАЧ М.О.,
ХАЛУС Е.А.

ЧЕТЫРЕХУРОВНЕВАЯ МОДЕЛЬ ПЛАНИРОВАНИЯ, ПРИНЯТИЯ РЕШЕНИЙ И ОПЕРАТИВНОГО УПРАВЛЕНИЯ В СЕТЕВЫХ СИСТЕМАХ С ОГРАНИЧЕННЫМИ РЕСУРСАМИ

В статье рассмотрена четырехуровневая модель планирования и оперативного управления в системах с сетевым представлением технологических процессов, включающая: построение агрегированной модели; построение согласованного плана выполнения агрегированных работ в соответствии с 31 критерием оптимальности; выбор наилучшего плана в блоке принятия решений; точное планирование; оперативное управление в случае частичного невыполнения полученного плана. Третий уровень модели формализован на основе оригинальной методологии как многоэтапная сетевая задача календарного планирования. Приведена общая методология реализации алгоритмического обеспечения четырехуровневой модели.

In the article the four-level planning and operational control model in systems with a network representation of the technological processes is considered. The model includes construction of the aggregated model; construction of the coordinated plan of the aggregated jobs execution according to 31 criteria of optimality; choosing of the best plan in the decision-making block; exact planning; operational control in case of partial fail of the agreed plan execution. The third level of model is formalized on the basis of original methodology as a multi-stage network scheduling problem. The general methodology for realization of the four-level model algorithmic maintenance is given.

Введение

В [1] изложена методология построения оперативных планов для организационно-социальных систем с сетевым представлением технологических процессов. Рассмотрена трехуровневая система планирования по 31 критерию оптимальности, включающая в себя блок принятия решений. Построение агрегированных планов на 1 и 2 этапе [1] позволяет после реализации блока принятия решений сформировать окончательный портфель заказов, после чего на третьем этапе строится результирующий пооперационный оперативный план.

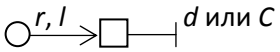
В статье [2] приводится методология создания формальной модели построения результирующего пооперационного плана, которая может быть сформулирована как специальная сетевая многоэтапная задача календарного планирования, позволяющая построить эффективный алгоритм ее решения, методология построения которого изложена в [3].

В данной статье представлена четырехуровневая модель планирования, принятия решений и оперативного управления, являющаяся обобщением модели планирования, изложенной в [1]. Третьим уровнем модели является многоэтапная сетевая задача календарного планирования, которая обобщает многоэтапную сетевую задачу, изложенную в [3]. Четвертый уровень предназначен для оперативного управления – оперативной коррекции построенного на третьем уровне плана, в случае его частичного невыполнения. Приведена общая схема реализации алгоритмического обеспечения модели.

Сетевая модель задачи

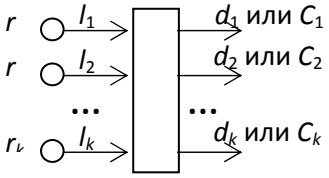
Рассмотрим базовые элементы, на основе которых синтезируется исходная сетевая модель исследуемой многоэтапной задачи.

Элемент типа 1.



○ – обозначение работы, □ – обозначение прибора, на котором выполняется работа, l – длительность выполнения работы, r – момент времени готовности работы к выполнению на приборе, C – момент окончания выполнения работы, d – директивный срок работы. В допустимом расписании $C \le d$. Символом ⊣ на сети обозначается окончание выполнения конечной работы (не имеющей преемников).

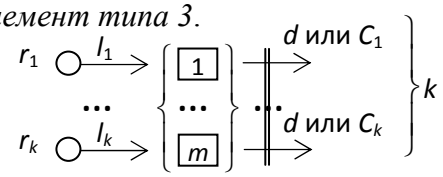
Элемент типа 2.



Прибор последовательно выполняет k работ. r_i – моменты готовности выполнения работ на

приборе, d_i – директивный срок выполнения i-й работы на приборе. В допустимом расписании $C_i \le d_i$. Прибор работает без прерываний.

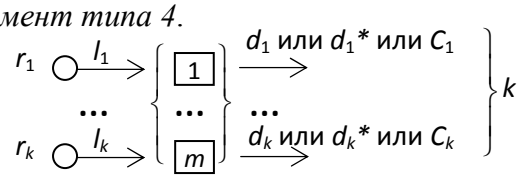
Элемент типа 3.



m независимых параллельных приборов выполняют k работ, которые должны быть выполнены к одному директивному сроку d. В допустимом расписании $C_i \le d$, $i = \overline{1,k}$. Приборы работают без прерываний. Моменты запуска приборов одинаковы, либо произвольны, либо заданы интервалы времени между моментами запуска приборов.

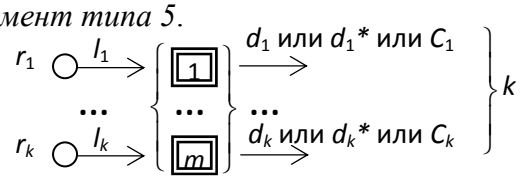
Элемент типа 3 соответствует случаю, когда работы, в силу больших временных затрат, выполняются не на одном, а на нескольких независимо работающих приборах равной производительности. При этом все работы одновременно должны перейти на следующую операцию.

Элемент типа 4.



m независимых параллельных приборов *равной* производительности, работающих без прерываний, выполняют k работ, которые должны быть выполнены к различным директивным срокам d_i. $d_i^* = [d_i - \Delta_i, d_i]$, $i = \overline{1,k}$, определяет диапазон получения допустимого решения: $C_i \le d_i$ либо $C_i \in d_i^*$, $\Delta_i > 0$. Моменты запуска приборов могут быть различны.

Элемент типа 5.



m независимых параллельных приборов *различной* производительности, работающих без прерываний, выполняют k работ, которые должны быть выполнены к различным директивным срокам d_i. $d_i^* = [d_i - \Delta_i, d_i]$, $i = \overline{1,k}$, определяет диапазон получения допустимого решения: $C_i \le d_i$ либо $C_i \in d_i^*$, $\Delta_i > 0$. Моменты запуска приборов могут быть различны.

Предметом исследования в данной статье является следующий класс сетевых моделей:

1. Сеть состоит из комбинации элементов (тип 1–5) и является ориентированным ацикличным графом.

Элементам (тип 2–5) соответствуют в сети приборы, которые в процессе непрерывной работы в произвольном порядке должны выполнить множество работ без прерывания. В произвольный момент времени может выполняться на одном элементе только одна работа. Элементы отличаются между собой типами связи в сети работ, выполненных приборами. Связи могут включать сборку одной работы из нескольких других и разузловку одной работы на несколько других. Элементу (тип 1) соответствует прибор, выполняющий только одну работу.

2. Сама сеть представляет собой ориентированный граф с двумя типами вершин – ○ – обозначает работу готовую к выполнению, за ней обязательно следует вершина □ – обозначение прибора, на котором выполняется непосредственно предшествующая ему работа.

3. Все ориентированные стрелки направлены слева направо.

4. Всю сеть можно разбить на три части.

4.1 Конечная часть сети.

Конечная часть может заканчиваться:

а) элементами типа 1: ○→□⊣C_i, где C_i – момент окончания выполнения последней работы (не имеющей преемников).

б) работой, выполненной на элементе (тип 2–5).

Пусть d_i – директивный срок i-й работы. В допустимом расписании для работ, непосредственно связанных с элементами типа 1, 2, 3, должно выполняться $C_i \le d_i$, а для работ, непосредственно связанных с элементами типа 4, 5, $C_i \in d_i^*$.

Конечная часть сети представляет собой произвольную комбинацию элементов типа 1–5, в которой удовлетворяется следующее условие: d_i однозначно задают для элементов типа 1–3 директивные сроки выполнения работ, а d_i^* однозначно задают для элементов типа 4, 5 диапазоны получения допустимого решения. Работы в произвольном порядке выполняются на элементах типа 2–5, у каждого из них в конечной части сети нет непосредственно предшествующих элементов типа 2–5 (без учета элементов типа 1).

4.2 Промежуточная часть сети.

Это произвольная сетевая комбинация элементов (тип 1–5), удовлетворяющая условию: наиболее поздние моменты готовности выполнения работ на элементах (типа 2–5) однозначно задают директивные сроки выполнения работ для непосредственно предшествующих (без учета элементов (типа 1)) элементов (типа 2–5). Непосредственное предшествование означает, что между непосредственно предшествующим элементом

(типа 2–5) и данным элементом (типа 2–5), существует ориентированный путь, содержащий только вершины соответствующие элементам (тип 1).

В промежуточной части сети могут находиться элементы (типа 2–5), непосредственно предшествующие более чем одному элементу (тип 2–5).

4.3 Начальная часть сети.

Начальная часть сети является произвольной комбинацией элементов (тип 1–5), удовлетворяющая следующим условиям:

а) имеются все вершины сети вида $\boxed{1_j}$, $j=\overline{1,p}$, где цифра 1 означает, что вершине $\boxed{1_j}$ не предшествует ни одна вершина второго типа (см. п. 2).

б) произвольному элементу (тип 2–5) непосредственно не предшествует элемент (тип 2–5).

в) рассмотрим в сети полную группу элементов (тип 2–5). Моменты начала выполнения работ элементов этой группы однозначно задают моменты начала выполнения работ всех элементов типа $\boxed{1_j}$, $j=\overline{1,p}$.

Примечания:

1) Сеть точно отражает реальный технологический процесс.

2) Так как выполнение работ может быть разделено во времени, в сети одни и те же физически существующие приборы могут быть представлены различными вершинами второго типа.

Требование работы приборов без прерываний для элементов (тип 2–5) не всегда является жестким: если одни и те же приборы должны выполнять различные группы работ, которые (заранее известно) разнесены по времени, то в результирующей модели сетевой многоэтапной задачи календарного планирования реально существующие приборы (группы приборов) заменяются работающим без прерываний виртуальным множеством приборов (групп приборов).

Это предположение позволяет исключить в рассматриваемой сети следующую конфигурацию: для любого из элементов типа 2–5 не может быть ситуаций – выходная работа (работы) с точностью до элемента типа 1 непосредственно связаны с конечной работой сети, а остальные работы с точностью до элемента типа 1 непосредственно связаны с входными работами элементов типа 2–5. То есть физически существующее оборудование разбивается на виртуальные, часть из которых находится в начальной и промежуточной части сети, а часть – в конечной.

На рисунке 1 приведен пример сети, принадлежащей сформулированному классу и содержащей все 5 типов элементов.

Общая схема реализации алгоритмического обеспечения четырехуровневой модели планирования, принятия решений и оперативного управления в сетевых системах

Алгоритмическое обеспечение сетевой мо-

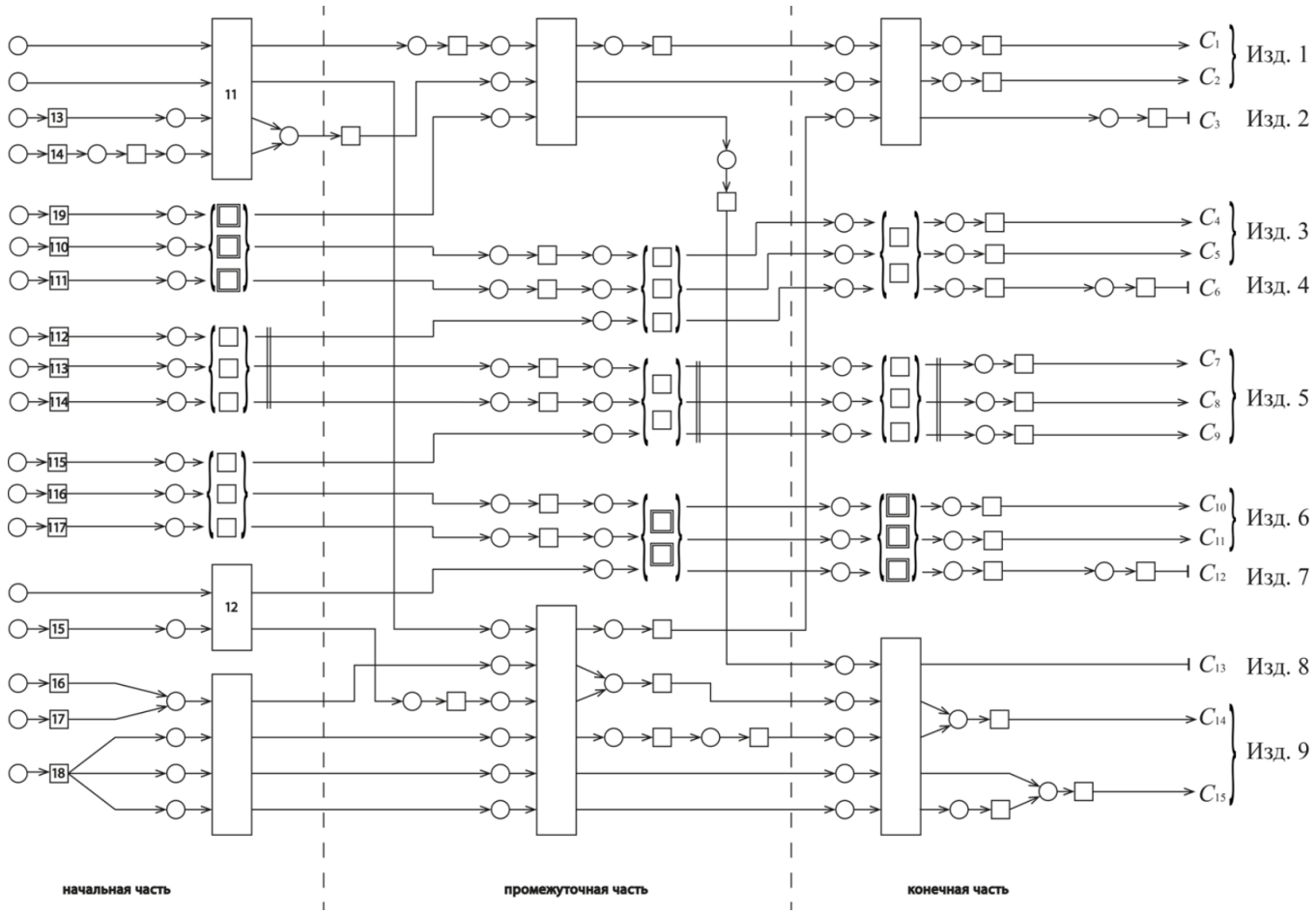


Рис. 1. Пример исходной сети

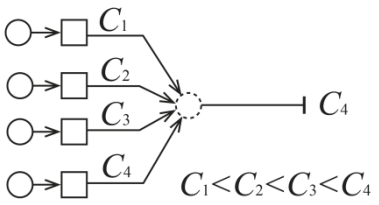
дели состоит из пяти основных блоков. Приведем их краткое описание.

Блок 1. Построение трехуровневой агрегированной модели планирования:

1) *первый уровень* – построение технологической сети, задающей технологию выполнения изделий множества I (под изделием можем пониматься серия однотипных изделий), задание 31 критерия оптимальности (пяти базовых критериев и синтетических критериев, являющихся всеми возможными линейными свертками базовых). Структура сети описана в разделе статьи «Сетевая модель задачи».

Так как приведенные ниже функционалы однозначно задаются моментами окончания выполнения изделий (серий изделий), то конечным работам должен соответствовать момент окончания выполнения изделий, а в исходной сети выполнению изделий (серий изделий) может соответствовать множество конечных вершин. В этом случае эти работы объединяются в одну фиктивную работу, для которой назначается директивный срок, соответствующий директивному сроку выполнения изделия. Длительность выполнения фиктивной работы равна нулю. На рис. 2 приведена сеть, в которой конечные работы объединены в фиктивные работы, показанные вершинами ◌.

Примечание. Технологический процесс выполнения изделия однозначно должен позволять определить конечную работу, которая гарантированно выполняется позже остальных. Все конечные вершины, входящие в фиктивную работу, кроме самой поздней, исключаются из рассмотрения при планировании, т.к. заведомо выполняются до директивного срока, например:



где C_i – момент завершения работы i.

2) *второй уровень* агрегированной модели планирования: а) агрегация работ и приборов;

б) 31 критерий оптимальности остаются неизменными;

в) определение критических путей изделий.

3) *третий уровень* – представление системы планирования в виде одного прибора: построение графа на критических путях изделий с общими вершинами (агрегированные работы, принадлежащие разным критическим путям и выполняемые в рамках одного захода в ресурс, объединяются в общие вершины, если такое

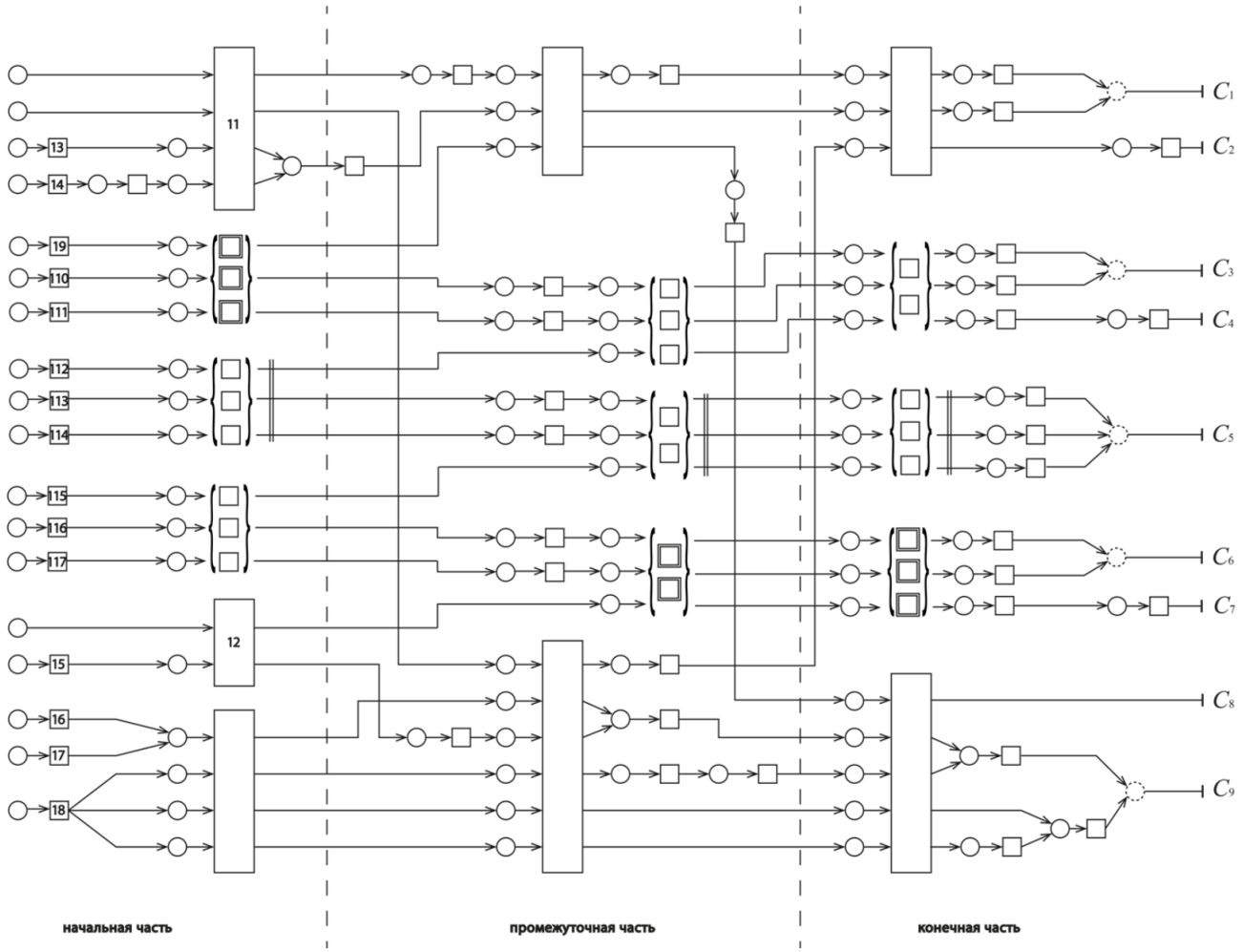


Рис. 2. Сеть первого уровня с фиктивными работами

объединение, определяемое технологией изготовления изделий, позволяет сократить длительность прохождения изделий в системе); 31 критерий оптимальности остаются неизменными.

Блок 2. Получение оптимального расписания по модели планирования третьего уровня. Значение функционала однозначно определяется моментами окончания выполнения изделий (серий изделий):

а) для каждого из пяти базовых критериев построение задачи минимизации суммарного взвешенного момента окончания выполнения заданий (МВМ) [1], из которых четыре аппроксимируют базовые критерии; для каждого синтетического критерия построение соответствующей задачи МВМ для каждой базовой составляющей синтетического критерия, кроме базового критерия 1.

б) решение для каждого из критериев соответствующего количества задач МВМ (по числу базовых составляющих синтетического критерия) на графе на критических путях, получение в результате решения ряда приоритетно-упорядоченных последовательностей выполнения агрегированных работ;

в) построение согласованного плана выполнения работ на втором уровне агрегированной модели в соответствии с 31 критерием оптимальности, значение функционала однозначно определяется моментами окончания изготовления изделий (серий изделий). Если планирование осуществляется по синтетическому критерию оптимальности, являющемуся линейной сверткой нескольких базовых критериев, то все изделия (серии изделий) разбиваются на группы, каждая из которых имеет общую базовую составляющую критерия оптимальности (пример будет приведен ниже). На этапе согласованного планирования выполнения изделий (серий изделий) группы работ выполняются последовательно. При этом рассматриваются все возможные комбинации очередности выполнения групп изделий (серий изделий). В рамках одной группы порядок определяется приоритетно-упорядоченной последовательностью, полученной в результате решения задачи МВМ, соответствующей этой группе.

Блок 3 – блок принятия решений, в котором осуществляется анализ полученных планов по разным критериям оптимальности, согласование с заказчиком всех необходимых изменений. В случае изменения портфеля заказов (корректировка сроков выполнения изделий (серий изделий), изменение объема заказов, исключение отдельных заказов) генерация нового согласованного плана (блоки 1 и 2). В блоке принятия решений могут генерироваться различные варианты портфеля заказов на планируемый период. В этом случае выбирается лучший из них по критерию минимизации риска неполучения прибыли [1].

Блок 4. 1) Дезагрегация агрегированных работ.

2) По выбранному в блоке 3 согласованному плану построение сетевой многоэтапной задачи календарного планирования в соответствии с общим критерием оптимальности, указанным ниже;

3) решение построенной сетевой многоэтапной задачи календарного планирования.

Блок 5. Оперативное управление – оперативная коррекция построенного на третьем уровне плана в случае его частичного невыполнения.

Рассмотрим общую схему более подробно.

Блок 1. Задача ***первого уровня агрегированной модели*** задается сетью, в которой конечные работы объединены в фиктивные работы (см. рис. 2). Необходимо построить распределение (согласованный план) выполнения изделий (серий изделий) по ресурсам с учетом критериев оптимальности, указанных ниже.

Базовые критерии:

Критерий 1. Максимизация суммарной прибыли системы планирования в случае отсутствия директивных сроков.

В обеспечении прибыльности системы планирования важное значение играет фактор времени. В выигрыше будет тот, кто обеспечивает максимально быстрое выполнение заказов и сокращение времени выхода на рынок новых товаров. При отсутствии директивных сроков прибыль от реализации i-го изделия (i-й серии изделий) является функцией времени и равна $P_i(t) = \omega_i(T) \cdot (T - C_i)$, где $\omega_i(T)$ – весовой коэффициент i-го изделия (i-й серии изделий), определенный экспериментальным путем; T – плановый период; $C_i \le T$ – момент окончания выполнения i-го изделия (i-й серии изделий), соответствующий моменту окончания выполнения его конечной вершины. Критерий максимизации суммарной прибыли системы планирования в этом случае определяется выражением

$$F_1 = \sum_{i=1}^{n} P_i(t) = \sum_{i=1}^{n} \omega_i(T) \cdot (T - C_i) + P - З = \\ = T \cdot \sum_{i=1}^{n} \omega_i(T) - \sum_{i=1}^{n} \omega_i(T) \cdot C_i + P - З \to \max, \quad (1)$$

где P – гарантированный минимальный доход от продажи (выполнения) всех n изделий (серий изделий), $З$ – все издержки. Максимизируемая функция имеет вид:

$$\max \left\{ \sum_{i=1}^{n} \omega_i(T) \cdot (T - C_i) \right\} + P - З .$$

Таким образом, *критерий 1* имеет вид:

$$\min \sum_{i=1}^{n} \omega_i(T) \cdot C_i .$$

Критерий F_1 эквивалентен критерию минимизации суммарного взвешенного момента окончания выполнения изделий (МВМ) при отношении порядка, заданным графом, построенным на критических путях изделий.

Критерий 2. Максимизация суммарной прибыли системы планирования при условии: для всех изделий $i \in I$ введены директивные сроки d_i, которые не могут быть нарушены (планирование «точно в срок»):

$$\max\left\{\sum_{i=1}^{n} \omega_i U_i\right\} - З, \text{ где } U_i = \begin{cases} 1, C_i = d_i \\ 0, C_i \neq d_i \end{cases},$$

ω_i – прибыль от выполнения i-го изделия, если оно выполнено точно в срок, $З$ – все издержки. Таким образом, *критерий 2* имеет вид:

$$\max\left\{\sum_{i=1}^{n} \omega_i U_i\right\}.$$

Критерий 3. Максимизация суммарной прибыли системы планирования при условии: для всех изделий $i \in I$ введены директивные сроки d_i, необходимо минимизировать суммарное взвешенное запаздывание выполнения изделий относительно директивных сроков:

$$\max\left\{P - \sum_{i=1}^{n} \omega_i \max(0, C_i - d_i)\right\} - З,$$

где P – гарантированный минимальный доход от продажи (выполнения) всех n изделий, если все они выполнены без запаздывания, $З$ – все издержки; ω_i – штраф за запаздывание окончания выполнения i-го изделия относительно директивного срока на единицу времени. Таким образом, *критерий 3* имеет вид:

$$\min\left\{\sum_{i=1}^{n} \omega_i \max(0, C_i - d_i)\right\}.$$

Величина $\omega_i \max(0, C_i - d_i)$ – уменьшение дохода P в случае выполнения изделия i с запаздыванием $C_i - d_i$. Решение по выполнению или отказу от выполнения таких изделий принимается в блоке принятия решений.

Критерий 4. Для всех изделий $i \in I$ введены директивные сроки d_i. Для каждого изделия указана величина ω_i – абсолютная прибыль от выполнения изделия, не зависящая от момента окончания выполнения изделия в том случае, если изделие выполняется без запаздывания относительно директивного срока, иначе прибыль системы планирования по этому изделию равна нулю. Задача – максимизировать суммарную прибыль системы планирования:

$$\max\left\{\sum_{i=1}^{n} \omega_i U_i\right\} - З, \text{ где } U_i = \begin{cases} 1, C_i \leq d_i \\ 0, C_i > d_i \end{cases},$$

где ω_i – прибыль от выполнения i-го изделия, если оно выполнено без запаздывания относительно директивного срока; $З$ – риск уменьшения прибыли из-за срыва выполнения изделия в срок. Таким образом, *критерий 4* имеет вид:

$$\max\left\{\sum_{i=1}^{n} \omega_i U_i\right\}.$$

Критерий 5. Для всех изделий заданы директивные сроки d_i. Необходимо минимизировать суммарный штраф системы планирования, как за опережение, так и за запаздывание относительно директивных сроков:

$$\max\left\{P - \sum_{i=1}^{n} \omega_i \left|C_i - d_i\right|\right\} - З,$$

где P – гарантированный минимальный доход от продажи (выполнения) всех n изделий (изделий), если все они выполнены без опережения и запаздывания, $З$ – все издержки; ω_i – штраф за отклонение момента окончания выполнения i-го изделия от директивного срока на единицу времени. Таким образом, *критерий 5* имеет вид:

$$\min \sum_{i=1}^{n} \omega_i \left|C_i - d_i\right|.$$

Величина $\omega_i |C_i - d_i|$ – уменьшение дохода P в случае выполнения изделия i с запаздыванием $C_i - d_i$. Решение по выполнению или отказу от выполнения этих изделий принимается в блоке принятия решений.

Синтетические критерии (линейная свертка всех возможных комбинаций пяти базовых критериев). Приведем из 26 возможных критериев этого типа два примера:

Критерий 6. Максимизация суммарной прибыли системы планирования при условии: для некоторых изделий $i \in \overline{1,k}$ заданы директивные сроки, которые не могут быть нарушены, для остальных изделий $d_i = 0$:

$$\max\left\{\sum_{i=1}^{k} \omega_i U_i + \sum_{i=k+1}^{n} \omega_i(T) \cdot (T - C_i)\right\} + P - З,$$

где $U_i = \begin{cases} 1, C_i = d_i \\ 0, C_i \neq d_i \end{cases}$, P – гарантированный минимальный доход от продажи (выполнения) изделий (изделий) $i = \overline{k+1, n}$, $З$ – все издержки; ω_i – прибыль от выполнения i-го изделия, если оно выполнено точно в срок; $\omega_i(T)$ – весовой коэффициент изделия i (имеет тот же смысл, что и в критерии 1). Таким образом, *критерий 6* имеет вид:

$$\max\left\{\sum_{i=1}^{k}\omega_i U_i + \sum_{i=k+1}^{n}\omega_i(T)\cdot(T-C_i)\right\}.$$

Критерий 7. Соответствует критерию 3 с дополнительным условием: для некоторых изделий $i \in \overline{1,k}$ директивные сроки не могут быть нарушены:

$$\max\left\{\sum_{i=1}^{k}\omega_i' U_i - \sum_{i=k+1}^{n}\omega_i'' \max(0, C_i - d_i)\right\} + P - З,$$

где $U_i = \begin{cases}1, C_i = d_i\\ 0, C_i \neq d_i\end{cases}$, ω_i' – прибыль от выполнения i-го изделия, если оно выполнено точно в срок; ω_i'' – штраф за запаздывание окончания выполнения i-го изделия относительно директивного срока на единицу времени, P – гарантированный минимальный доход от выполнения изделий $i = \overline{k+1,n}$, если они выполнены в срок, $З$ – все издержки. Таким образом, *критерий 7* имеет вид:

$$\max\left\{\sum_{i=1}^{k}\omega_i' U_i - \sum_{i=k+1}^{n}\omega_i'' \max(0, C_i - d_i)\right\}.$$

Величина $\omega_i''\max(0, C_i - d_i)$ – уменьшение дохода P в случае выполнения изделия i с запаздыванием $C_i - d_i$. Решение по выполнению или отказу от выполнения таких изделий принимается в блоке принятия решений.

Второй уровень агрегированной модели является агрегированным представлением первого уровня. Агрегирование с целью уменьшения размерности исходной задачи осуществляется с помощью построения агрегированных работ и ресурсов. В результате выполнения агрегации свойства модели сохраняются. Конечные работы первого уровня агрегированной модели совпадают с конечными работами второго уровня.

В соответствии с [1], агрегированной работой называется совокупность работ, выполняемых по одному изделию в рамках одного захода в агрегированный ресурс (агрегированные ресурсы на схеме отображаются элементами типа 1 и 2). Длительность выполнения агрегированной работы определяется: по элементам типа 2 – суммой длительностей работ, входящих в ее состав; по элементам типа 3, 4 – суммой длительностей работ, входящих в ее состав, деленной на количество приборов, входящих в состав элемента; по элементу типа 5 приведенная длительность каждой работы, входящей в состав агрегированной работы, определяется по формуле [4, с. 194] $l = 1\Big/\sum_{j=1}^{m}\left(1/l_j\right)$, где l_j – длительность работы на приборе j.

Необходимо построить согласованный план выполнения комплексов агрегированных работ ресурсами с учетом 31 указанных выше критериев оптимальности при ограничениях:

- прерывания агрегированных работ при выполнении запрещены;
- агрегированная работа не передается в другие ресурсы до ее полного завершения.

Сеть в агрегированном виде приведена на рис. 3. Несмотря на меньшую размерность задачи второго уровня, она все же остается труднорешаемой.

Третий уровень агрегированной модели (блок 1) отвечает уровню, при котором система планирования представляется в виде одного прибора.

Для упрощения модели второго уровня вводятся следующие ограничения:

- длительность выполнения каждого изделия определяется его критическим путем;
- общие агрегированные работы разных изделий лежат на их критических путях и выполняются в одном ресурсе (элементе).

Таким образом, для каждого изделия определяется его критический путь – маршрут максимальной длительности от начала изготовления изделия до его полного выполнения, строится по агрегированным работам (жирные линии на рис. 3).

На основе критических путей изделий и набора общих вершин строится ориентированный ацикличный граф, называемый графом на критических путях (рис. 4). Длительность выполнения общей вершины определяется суммой длительностей агрегированных работ, входящих в ее состав. На рис. 4 общие вершины отображены символом ◎.

Требуется решить задачу на графе на критических путях изделий по 31 критерию оптимальности. Рассмотрим критерий 1.

1) На критических путях изделий отсутствуют общие вершины. В соответствии с утверждением 9.1 [1], минимальное значение суммарной взвешенной длительности прохождения (взвешенной длительности производственного цикла – в наших терминах $\min\sum_{i=1}^{n}\omega_i\cdot C_i$) независимых изделий в системе с одним прибором достигается при расписании, для которого выполняется $\omega_1/l_1 \geq \omega_2/l_2 \geq \ldots \geq \omega_n/l_n$, где ω_i и l_i – соответственно вес и длительность i-го по порядку изделия в оптимальном расписании. Следовательно, задача третьего уровня агрегированной модели решается простым упорядочением изделий по невозрастанию их приоритетов, определяемых отношением веса изделия к длине его критического пути.

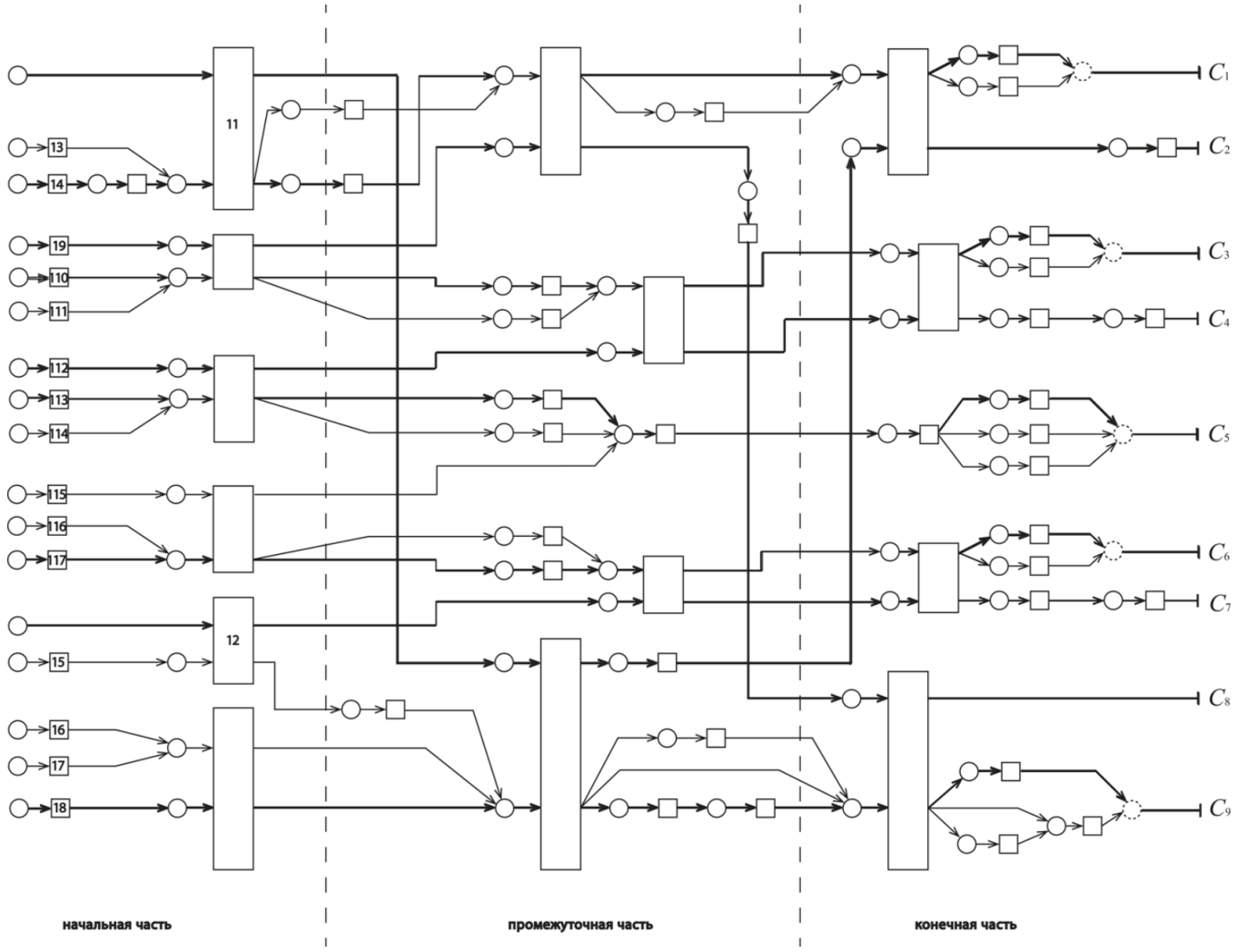


Рис. 3. Агрегированное представление сети на втором уровне

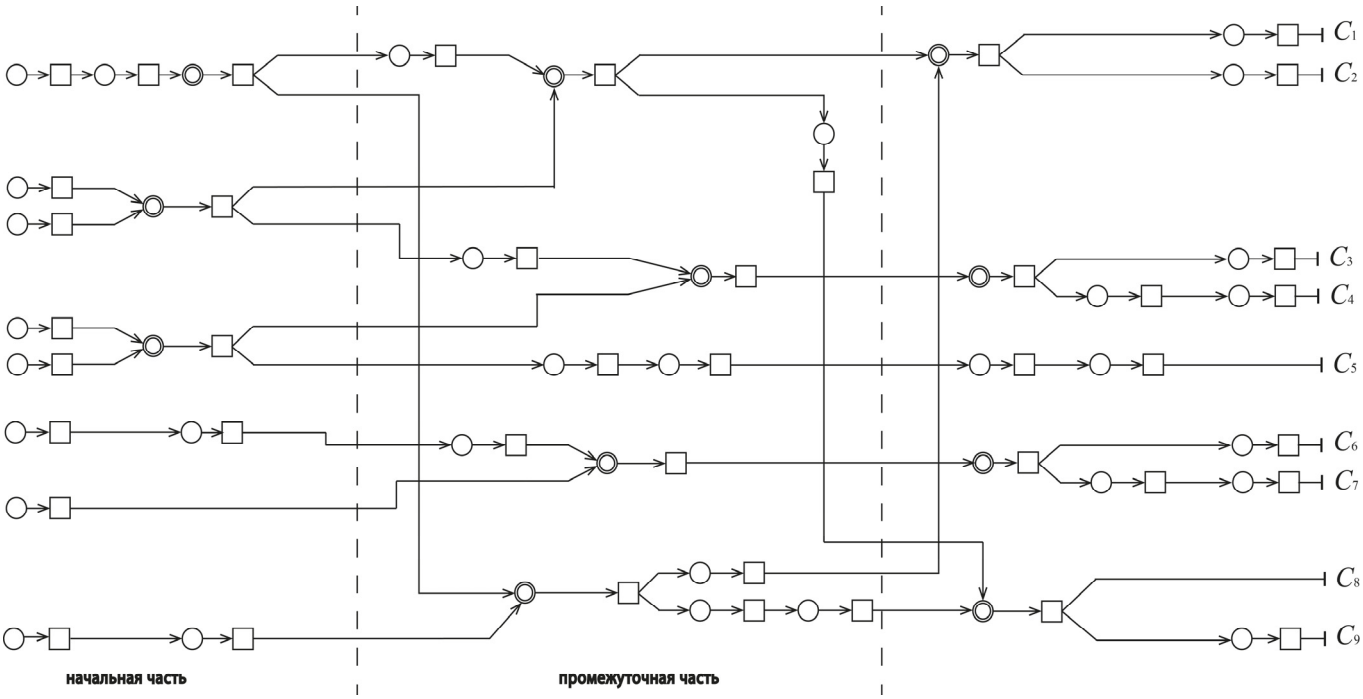


Рис. 4. Граф на критических путях изделий с общими вершинами

2) Критические пути в графе связаны общими вершинами, т.е. ограничение на выполнение изделий задано построенным ориентированным графом на критических путях изделий, в котором

веса всех агрегированных работ, кроме конечных, равны нулю, а веса конечных работ равны весам изделий ω_i, отражающих прибыль от реализации i-го изделия. В этом случае оптимальным решением задачи МВМ является приоритетно-упорядоченная последовательность выполнения агрегированных работ $\sigma^{опт}$, определяющая очередность назначения изделий на выполнение.

Задача решается при следующих ограничениях: простои прибора при выполнении агрегированных работ запрещены; прерывания агрегированных работ при выполнении запрещены.

Для решения задачи МВМ в данной постановке разработан эффективный эвристический алгоритм на основе точного ПДС-алгоритма [1], позволяющий решать задачи большой размерности. Для других базовых критериев эффективных алгоритмов решения задачи не существует. Покажем, что для получения очередности выполнения агрегированных работ в соответствии с другими базовыми критериями, можно эффективно использовать аппроксимирующие задачи МВМ.

Общее свойство всех исследуемых функционалов: для реализации эвристической стратегии достижения оптимального значения функционала в первую очередь назначать на выполнение агрегированные работы, принадлежащие изделию, имеющему высший приоритет.

Таким образом, необходимость аппроксимации задач планирования по остальным базовым критериям соответствующими задачами МВМ обоснована следующими причинами: 1) глубокая связь между этими задачами, проиллюстрированная выше; 2) существование эффективного приближенного алгоритма для решения задачи МВМ, базирующегося на идеях ПДС-алгоритма для задачи МВМ [1].

Для каждого из 31 критерия оптимальности строится точная или аппроксимирующая задача МВМ, в результате решения которой формируется приоритетно-упорядоченная последовательность агрегированных работ, определяющая очередность их выполнения.

Критерий 1 является точной задачей МВМ. Рассмотрим построение аппроксимирующих задач МВМ для базовых критериев 2–5.

Построение аппроксимирующей задачи МВМ для *критерия 2* осуществляется путем замены функционала $\max\sum_{i=1}^{n}\omega_i U_i$ на $\min\sum_{i=1}^{n}\omega_i C_i$, где C_i – момент окончания выполнения i-го изделия, ω_i – прибыль от выполнения i-го изделия, если оно выполнено точно в срок. Приведем обоснование возможности такого сведения [1].

1) Рассмотрим частный случай, когда задания являются независимыми. В соответствии с утверждением 9.1 [1], первым выполняется изделие, у которого отношение прибыли (в соответствии с критерием 2) к длительности является максимальным, т. е. максимальной является удельная прибыль на единицу времени производственного цикла. Таким образом, в задаче по критерию 2 логично первым реализовать выполнение изделия с максимальной удельной прибылью на единицу производственного цикла ω_1/l_1 (максимально эффективное использование времени производственного цикла), затем изделие с удельной прибылью ω_2/l_2 и т.д. Альтернативой этой эвристике является лишь полный перебор вариантов назначения изделий на выполнение.

2) Общий случай. Ограничение на выполнение заданий задано построенным ориентированным графом на критических путях. В этом случае оптимальным решением задачи МВМ является приоритетно-упорядоченное расписание, в котором последовательность изделий разбита на множества максимальных приоритетов $G_i, i=\overline{1,k}$, $p(G_1)\geq p(G_2)\geq\ldots\geq p(G_k)$. Таким образом, если первыми выполняются работы оптимального расписания, соответствующие множеству G_1, этим гарантируется, что на начальном отрезке производственного цикла длительностью $\sum_{i\in G_1} l_i$ выполняются изделия, для которых достигается максимальная усредненная прибыль на единицу времени производственного цикла:

$$p(G_1)=\sum_{i\in G_1}\omega_i \Big/ \sum_{i\in G_1} l_i .$$

Обоснование использования оптимального расписания задачи МВМ в общем случае для критерия 2 аналогично рассмотренному выше частному случаю с учетом замены термина «максимальная удельная прибыль на единицу времени производственного цикла» на термин «максимальная усредненная удельная прибыль на единицу времени производственного цикла».

Обоснование возможности построения аппроксимирующих задач МВМ для остальных критериев – аналогичное приведенному выше.

Для *критерия 3* функционал $\min\sum_{i=1}^{n}\omega_i\times\times\max(0, C_i-d_i)$ заменяется на $\min\sum_{i=1}^{n}\omega_i C_i$, где ω_i –

штраф за запаздывание окончания выполнения i-го изделия относительно директивного срока на единицу времени.

Для критерия 4 функционал $\max\sum_{i=1}^{n}\omega_i U_i$ заменяется на $\min\sum_{i=1}^{n}\omega_i C_i$, где ω_i – абсолютная прибыль от выполнения задания, не зависящая от момента окончания выполнения задания в том случае, если задание выполняется без запаздывания относительно директивного срока, иначе прибыль предприятия по этому заданию равна нулю. Обоснование этой процедуры аналогично обоснованию для критерия 2 [1].

Для *критерия 5* построение аппроксимирующей задачи МВМ осуществляется путем замены функционала $\min\sum_{i=1}^{n}\omega_i\left|C_i-d_i\right|$ на $\min\sum_{i=1}^{n}\omega_i C_i$, где ω_i – штраф за отклонение момента окончания выполнения i-го изделия от директивного срока на единицу времени.

Для каждой базовой составляющей синтетического критерия строится аппроксимирующая задача МВМ и создается граф на критических путях. В качестве примера рассмотрим построение аппроксимирующих задач МВМ для определения очередности запуска на выполнение агрегированных работ для критерия 7. Рассмотрим следующий синтетический критерий, базовыми для которого являются выполнение изделий точно в срок (критерий 2) и минимизация суммарного взвешенного запаздывания (критерий 3):

$$\max\left\{\sum_{i=1}^{k}\omega_i' U_i-\sum_{i=k+1}^{n}\omega_i''\max\left(0,C_i-d_i\right)\right\}. \qquad (2)$$

Следовательно, для синтетического критерия решаются две аппроксимирующие задачи МВМ при отношении порядка, заданном графом на критических путях:

$$\min\sum_{i=1}^{k}\omega_i' C_i \text{ и } \min\sum_{i=k+1}^{n}\omega_i'' C_i,$$

где ω_i' имеет тот же смысл, что и в критерии 2, а ω_i'' – тот же смысл, что и в критерии 4.

Блок 2. Для каждого из 31 критерия оптимальности решается соответствующее количество задач МВМ, в которых длительности и веса работ определяются длительностями и весами агрегированных работ в графе на критических путях (веса всех вершин, кроме конечных, равны нулю).

Далее граф на критических путях дополняется агрегированными работами, не лежащими на критических путях изделий, в соответствии с технологическими связями, то есть, приводится к виду сети на втором уровне (рис. 3). На этом графе реализуется построение согласованного плана выполнения агрегированных работ:

1) Если планирование осуществляется по базовому критерию, то:

а) для критерия 1, изделия распределяются с начала планового периода, и по концу выполнения изделий устанавливается их директивные сроки, которые согласовываются в блоке принятия решений;

б) при планировании по другим базовым критериям изделия распределяются с конца, начиная от директивных сроков. В результате определяются моменты запуска изделий на выполнение.

2) Если планирование осуществляется по синтетическому критерию, изделия делятся на группы, каждая из которых имеет общий для этой группы базовый критерий оптимальности. Порядок агрегированных работ в каждой группе определяется последовательностью $\sigma^{\text{опт}}$, полученной в результате решения задачи МВМ. Выполнение группы изделий осуществляется без прерываний до полного завершения группы.

Задача распределения групп изделий на выполнение сводится к определению очередности выполнения групп изделий и для получения точного решения этой задачи реализуется полным перебором. Распределение реализуется следующим образом:

а) если в состав синтетического критерия не входит критерий 1, то распределение всех групп осуществляется с конца, начиная от директивных сроков. По результатам распределения для каждого изделия определяются моменты запуска на выполнение.

б) если планирование осуществляется по синтетическому критерию, в состав которого входит критерий 1, то распределение группы изделий по критерию 1 осуществляется с начала планового периода. Остальные группы распределяются с конца по вышеописанным правилам.

Примечание. При планировании по базовому критерию 2–5 или синтетическому критерию может оказаться, что в результате распределения моменты начала выполнения начальных работ изделия (серии изделий) будут раньше начала планового периода. В таком случае, расписание по этому изделию и по связанным с ним общими вершинами другим изделиям сдвигается дальше по времени и срывается директивные сроки этих изделий. Поэтому в блоке принятия решений требуется выполнить корректировку плана по этим изделиям.

Алгоритмы согласованного планирования описаны в [1]. В качестве примера рассмотрим алгоритм согласованного планирования выполнения агрегированных работ, входящих в состав изделий 6 и 7. Пусть выполнение изделия 6 планируется по критерию 2 (точно в срок), а выполнение изделия 7 – по критерию 3. Синтетический критерий оптимальности определяется формулой (2). Граф на критических путях для этих изделий приведен на рис. 5. Предполагается, что приоритет изделия 6 значительно выше, чем приоритет изделия 7. Поэтому в структуре последовательности $\sigma^{опт}$ в первую очередь выполняются агрегированные работы, принадлежащие изделию 6 и их предшественники, принадлежащие другим изделиям – в данном случае, изделию 7.

Распределение агрегированных работ осуществляется на основании следующей эвристики. На первом шаге выбирается агрегированная работа с максимальным приоритетом из списка конечных работ и назначается таким образом, чтобы момент окончания ее выполнения соответствовал директивному сроку изделия. На каждом следующем шаге на ресурс назначается агрегированная работа с максимальным временем запуска в ресурсе и максимальным приоритетом. Максимальное время запуска агрегированной работы в ресурсе определяется наиболее поздним временем окончания выполнения агрегированной работы минус длительность ее выполнения. Пóзднее время окончания выполнения агрегированной работы определяется минимальным из времен запуска всех ее преемников по сети, определенных при назначении преемников на выполнение.

При таком распределении в первую очередь на выполнение назначаются изделия с наивысшим приоритетом максимально близко к директивным срокам без их нарушения.

Распределение изделий 6 и 7 по приборам приведено на рис. 6. Как видно из рисунка, изделие 6 не может быть выполнено в срок, т.к. момент его начала выполнения вышел за пределы планового периода. Поэтому в блоке принятия решений выполняется корректировка плана по этому изделию.

Примечание. Если изделий (серий изделий) небольшое количество, то можно исключить построение приоритетно-упорядоченной последовательности на модели третьего уровня, в этом случае на уровне согласованного планирования выбирается наилучший согласованный план, полученный полным перебором возможных последовательностей очередности выполнения изделий в приведенном алгоритме согласованного планирования.

Блок 3 – Блок принятия решений, описан выше.

Блок 4. После блока принятия решений по полученному портфелю заказов и директивным срокам необходимо построить допустимый точный пооперационный план выполнения изделий,

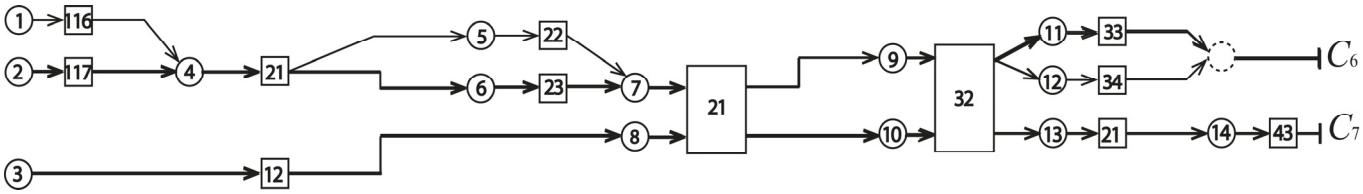


Рис. 5. Критические пути изделий 6 и 7 для согласованного планирования

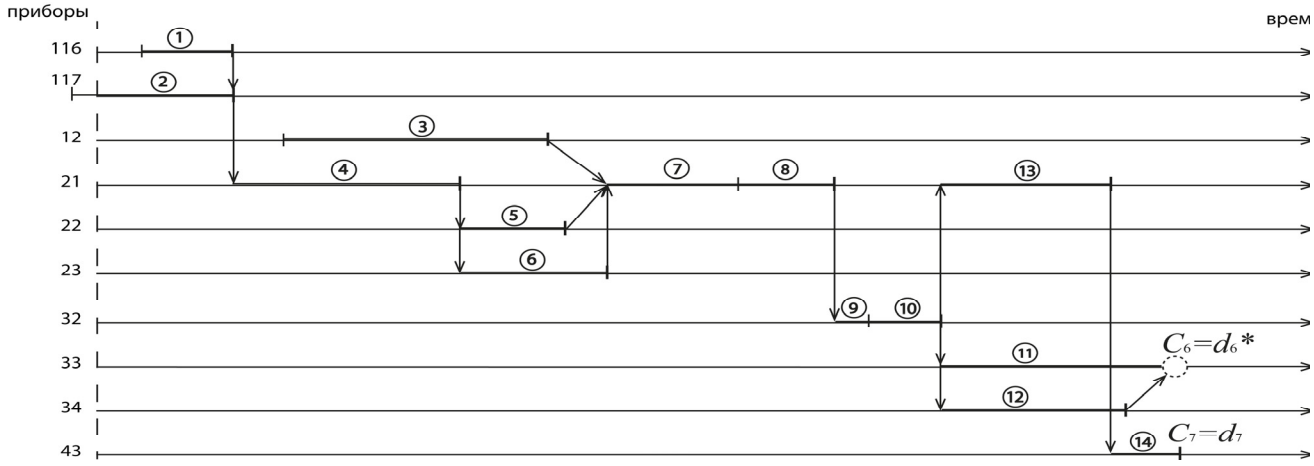


Рис. 6. Распределение выполнения агрегированных работ изделий 6 и 7

который является результатом решения сетевой многоэтапной задачи календарного планирования. Вид допустимого расписания и критерии оптимальности решения сетевой задачи планирования приведены ниже. Эти критерии оптимальности эффективно аппроксимируют любой из 31 критерия решения исходной задачи..

Определение. Допустимым является расписание, в котором для всех конечных работ, соответствующих готовым изделиям, непосредственно связанных с элементами типа 1, 2, 3, справедливо $C_i \le d_i$, $i \in I$, а для конечных работ, непосредственно связанных с элементами типа 4, 5, справедливо: $C_i \in d_i^*$, $d_i^* = [d_i - \Delta_i, d_i]$, $i \in I$, где $[d_i - \Delta_i, d_i]$ определяет диапазон получения допустимого решения.

Примечание. Δ_i строго больше нуля и выбирается из следующих соображений. Если исходный критерий планирования i-го изделия – выполнение работ точно в срок, то Δ_i практически всегда небольшое, но больше нуля. Если же изделие i оптимизируется не по критерию точно в срок, то длина интервала такая, что попадание момента окончания в этот интервал считается оптимальным с заданной инженерной точностью (например, точностью допустимых потерь на хранение). Δ_i может быть большим, если:

1) исходный критерий требует выполнения работ до директивного срока, тогда, если выполнение осуществляется раньше, издержки несущественны;

2) критерий – максимизация прибыли. Тогда потери на величину Δ_i несущественны, т.к. Δ_i существенно меньше длительности технологического цикла, однако величина Δ_i не должна менять оптимальную последовательность выполнения изделий, полученную в блоке 2.

Расписание является *оптимальным* по составному критерию, если оно отвечает следующим условиям:

1) расписание допустимое;

2) из всех допустимых расписаний оптимальным может быть только то, в котором достигается минимум суммарного опережения относительно конечных директивных сроков;

3) а) оптимальным является допустимое расписание, удовлетворяющее пунктам 1) и 2), в котором на множестве начальных работ достигается максимум:

$$\max\left(\min_j r_{1j}\right), j = \overline{1, p}$$

где r_{1j} – момент запуска работ на приборе $\boxed{1_j}$ (в сети, показанной на рис. 1, p = 18);

б) оптимальным является допустимое расписание, удовлетворяющее пунктам 1) и 2), в котором на множестве начальных работ достигается максимум:

$$\max\left(\min_j r_{1j}\right) = r_{1j_1}, j = \overline{1, p}$$

$$\max\left(\min_j r_{1j}\right) = r_{1j_l}, j = \overline{1, p}, l = \overline{2, p}, j \ne j_k, k = \overline{1, l-1}$$

(в сети, показанной на рис. 1, p = 18).

Примечание. Данный критерий оптимальности является хорошей аппроксимацией критериев, связанных с критерием «точно в срок» и не противоречит критериям, связанными с директивными сроками «не позже, чем» и критериям максимизации прибыли. Такая аппроксимация обосновывается эффективностью алгоритма решения сформулированной многоэтапной сетевой задачи, методология решения которой изложена в [3] для частного случая.

В зависимости от реальной технологии, интервалы и директивные сроки в процессе решения задачи будут пересчитываться для конечной части сети:

а) как интервалы на выходе из приборов типа 4, 5, предшествующих конечной работе;

б) как директивные сроки на выходе из приборов типа 2, 3, предшествующих конечной работе.

На рис. 7 показаны более подробно элементы конечной части сети (фрагмент рис. 2), тут $d_i^* = [d_i - \Delta_i, d_i]$, по директивным срокам d_i и интервалам d_i^* однозначно пересчитываются директивные сроки работ на выходе из приборов d', d_j' и интервалы $d_j'^*$, посредством сдвига на длительность последней работы.

Примечание. При построении допустимого расписания все конечные вершины и соответствующие им фрагменты сети исключаются из сетевой многоэтапной задачи, кроме вершин, определяемых самыми поздними работами, и соответствующих им фрагменты сети, т.к. после их исключения моменты выполнения соответствующих работ восстанавливаются однозначно.

Четвертый уровень реализуется тогда, когда построенный на третьем уровне точный план изготовления изделий частично не выполняется. В этом случае вводятся алгоритмы коррекции плана выполнения работ элементами, соответствующих исходному 31 критерию с новыми весовыми коэффициентами. Действительно, в блоке принятия решений исходные весовые коэффициенты были использованы для корректировки портфеля заказов и определения согласованных сроков выполнения изделий, а на четвертом уровне коэффициенты, связанные со

срывом сроков, могут быть существенно больше.

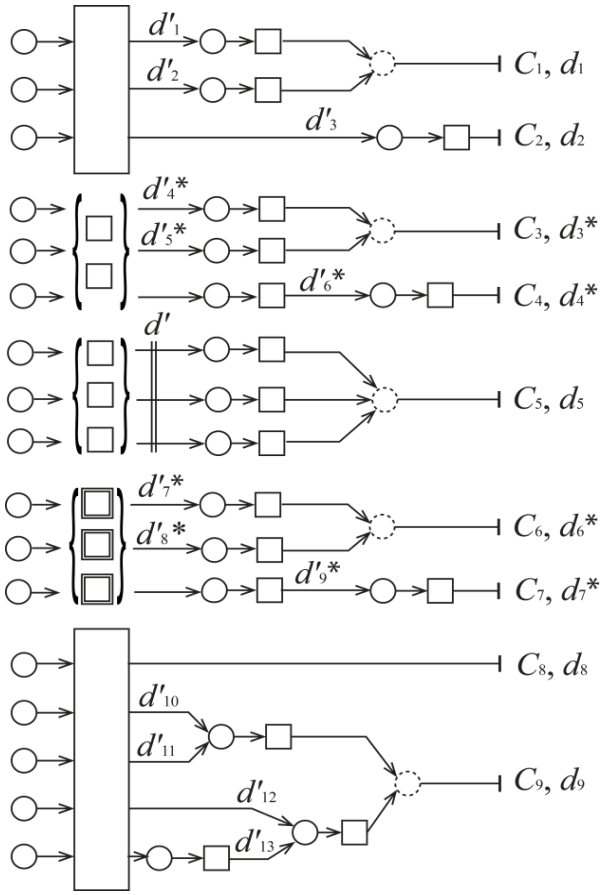


Рис. 7. Детализация конечной части

На четвертом уровне рассматривается следующая задача. Пусть для элементов типа 2–5 конечной части сети нарушается оптимальное расписание, полученное на третьем уровне: изменились моменты готовности начальных работ на входе приборов, а также их состав. Таким образом, корректировка плана сводится к нахождению оптимального расписания по оставшимся входным работам на обслуживание элементами типа 2–5 конечной части сети, минимизирующее либо суммарное опережение и запаздывание, либо суммарное взвешенное запаздывание работ относительно директивных сроков [1, 5]. С этой целью будут использованы эффективные точные ПДС-алгоритмы решения труднорешаемых задач [1].

По полученным моментам готовности всех входных работ элементов типа 2–5 конечной части сети, включая моменты готовности работ, приведших к необходимости корректировки, строится новое допустимое расписание, реализующее моменты готовности всех входных работ элементов типа 2–5 конечной части сети и являющееся оптимальным по п. 3 составного критерия оптимальности. С учетом полученного плана, реально существующих резервов рабочего времени, резервов мощности и количества приборов, необходимо гарантировать реализацию моментов готовности входных работ элементов типа 2–5 конечной части сети посредством итеративной процедуры в диалоговом режиме со специалистом по планированию.

Выводы

Приводится четырехуровневая модель планирования, принятия решений и оперативного управления в сетевых системах с ограниченными ресурсами. На основе оригинальной методологии создания формальной модели построения результирующего пооперационного плана [2] третий уровень четырехуровневой модели впервые представлен как многоэтапная сетевая задача календарного планирования. Приведены базовые элементы, на основе которых строится сетевая модель. Представлены критерии оптимизации и общая схема реализации алгоритмического обеспечения четырехуровневой модели. В состав алгоритмического обеспечения входят эффективные точные ПДС-алгоритмы для ряда труднорешаемых комбинаторных задач [1].

Список литературы

1. Згуровский М.З., Павлов А.А. Принятие решений в сетевых системах с ограниченными ресурсами: Монография.– К.: Наукова думка. – 2010. – 573 с.
2. Павлов А.А., Мисюра Е.Б., Халус Е.А., Сперкач М.О., Аракелян Г.А. Результирующая формализация первого уровня трехуровневой модели оперативного планирования и принятия решений по критерию минимизации суммарного опережения директивных сроков // Вісник НТУУ “КПІ”. Серія «Інформатика, управління та обчислювальна техніка». – К.: “ВЕК+”, 2012. – №56.– С. 56–57.
3. Павлов А.А., Сперкач М.О., Халус Е.А. Субоптимальный полиномиальный алгоритм решения одного класса многоэтапных сетевых задач календарного планирования // Вісник НТУУ “КПІ”. Серія «Інформатика, управління та обчислювальна техніка». – К.: “ВЕК+”, 2012. – №57.– С. 51–55.
4. Конвей Р.В., Максвелл У.Л., Миллер Л.У. Теория расписаний.– М.: Наука, 1975.– 359 с.
5. Павлов О.А., Місюра О.Б., Шевченко К.Ю. Побудова ПДС-алгоритму розв’язання задачі мінімізації сумарного зваженого запізнення виконання робіт на одному приладі // Вісник НТУУ “КПІ”. Серія «Інформатика, управління та обчислювальна техніка». – К.: “ВЕК+”, 2012. – №56.– С. 58–70.

УДК 519.854.2

ПАВЛОВ А.А.,
МИСЮРА Е.Б.,
ЛИСЕЦКИЙ Т.Н.

СОСТАВЛЕНИЕ РАСПИСАНИЯ ВЫПОЛНЕНИЯ НЕЗАВИСИМЫХ ЗАДАНИЙ ИДЕНТИЧНЫМИ ПАРАЛЛЕЛЬНЫМИ ПРИБОРАМИ, МОМЕНТЫ ЗАПУСКА КОТОРЫХ МЕНЬШЕ ОБЩЕГО ДИРЕКТИВНОГО СРОКА

Рассматривается задача планирования выполнения заданий параллельными приборами равной производительности для случая разных моментов запуска приборов T_i на выполнение работ при условии $T_i < d$. Критерий оптимизации: минимизация суммарного запаздывания относительно общего директивного срока. Предложен ПДС-алгоритм ее решения. Сформулированы признаки оптимальности полиномиальной составляющей алгоритма. Приведен пример решения задачи.

The problem of scheduling tasks on parallel machines of equal performance in the case of different starting times of machines T_i ($T_i < d$) is considered. The optimization criterion: minimizing total tardiness regarding a common due date. The PDC-algorithm to solve it is proposed. Optimality conditions of the polynomial component of the algorithm are formulated. An example of the problem solution is given.

Введение

ПДС-алгоритмы [1] – это алгоритмы с полиномиальной и экспоненциальной составляющей. В отличие от известных, эти алгоритмы относят решаемую индивидуальную задачу к подклассу полиномиально разрешимых в процессе анализа ее решения. В противном случае задача решается экспоненциальной составляющей.

ПДС-алгоритм решения исследуемой задачи включает полиномиальную составляющую и приближенный алгоритм и строится только на направленных перестановках. В результате решения задачи получаем либо строго оптимальное решение полиномиальной составляющей алгоритма, либо приближенное с верхней оценкой отклонения от оптимального.

Постановка задачи

Задано множество заданий $J = \{1, 2,..., n\}$, m приборов равной производительности, для каждого задания $j \in J$ известна длительность выполнения l_j. Все задания имеют общий директивный срок d. Моменты запуска приборов на выполнение работ T_i, $i = \overline{1,m}$, различны, и выполняется $T_i < d$, $i = \overline{1,m}$.

Необходимо построить расписание σ выполнения заданий $j \in J$ на m приборах такое, чтобы достигался минимум функционала:

$$F(\sigma) = \sum_{j \in J} \max[0; C_j(\sigma) - d],$$

где $C_j(\sigma)$ – момент завершения выполнения задания j в последовательности σ.

Сформулированная задача (назовем ее МСЗПР) относится к классу *NP*-трудных [2]. В [1] представлен ПДС-алгоритм решения сформулированной задачи при условии, что все задания множества J поступают одновременно, процесс обслуживания каждого задания можно начать в любой момент времени, и он будет протекать без прерываний до завершения обслуживания задания.

Рассмотрим свойства исследуемой задачи.

Пронумеруем задания множества $J = \{1, 2,..., n\}$ по неубыванию значений l_j, а приборы – по неубыванию значений T_i. На первом шаге выбираем задание j с минимальным l_j и назначаем на прибор i с минимальным T_i. Определяем время освобождения прибора i: $T_i^{осв} = T_i^{осв} + l_j$. Далее выбираем очередное задание для назначения на выполнение и назначаем на прибор с минимальным временем освобождения $T_i^{осв}$. Такую процедуру выполняем до тех пор, пока не будут распределены на выполнение все задания. Обозначим полученное расписание через $\sigma^{уп}$.

Для удобства изложения используем принятые в [1] обозначения:

$P_i(\sigma)$ – множество незапаздывающих заданий в расписании прибора i;

$S_i(\sigma)$ – множество запаздывающих заданий в расписании прибора i, для которых выполняется:

$$S_j^н < d,\ C_j > d,\ \forall j \in S_i(\sigma),$$

где $S_j^н$ — момент начала выполнения задания j;

$Q_i(\sigma)$ – множество запаздывающих заданий в расписании прибора i, для которых выполняются условия:

$$S_j^н < d,\ \forall j \in Q_i(\sigma),$$

$$P = \bigcup_{i=\overline{1,m}} P_i;\ S = \bigcup_{i=\overline{1,m}} S_i;\ Q = \bigcup_{i=\overline{1,m}} Q_i;$$

$R_i(\sigma)$ – резерв времени прибора i в расписании σ:

$$R_i(\sigma) = d - \sum_{j \in P_i(\sigma)} l_j;$$

$\Delta_i(\sigma)$ – запаздывание в выполнении задания $j \in S_i(\sigma)$ относительно директивного срока:

$$\Delta_i = \sum_{j \in P_i(\sigma) \bigcup S_i(\sigma)} l_j - d.$$

Для случая равных T_i в [1] сформулирована теорема, определяющая класс расписаний, который содержит оптимальное решение по рассматриваемому функционалу на множестве всех возможных расписаний, построенных для фиксированного множества заданий J. Используем для удобства изложения принятые нами обозначения.

Теорема 1 [3]. Существует оптимальное расписание, при котором выполняются условия:

1) $P \bigcup S = \{1, 2,..., |P \bigcup S|\}$;
2) если $P \bigcup S < n$, то $\sum_{j \in P_i \bigcup S_i} l_j \geq d$, и $Q_i \setminus S_i$ содержит те и только те элементы, которые отличаются от $|P \bigcup S| + i$ на величину, кратную m, $i = \overline{1,m}$.

Обозначим через Ψ_{PS} класс расписаний, удовлетворяющий условию теоремы 1.

Утверждение 1. Расписание $\sigma^{\text{уп}}$ принадлежит классу Ψ_{PS}.

Справедливость этого утверждения очевидна и следует непосредственно из построения последовательности $\sigma^{\text{уп}}$.

Исследуем общие свойства всех возможных расписаний σ, построенных на множестве заданий J посредством направленных перестановок в последовательности $\sigma^{\text{уп}}$ и принадлежащих классу Ψ_{PS}.

Обозначим через $\Psi_{PS}(\mu)$ класс расписаний, удовлетворяющий условию $|P(\sigma) \bigcup S(\sigma)| = \mu$, μ – натуральное число. Можно сказать, что $\Psi_{PS}(\mu) \subset \Psi_{PS}$. Оптимальное расписание σ^* принадлежит хотя бы одному из классов $\Psi_{PS}(\mu)$ при некотором $\mu = \mu^*$. Число различных непустых классов $\Psi_{PS}(\mu)$ не превышает m. [3]

Пусть $\overline{\mu}$ ($\underline{\mu}$) – наибольшее (наименьшее) значение μ, при котором $\Psi_{PS} \neq \varnothing$. Обозначим через $c(\sigma)$ число, равное остатку от деления $n - |P(\sigma) \bigcup S(\sigma)|$ на m при $n - |P(\sigma) \bigcup S(\sigma)| > 0$, и равное m в противном случае. В теореме 2 приведены достаточные условия, при которых расписанию $\sigma \in \Psi_{PS}(\mu)$ соответствует наименьшее суммарное запаздывание среди всех расписаний, принадлежащих классу $\Psi_{PS}(\mu)$, $\underline{\mu} \leq \mu \leq \overline{\mu}$ [3].

Теорема 2. Расписание $\sigma \in \Psi_{PS}(\mu)$ является оптимальным в $\Psi_{PS}(\mu)$, если при этом величина $z(\sigma) = \sum_{i=1}^{c(\sigma)} \Delta_i(\sigma)$ достигает наименьшего значения [3].

Обозначим через σ^* оптимальное расписание выполнения множества заданий J.

Следующая теорема позволяет оценить отклонение произвольного расписания от оптимального.

Теорема 3. В расписании $\sigma \in \Psi_{PS}$ выполняется $F(\sigma) - F(\sigma^*) \leq z(\sigma)$ [3].

Следствие. Если $z(\sigma) = 0$, то расписание σ оптимально.

Выделим из класса Ψ_{PS} класс расписаний $\Psi_P \in \Psi_{PS}$, удовлетворяющий дополнительно следующим условиям:

1) $P = \{1, 2,..., |P|\}$;
2) $\min_{j \in S(\sigma)} l_j > \max_{i=\overline{1,m}} R_i(\sigma)$;
3) $S_{j_k}^H \leq S_{j_l}^H$, если $l_{j_k} \leq l_{j_l}$, $\forall j_k, j_l \in S(\sigma)$.

Утверждение 2. Расписание $\sigma^{\text{уп}}$ принадлежит классу Ψ_P.

Доказательство очевидно, т.к. расписание $\sigma^{\text{уп}}$ удовлетворяет всем сформулированным выше условиям для класса Ψ_P.

Пусть $|P| < n$. Обозначим: $P_{\min}$ – минимальное количество заданий множества P, при котором $\Psi_P \neq \varnothing$; $P_{\max}$ – максимальное количество заданий множества P.

Утверждение 3. Для всех возможных расписаний $\sigma \in \Psi_P$, построенных на множестве заданий J в задаче МСЗПР, справедливо: $P_{\max} - P_{\min} < m$.

Доказательство. Для любого расписания $\sigma \in \Psi_P$ выполняется

$$\sum_{j=1}^{|P(\sigma)|} l_j = md - \sum_{i=1}^{m} T_i - \sum_{i=1}^{m} R_i(\sigma).$$

Следовательно, $|P(\sigma)| = P_{\min}$ ($P_{\max}$) справедливо для такого расписания σ, в котором $\sum_{i=1}^{m} R_i(\sigma^{\text{уп}})$ имеет максимально возможное (минимально возможное) значение среди всех расписаний класса Ψ_P. Но для каждого прибора i расписания $\sigma \in \Psi_P$ выполняется $R_i(\sigma) < \min_{j \in S(\sigma)} l_j$.

Следовательно,

$$\sum_{i=1}^{m} R_i(\sigma) < m \cdot \min_{j \in S(\sigma)} l_j \leq \sum_{i=1}^{m} l_{|P(\sigma)|+i}$$

и $P_{\max} - P_{\min} < m$.

Справедливы следующие утверждения, доказательство которых очевидно.

Утверждение 4. При построении оптимального расписания в результате направленных перестановок возможны перемещения заданий только между множествами $P(\sigma)$ и $S(\sigma)$.

Доказательство основано на утверждении 3 и теореме 1.

Утверждение 5. При перестановке задания $j \in P(\sigma)$ с прибора i_k с бóльшим числом запаздывающих заданий на прибор i_l с меньшим числом запаздывающих заданий уменьшается Δ_{i_k} на величину, равную l_j.

Доказательство очевидно и вытекает из определений $\Delta_i(\sigma)$ и $R_i(\sigma)$.

Утверждение 6. Максимальная разность количества запаздывающих заданий на приборах в расписаниях $\sigma \in \Psi_P$ не превышает единицы.

Справедливость этого утверждения очевидна и следует непосредственно из определения класса Ψ_P.

Определение 1. Расписание с одинаковым числом запаздывающих заданий на приборах назовем равномерным.

Теорема 4. Равномерное расписание $\sigma \in \Psi_P$ является оптимальным [1].

Введем обозначения:

$L_{\max}$ – максимальное число запаздывающих заданий на приборах;

$L_{\min}$ – минимальное число запаздывающих заданий на приборах;

$i = \overline{1,k}$ – приборы с числом запаздывающих заданий $L_{\max}$;

$|P(\sigma)| = P$ – мощность множества $P(\sigma)$;

$\Delta_\Sigma(\sigma) = \sum_{i=1}^{k} \Delta_i$;

$R_\Sigma(\sigma) = \sum_{i=k+1}^{m} R_i$; $\Omega_\Sigma(\sigma) = \min\{R_\Sigma(\sigma), \Delta_\Sigma(\sigma)\}$.

Теорема 5 [1]. Если в расписаниях $\sigma \in \Psi_P$, $\sigma' \in \Psi_P$, построенных на заданном множестве заданий J в задаче МСЗПР, максимальное число запаздывающих заданий одинаково, то при $R_\Sigma(\sigma) \neq 0$ и $R_\Sigma(\sigma') \neq 0$ справедливо: $R_\Sigma(\sigma) - R_\Sigma(\sigma') = \Delta_\Sigma(\sigma) - \Delta_\Sigma(\sigma')$.

Доказательство. Пусть $md - \sum_{i=1}^{m} T_i = R^0$ и выполняется условие $R_1 \geq R_2 \geq R_3 \geq \ldots \geq R_m$:

$$R^0 = \sum_{j \in P(\sigma)} l_j + \sum_{j \in S(\sigma)} l_j - \sum_{i=1}^{m} \Delta_i = \sum_{j \in P(\sigma)} l_j - \sum_{i=1}^{k} \Delta_i(\sigma) + \sum_{j=p+1}^{p+k} l_j + \left(\sum_{j=p+k+1}^{p+m} l_j - \sum_{i=k+1}^{m} \Delta_i \right) =$$

$$= \sum_{j \in P(\sigma)} l_j + \left(\sum_{j=p+1}^{p+k} l_j - \Delta_\Sigma(\sigma) \right) + R_\Sigma(\sigma) = const. \quad (1)$$

Рассмотрим следующие случаи:

а) $P(\sigma) = P(\sigma')$, справедливость теоремы очевидна в соответствии с (1);

б) $P(\sigma) < P(\sigma')$, в этом случае задания $j \in S(\sigma)$, $j = \overline{p+1, p+k'}$, $k' < k$, стали незапаздывающими, т. е. вошли в множество $P(\sigma')$:

$$R^0 = \left(\sum_{j \in P(\sigma)} l_j + \sum_{j=p+1}^{p+k'} l_j \right) + \left(\sum_{j=p+k'+1}^{p+k} l_j - \Delta_\Sigma(\sigma') \right) + R_\Sigma(\sigma'),$$

теорема справедлива согласно (1);

в) $P(\sigma) > P(\sigma'')$, $\Delta_\Sigma(\sigma) < \Delta_\Sigma(\sigma'')$; в этом случае число запаздывающих заданий на приборах $i = \overline{k+1, k+k''}$, $k'' < m$ увеличилось, т.к. часть заданий из множества $P(\sigma)$ стали запаздывающими:

$$R^0(\sigma) = \sum_{j \in P(\sigma^{\text{уп}})} l_j - \sum_{j=p-k''+1}^{p} l_j + + \left(\sum_{j=p-k''+1}^{p} l_j + \sum_{j=p+1}^{p+k} l_j - \Delta_\Sigma(\sigma') \right) + R_\Sigma(\sigma').$$

Теорема справедлива в соответствии с (1). Теорема доказана.

Для задачи МСЗПР также справедливы следующие теоремы и утверждения, доказательство которых приведено в [1]:

Теорема 6. Для любых двух расписаний $\sigma \in \Psi_P$ и $\sigma' \in \Psi_P$ справедливо следующее:

$$F(\sigma) - F(\sigma') = \Omega_\Sigma(\sigma) - \Omega_\Sigma(\sigma'). \quad (2)$$

Теорема 7. Если в расписании $\sigma \in \Psi_P$ выполняется $\Omega_\Sigma(\sigma) = \min\{R_\Sigma(\sigma), \Delta_\Sigma(\sigma)\} = 0$, то расписание σ оптимально.

Величина $\Omega_\Sigma(\sigma) = \min\{R_\Sigma(\sigma), \Delta_\Sigma(\sigma)\}$ является основной характеристикой расписания $\sigma \in \Psi_P$, где $\Delta_\Sigma(\sigma)$ показывает, насколько можно теоретически уменьшить значение функционала $F(\sigma)$, чтобы получить оптимальное расписание. Суммарный резерв $R_\Sigma(\sigma)$ показывает, какие резервы существуют для получения оптимального расписания.

Утверждение 7. Пусть $\sigma \in \Psi_P$, тогда если $z(\sigma) = 0$, то $\Omega_\Sigma(\sigma) = 0$.

Теорема 8. Для расписания $\sigma \in \Psi_P$ справедливо следующее соотношение:

$$F(\sigma) - F(\sigma^*) \leq \Omega_\Sigma(\sigma).$$

Таким образом, для расписаний $\sigma \in \Psi_P$ величина $\Omega_\Sigma(\sigma)$ является оценкой отклонения значения суммарного запаздывания оптимального расписания.

Утверждение 8. Для расписаний $\sigma \in \Psi_P$ выполняется $\Omega_\Sigma(\sigma) \leq z(\sigma)$.

Как было показано выше, при выполнении $z(\sigma) = 0$ выполняется равенство $\Omega_\Sigma(\sigma) = 0$, а при $z(\sigma) > 0$ справедливо $z(\sigma) = \Delta_\Sigma(\sigma)$. Следовательно, для любого расписания $\sigma \in \Psi_P$ выполняется $\Omega_\Sigma(\sigma) \le z(\sigma)$.

Рассмотрим новый класс $\Psi(\sigma_p) \in \Psi_{PS}$, который состоит из произвольных расписаний σ, полученных в результате направленных последовательных перестановок, выполненных в произвольном порядке и уменьшающих $\Delta_\Sigma(\sigma)$, и, следовательно, $R_\Sigma(\sigma)$ [1]. Эти перестановки последовательно выполняются с некоторого расписания $\sigma \in \Psi_P$ и осуществляются посредством переноса незапаздывающих заданий между приборами I_Δ и I_R в текущем расписании σ_k, где $I_R(\sigma)$ – множество номеров приборов расписания σ, на которых запаздывает меньшее число заданий; $I_\Delta(\sigma)$ – множество номеров приборов расписания σ, на которых запаздывает большее число заданий.

При этом порядок выполнения работ на приборах, кроме указанных выше, не изменяется. В результате проведенной перестановки в полученном расписании σ_{k+1} может измениться количество запаздывающих заданий только на одном из приборов из множества $I_\Delta(\sigma_k)$ с номером i_1 и на одном приборе из множества $I_R(\sigma_k)$ с номером i_2. Перестановка является запрещенной, если в расписании σ_{k+1} количество заданий на приборе i_2 больше количества заданий на приборе i_1.

Обозначим класс таких расписаний как $\Psi(\sigma_P)$.

Теорема 9. Для любого расписания $\sigma \in \Psi(\sigma_P)$ справедлива оценка отклонения показателя качества от оптимального значения:

$$F(\sigma) - F(\sigma^*) \le \Omega_\Sigma(\sigma).$$

Следствие. Для расписаний $\sigma \in \Psi(\sigma_P)$ справедливы теоремы 7, 8 и утверждение 8.

Таким образом, в расписаниях $\sigma \in \Psi(\sigma_P)$ изменение значения функционала так же, как и в расписаниях $\sigma \in \Psi_P$, определяется величинами $\Delta_\Sigma(\sigma)$, $R_\Sigma(\sigma)$.

Теорема 10. Если в расписании $\sigma \in \Psi(\sigma_P)$ $\Omega_\Sigma(\sigma) = \min\{R_\Sigma(\sigma), \Delta_\Sigma(\sigma)\}$ достигает наименьшего значения, то расписание σ оптимально.

В следующих утверждениях, доказательство которых очевидно, сформулированы свойства, характеризующие задания $j \in S \cup Q$ в расписаниях $\sigma \in \Psi(\sigma_P)$.

Утверждение 9. Пусть $S'_k(\sigma) \cup Q'_k(\sigma)$, $S''_l(\sigma) \cup Q''_l(\sigma)$ – множества запаздывающих заданий на приборах k и l, соответственно. В расписаниях, полученных в результате перестановки множеств запаздывающих заданий между приборами, то есть при выполнении $S''_k(\sigma) \cup Q''_k(\sigma)$, $S'_l(\sigma) \cup Q'_l(\sigma)$, отклонение показателя качества от оптимального расписания определяется функцией $\Omega_\Sigma(\sigma) = \min\{R_\Sigma(\sigma), \Delta_\Sigma(\sigma)\}$.

Пусть на приборах k, l, r в расписании σ задания $j \in S \cup Q$ пронумерованы следующим образом: $j_k, j_{k+m}, j_{k+2m}, \ldots, j_l, j_{l+m}, j_{l+2m}, \ldots, j_r, j_{r+m}, j_{r+2m}, \ldots$

Определение 2. Задания j_k, j_l, j_r, или $j_{k+m}, j_{l+m}, j_{r+m}, \ldots$ или $j_{k+2m}, j_{l+2m}, j_{r+2m}, \ldots$ назовем запаздывающими заданиями одного уровня.

Утверждение 10. На приборах с одинаковым числом запаздывающих заданий задания одного уровня можно менять местами. При таких перестановках значение функционала (показателя качества) не изменяется.

Утверждение 11. От расписания $\sigma \in \Psi(\sigma_P)$ всегда можно перейти к расписанию $\sigma \in \Psi_{PS}$ с тем же значением показателя качества посредством перестановки запаздывающих заданий одного уровня.

Описание ПДС-алгоритма

На основании приведенных теоретических результатов разработан эффективный ПДС-алгоритм решения поставленной задачи. Алгоритм состоит из двух этапов: Этап 1 (Алгоритм А0) – построение начального расписания $\sigma^{уп}$. Этап 2 (Алгоритм А) – построение оптимального расписания σ^*.

Описание Алгоритма А0.

1. Пронумеруем задания множества $J = \{1, 2, \ldots, n\}$ по неубыванию значений l_j.

2. Пронумеруем приборы по неубыванию значений T_i.

3. Полагаем $T_i^{осв} = T_i \ \forall i = \overline{1, m}$.

4. Выбираем задание j с минимальным l_j из неназначенных заданий и назначаем на прибор i с минимальным временем освобождения $T_i^{осв}$.

5. Определяем новое время освобождения прибора i: $T_i^{осв} = T_i^{осв} + l_j$.

6. Если распределены на выполнение все задания, конец алгоритма. Иначе переход на п. 4.

Обозначим полученное расписание через $\sigma^{уп}$.

В Алгоритме А используются следующие типы перестановок.

Перестановка $1P\text{-}0P\text{-}\Delta$. С прибора $h \in I_\Delta(\sigma)$ на прибор $r \in I_R(\sigma)$ перемещается задание j такое, что $l_j \ge \Delta h(\sigma)$, $l_j \le R_r(\sigma)$.

Перестановка $1P\text{-}0P\text{-}R\Delta$. С прибора $h \in I_\Delta(\sigma)$ на прибор $r \in I_R(\sigma)$ перемещается задание j такое, что $l_j < \Delta h(\sigma)$, $l_j > R_r(\sigma)$.

Перестановка 1*P*-0*P*-*R*. С прибора $h \in I_\Delta(\sigma)$ на прибор $r \in I_R(\sigma)$ перемещается задание j такое, что $l_j \le \Delta h(\sigma)$, $l_j \le R_r(\sigma)$.

Описание Алгоритма А.

1. Построение начального расписания $\sigma^{\text{уп}}$ по Алгоритму А0.

2. Определяем $\Omega_\Sigma(\sigma^{\text{уп}})$. Если $\Omega_\Sigma(\sigma^{\text{уп}}) = 0$, то расписание $\sigma^{\text{уп}}$ оптимально, конец алгоритма. В противном случае переходим к пункту 3.

3. Если $R_\Sigma(\sigma) \ge \Delta_\Sigma(\sigma)$, выполняем пункт 4. Иначе полагаем $\sigma = \sigma^{\text{уп}}$, переходим к п. 9.

4. Полагаем $\sigma = \sigma^{\text{уп}}$, $h = 1$, $f = f(\sigma)$, где f – количество приборов с бóльшим числом запаздывающих заданий.

5. Если для прибора h возможна перестановка 1*P*-0*P*-Δ, то выполняем ее. Получаем расписание σ′. Переходим к п. 6, в случае отсутствия перестановки переходим к п. 7.

6. Полагаем $h = h + 1$. Если $h \le f$, переходим к п. 5, иначе к п. 8.

7. Если для прибора h возможна перестановка 1*P*-0*P*-*R*, то выполняем ее. Переходим к п. 6.

8. Если $\Omega_\Sigma(\sigma) = 0$, то реализовалась полиномиальная составляющая алгоритма, расписание σ оптимально. Определяем значение оптимизируемого функционала. Иначе $\Omega_\Sigma(\sigma)$ – оценка отклонения функционала от оптимального, конец алгоритма.

9. Полагаем $r = f(\sigma) + 1$.

10. Если для прибора r возможна перестановка 1*P*-0*P*-*R*Δ, выполняем ее. Получаем расписание σ′, переходим к п. 11. В противном случае переходим к п. 12.

11. Полагаем $r = r + 1$. Если $r \le m$, идем на п. 10, иначе на п. 13.

12. Если для прибора r возможна перестановка 1*P*-0*P*-*R*, выполняем ее, получаем расписание σ′. Переходим к п. 11.

13. Если $\Omega_\Sigma(\sigma) = 0$, то реализовалась полиномиальная составляющая алгоритма, расписание σ оптимально. Определяем значение оптимизируемого функционала. Иначе $\Omega_\Sigma(\sigma)$ – оценка отклонения функционала от оптимального, конец алгоритма.

Трудоемкость алгоритма А определяется полиномом $O(mn \log n)$.

Пример.

Задано: $m = 4$, $n = 17$, $d = 10$, значения T_i и l_j:

j	1	2	3	4	5	6	7	8	9
l_j	1	1	1	2	2	2	3	3	3
j	10	11	12	13	14	15	16	17	
l_j	4	4	4	5	5	5	6	6	

i	1	2	3	4
T_i	0	1	2	3

Начальное расписание $\sigma^{\text{уп}}$:

i	j	l_j	d	c_j	F	i	j	l_j	d	c_j	F
1	1	1	10	1		2	17	6	10	19	9
1	5	2	10	3		3	3	1	10	3	
1	9	3	10	6		3	7	3	10	6	
1	11	4	10	10		3	12	4	10	10	
1	15	5	10	15	5	3	16	6	10	16	6
2	2	1	10	2		4	4	2	10	5	
2	6	2	10	4		4	8	3	10	8	
2	10	4	10	8		4	14	5	10	13	3
2	13	5	10	13	3						

$F_\Sigma(\sigma^{\text{уп}}) = 26$, $\Delta_\Sigma(\sigma^{\text{уп}}) = 3$, $R_\Sigma(\sigma^{\text{уп}}) = 2$.

Задание 6 с прибора 2 переносим на прибор 4. Получаем расписание σ^*:

i	j	l_j	d	c_j	F	i	j	l_j	d	c_j	F
1	1	1	10	1		3	3	1	10	3	
1	5	2	10	3		3	7	3	10	6	
1	9	3	10	6		3	12	4	10	10	
1	11	4	10	10		3	16	6	10	16	6
1	15	5	10	15	5	4	4	2	10	5	
2	1	1	10	2		4	6	2	10	7	
2	10	4	10	6		4	8	3	10	10	
2	13	5	10	11	1	4	14	5	10	15	5
2	17	6	10	17	7						

$F_\Sigma(\sigma^*) = 24$. Полученное расписание оптимально, т.к. $R_\Sigma(\sigma^*) = 0$.

Выводы

Исследованы и теоретически обоснованы свойства сформулированной задачи. Получены признаки оптимальности полиномиальной составляющей ПДС-алгоритма, оценка отклонения от оптимального для экспоненциальной составляющей, разработан эффективный ПДС-алгоритм решения задачи с трудоемкостью $O(mn \log n)$. В этой задаче задается полиномиальное ограничение на число вычислений от размерности задачи, и характеристика $\Omega_\Sigma(\sigma) = \min\{R_\Sigma(\sigma), \Delta_\Sigma(\sigma)\}$ является оценкой отклонения от оптимального значения функционала. Приведен пример.

Список литературы

1. Згуровский М.З., Павлов А.А. Принятие решений в сетевых системах с ограниченными ресурсами: Монография.– К.: Наукова думка. – 2010. – 573 с.
2. Гери М.Р., Джонсон Д.С. Вычислительные машины и труднорешаемые задачи. – М.: Мир, 1982.–416с.
3. Танаев В.С., Шкурба В.В. Введение в теорию расписаний. – М.: Наука, 1975.– 256 с.

УДК 004.[832+75]

ТЕЛЕНИК С.Ф.,
АМОНС О.А.,
ЄФРЕМОВ К.В.,
ЖУК С.В.

СЕМАНТИЧНА ІНТЕГРАЦІЯ РІЗНОРІДНИХ ІНФОРМАЦІЙНИХ РЕСУРСІВ

В статье рассмотрена проблема интеграции информационных ресурсов Мировой системы данных. Проанализированы особенности информационного обмена данными центров Мировой системы данных в контексте поддержки междисциплинарных исследований. Выполнен аналитический обзор систем и технологий интеграции разнородных информационных ресурсов, разработана их систематизация по комплексу признаков. Предложен общий подход к решению проблемы интеграции на основе web-ориентированных технологий, способа организации взаимодействия источников даннях с посредником и моделей дескриптивной логики в терминах OWL, RDF и SPARQL. Рассмотрены модели, используемые для реализации подхода.

The article considers the problem of information resources integration of World Data System. There were analyzed the features of the information exchange performed by Data Centers of World Data System in the context of interdisciplinary research support. An analytical review of systems and technologies of heterogeneous information resources integration was carried out, their classification by the traits complex was developed. There was proposed a general approach to the solving of the integration problem on the basis of web-technologies, on the basis of the method of interaction organization between data sources with the mediator, and descriptive logic models in terms of OWL, RDF and SPARQL. The models used for the approach implementation have been considered.

Вступ

Існуючі центри даних Світової системи даних надають користувачам доступ до накопичених за великий період неоднорідних даних і різноманітні засоби та інструменти їх оброблення і аналізу [1]. Однак користувачі змушені переборювати певні труднощі при спробах отримати потрібні дані, якщо вони розподілені між декількома джерелами і потребують опрацювання із застосуванням декількох засобів та інструментів їх оброблення і аналізу. Ця комплексна проблема досить умовно поділяється на інтеграцію даних і застосувань, оскільки зазначені складові в реальних інформаційних системах тісно пов'язані. Авторами в праці [2] запропонований підхід до інтеграції застосувань. Однак його реалізація вимагає вирішення проблеми інтеграції даних у випадках, коли застосування мають опрацьовувати дані із різних і часто гетерогенних джерел. Одним із ефективних рішень цієї проблеми може стати створення семантизованого каталогу даних, який надає доступ до віддалених джерел даних за принципом «єдиного вікна». Для реалізації цієї концепції і створення інформаційних технологій реальної інтеграції різнорідних інформаційних ресурсів необхідно вирішити низку проблем, які й складають предмет виконаних у статті досліджень.

Сутність і проблематика інтеграції джерел даних

За більш ніж 50 років системою Світових центрів даних (СЦД, World Data Centers – WDC) і Федерацією астрономічних і геофізичних служб аналізу даних (ФАГС, Federation of Astronomical and Geophysical data analysis Services – FAGS) були накопичені величезні масиви даних і інформації, набутий великий досвід з організації міжнародного обміну даними, створені і впроваджені інформаційні технології (ІТ) довгострокового зберігання, аналізу та оброблення даних.

Однак стрімкий розвиток ІТ на переломі тисячоліть змінив традиційні уявлення про інформаційні процеси. Хмарні обчислення і широке впровадження сервісного підходу сприяли формуванню ІТ-середовища, якому притаманні консолідація програмно-апаратних засобів, конвергентність сервісів, віртуалізація ресурсів. Понад 50 функціонуючих на сьогодні СЦД не повною мірою інтегрувалися в нове середовище. Їх потужні системи накопичення і оброблення інформації виявилися не цілком адекватними потребам науки, недостатньо гнучкими для їх використання в міждисциплінарних дослідженнях.

Тому перед створеною у 2008 р. новою між-

дисциплінарною структурою – Світовою системою даних (ССД, World Data System - WDS), яка інтегрує осередки СЦД і ФАГС, поставлене завдання впровадити новий, координований глобальний підхід до організації збереження, оброблення і обміну науковими даними та інформацією, який гарантує універсальний рівноправний доступ до якісних даних та інформації для досліджень, освіти та прийняття рішень. Науковому комітету, який координує діяльність ССД необхідно вирішити низку проблем, пов'язаних з уніфікацією форматів і протоколів передачі даних, організацією контролю якості наукових даних та інформації та ін. [1].

По суті мова йде про більш масштабні проблеми інтеграції джерел даних, раціональної організації збереження, оброблення і поширення даних, формування ІТ-спільноти, здатної виконувати складні міждисциплінарні дослідження. Стисло розглянемо зазначені проблеми, перш ніж зосередитися на проблемі інтеграції джерел.

Світові центри даних, як правило, мають дисциплінарну спрямованість. Їх інформаційно-довідкова діяльність (а це один із основних напрямів діяльності СЦД), пов'язана з підтримкою архівів і баз даних, створенням та актуалізацією інвентаризаційних каталогів, формуванням метаданих, наданням необхідних консультацій кінцевим користувачам, можлива лише за наявності в СЦД фахівців у відповідних наукових галузях. Так, СЦД з геоінформатики та сталого розвитку (СЦД-Україна) в Україні покликаний забезпечити доступ українському науковому співтовариству до глобальних інформаційних ресурсів Міжнародної ради з науки в галузі наук про Землю, планетарної та космічної фізики та відповідних суміжних дисциплін, а також забезпечувати збір і зберігання національних наукових даних з зазначених дисциплін та їх репрезентацію світовому співтовариству. На СЦД-Україна також покладено збір, оброблення та аналіз світових даних, необхідних для досліджень у сфері сталого розвитку.

Впровадження сервісного підходу в такій великій розподіленій системі має особливості і становить складну проблем. Етапи еволюції програмних сервісів надання даних у СЦД [3]:

1. Дані надавалися користувачу просто як великі файли, причому усі завдання з пошуку та оброблення даних у ньому користувач повинен був виконувати самостійно.

2. Збережені СЦД дані було перенесено в бази даних, а користувачеві було надано засоби для введення запиту на них. Такий підхід вимагав від користувача знання мови запитів відповідної СУБД та інформації про схему БД, до яких необхідно виконати запит.

3. Клієнт-серверні технології локальних мереж обумовили в СЦД появу «тонких» та «товстих» застосувань-клієнтів, які надавали користувачам форми для вибору даних. Вони інкапсулювали в собі засоби генерування запитів до джерела даних, забезпечували користувачеві дружній інтерфейс, але розроблення таких форм часто вимагало істотних трудовитрат, і в більшості випадків – індивідуального підходу до кожного джерела (або файлу) даних.

4. Форми вибору даних еволюціонували у фасетні форми пошуку даних, у яких форми пошуку даних автоматично генеруються для кожного джерела даних і можна використовувати додаткові описи для збереження метаданих кожного джерела.

5. Створення системи сервісів надання даних за принципом «єдиного вікна» для доступу до даних. Така організація передбачає інтеграцію джерел даних в єдину систему з урахуванням їх семантики. Важливі на етапах 1 – 4 каталоги посилань до засобів надання даних та їх метаописи, необхідні для структурування доступних джерел даних, з яких розпочинав взаємодію із сервісом надання даних користувач, перетворюються у невід'ємну частину власне сервісу надання даних. Втілення цієї ідеї породжує наукову проблему інтеграції інформаційних ресурсів різнорідних джерел.

Розглянемо проблему інтеграції та каталогізації інформаційних ресурсів більш прискіпливо, щоб виділити її задачі і особливості, які необхідно врахувати при їх розв'язанні. По-перше, вимагають вирішення задачі інтеграції даних і застосувань, які вже давно стоять перед багатьма організаціями, проте до останнього часу технологічні можливості в цій сфері були досить обмеженими. По-друге, оброблення запитів до розподілених джерел даних породжує фундаментальні задачі побудови формалізмів для метаописання джерел і формування запитів, їх розбору (парсингу), формування і оптимізації плану виконання запитів [4]. Вони виникають у будь-яких системах зберігання даних – розподілених, централізованих чи паралельних – оскільки оброблювач запитів має отримати запит, зрозуміти значення його параметрів, сформувати план виконання запиту, оптимізувати і вико-

нати його, щоб отримати потрібні користувачам і застосуванням дані.

Формальні засоби метаописання джерел і формування запитів звичайно визначаються на основі логічних, алгебраїчних і графічних формалізмів. Метаопис має зберігати необхідну для формулювання, парсингу, перетворення і оптимізації запиту інформацію. Для цього він має містити схему джерела (джерел) даних (наприклад, визначення таблиць, уявлень, типи даних і т.д.), схеми поділу (наприклад, інформацію про те, що глобальні таблиці були розділені і як вони можуть бути відновлені), фізичну інформацію (розташування розділів таблиць, структуру індексів і статистику для оцінювання ефективності виконання плану).

Поданий у зручному для користувача вигляді запит має бути обробленим і переведеним у внутрішнє представлення, наприклад, граф запиту, зручне для парсингу.

Формування плану виконання запиту, насамперед, передбачає його еквівалентне перетворення, спрямоване на ліквідацію надлишкових предикатів, спрощення виразів і усунення вкладених запитів. На виході отримуємо план, який визначає послідовність виконання підзапитів до джерел даних та стратегію їх об’єднання. А для розподілених систем визначається, які частини запиту будуть виконуватися на яких джерелах даних.

Оптимізація плану обумовлена необхідністю врахувати при його виконанні фізичний стан системи – розмір таблиць, індекси, швидкість процесора тощо. Вона передбачає уточнення індексів і методів (наприклад, хешування і сортування), які будуть використовуватися для виконання запиту і його операцій (наприклад, об'єднання і групування), і в якому порядку вони будуть застосовані. Також уточнюються вузли, на яких виконуються операції, остаточно формується послідовність виконання підзапитів до джерел даних і стратегія їх об’єднання. Звичайно плани подаються у вигляді дерев, вершинами яких є оператори (з'єднання, групування, сортування та ін.), а ребра вказують послідовність виконання операторів.

Оптимізація запитів у гетерогенній системі має бути універсальною і базуватися на оцінках «вартості» виконання запитів до джерел даних. Застосовуються такі стратегії оптимізації [4]:

1. Визначається узагальнена модель оцінювання вартості виконання запиту для всіх компонентів і її коефіцієнти корегуються для кожного компоненту за результатами виконання набору тестових запитів.

2. Визначається своя модель оцінювання вартості для кожного компоненту.

3. Вартість плану виконання запитів визначається на основі опрацювання статистичних даних моніторингу виконання компонентами запитів.

Найвищу точність оцінок вартості виконання запитів забезпечує другий підхід, реалізація якого найвитратніша.

Після оптимізації виникає задача трансформації плану виконання запиту у виконуваний план, яка може включати генерацію асемблероподібного коду для ефективного виконання окремих частин плану, наприклад предикатів і виразів, на джерелі даних.

Задача власне виконання запиту полягає у використанні загальних реалізацій для кожного оператора. Організація роботи сучасних підсистем виконання запитів базується на моделі ітератора, яка забезпечує реалізацію операторів і стандартний інтерфейс.

Загалом, виходячи із необхідності збереження дисциплінарності СЦД, для інтеграції їх різнорідних джерел даних найперспективнішим видається підхід, реалізація якого дозволить звертатися до даних, трансформуючи їх у сумісні формати у реальному часі і не зберігаючи в проміжних сховищах. Це дозволить доповнити діяльність СЦД широкими можливостями обміну даними і доступу кінцевих користувачів до ресурсів усіх СЦД, незалежно від їх розташування, форматів даних і програмних засобів аналізу і оброблення даних, що складе надійну основу для розвитку міждисциплінарних досліджень.

Але реалізація такого підходу вимагає розв’язання наведених вище задач.

Огляд існуючих рішень

Спроби вирішення проблеми інтеграції різнорідних джерел наукових даних у самому ж науковому середовищі робилися вже досить давно. Фактично ця проблема ініціалізована появою і широким впровадженням перших промислових СУБД. В реальних умовах формування розподілених систем оброблення інформації, важливу складову яких становили розподілені бази даних, були сформульовані ідеї багаторівневих архітектур, сформувалися уявлення про моделі даних і роль концептуального моделювання, проектування систем баз даних

як реалізацію міжрівневих відображень. По суті термінологічну основу інтеграції закладено у відомому всім представникам старшого покоління IT-фахівців звіту ANSI/X3/SPARC.

Щоб розібратися у величезному різноманітті існуючих систем, підходів, технологій і засобів інтеграції, визначимося із ознаками їх класифікації і критеріями оцінювання. Систематизуватимемо існуючі рішення за такими ознаками:

а) вид інтеграції джерел даних: EII (Enterprise Information Integration); ETL (Extract, Transform, Load); EAI (Enterprise Application Integration);

б) спосіб організації взаємодії джерел даних: з посередником; peer-to-peer (P2P);

в) місце розташування описів джерел даних (каталогу): на центральному вузлі; повної копії загального каталогу на кожному вузлі; зберігання на кожному вузлі каталогу тільки тих об'єктів, які на ньому зберігаються; гібридний варіант;

г) архітектура програмної реалізації інтегрованої інформаційної системи: традиційна; клієнт-серверна; web-орієнтована;

д) рівень описання і маніпулювання даними: синтаксичний; семантичний;

е) метамодель описання інтегрованих даних: базова (реляційна, об’єктно-орієнтована, об’єктна); моделі класу NoSQL; простір даних; моделі дескриптивної логіки (в термінах OWL, RDF); структуровані моделі на основі XML; гібридні (SQL/XML) та ін.

За критерії оцінювання систем інтеграції інформаційних ресурсів виберемо [5]: гетерогенність; розподіленість; автономність. За додаткові критерії їх оцінювання приймемо доступність, цілісність і ефективність.

В умовному координатному просторі зазначених критеріїв досить рел’єфно проявляється місце кожного з підходів до інтеграції, які, у свою чергу, характеризуються набором ознак. Стисло розглянемо особливості систем інтеграції, які закріплюємо за наведеними ознаками.

Розпочнемо з *виду* інтеграції, який визначається на основі праць [6, 7]. Реалізація систем EII, які використовуються для інтеграції даних в режимі реального часу, здійснюється на основі загального шлюзу (gateway) з єдиними мовою і точкою доступу до неузгоджених джерел даних. За рахунок віртуалізації вони надають застосуванням і користувачам гнучкий, незапланований доступ до даних з будь-якою періодичністю і обсягами.

Системи ETL, які використовуються для пакетної інтеграції, передбачають вилучення даних із гетерогенних джерел, їх трансформацію і завантаження в єдине сховище даних. Вони звичайно застосовуються як сховища добре документованих надійних даних для багатовимірного аналізу, наприклад, часових рядів. Системи цього класу забезпечують видалення дублювання, перевірку якості даних та інші процеси обслуговування співробітників організацій і підтримки їх комплексів бізнес-процесів.

Системи EAI використовуються для інтеграції програм. Перетворення даних виконуються застосуваннями, за рахунок координованої взаємодії яких і забезпечується інтеграція. Традиційно системи цього класу підтримують взаємодію застосувань в реальному часі для автоматизації бізнес-процесів. Вони забезпечують узгодженість при внесенні змін у застосування.

Важливим для розуміння сутності інтеграції джерел даних є їх поділ за *способом організації взаємодії джерел даних*. Для систем інтеграції з посередником, які базуються на концепції медіатора [4], клієнти підключаються до mediator-компонентів (посередників) як показано на рис. 1. Медіатор здійснює розбір, переписування та оптимізацію запиту, виконує деякі з його операторів. Для цього використовується каталог, у якому зберігається глобальна схема гетерогенної системи баз даних, встановлюється відповідність її частин кожному компоненту бази даних. На основі каталогу формулюються запити прикладних програм і користувачів, зовнішніх компонентів, оптимізуються запити.

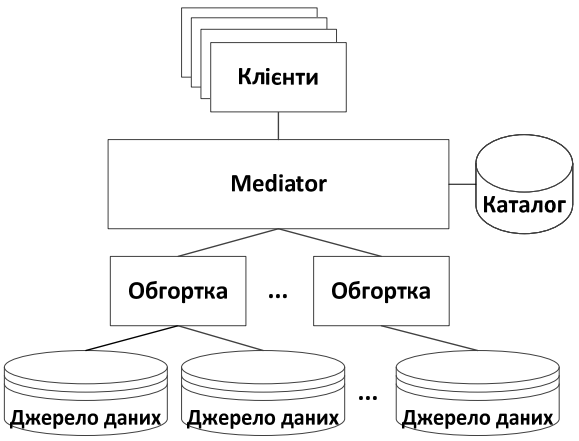


Рис. 1. Загальна схема системи з посередником

Щоб об’єднати деталі компонентів бази даних, створюються обгортки («wrapper», «адаптер»), пов'язані з кожним компонентом бази даних. Обгортки транслюють кожен запит ме-

діатора, щоб він був зрозумілий цільовій базі даних, і одержувані від бази даних результати, щоб вони були зрозумілі медіатору і сумісні з зовнішньою схемою бази даних і глобальною схемою гетерогенних баз даних. Створення обгорток не становить складну проблему.

Розширюваність забезпечується тим, що обгортки та компоненти бази даних можуть бути оновлені або нові компоненти баз даних можуть бути інтегровані без зміни посередника та існуючих обгорток.

Реалізації на основі медіаторів, насамперед, DISCO, Garlic, IBM Data Joiner, інструментів для побудови розподілених систем баз даних продемонстрували їх високу ефективність, особливо для інтеграції реляційних і об'єктно-орієнтованих баз даних. Ці інструменти вдало використовують переваги трирівневих архітектур програмних систем і видаються дуже зручними для інтеграції розподілених, насамперед, семантизованих джерел даних.

Структуру систем інтеграції даних на основі взаємодії P2P [8] складають автономні вузли (джерела даних), пов'язані між собою за рахунок відображень. Кожний з них надає свою частину інформації, доступної з розподіленого середовища. Цим децентралізованим системам притаманні гнучкість, здатність обробляти динамічні зміни в системі, оскільки вузли можуть підключатися або відключатися, не перериваючи роботу всієї системи.

Важливо розуміти, що в цих системах кожний вузол є системою інтеграції даних з медіатором, управляючи локально пов'язаними джерелами даних (локальний мапінг) за рахунок трансформованих в глобальну вузлову схему локальних відображень (глобальний мапінг). Вузли запрошують інформацію у будь-якого вузла (від зовнішніх користувачів або інших вузлів) на основі відображень, які зберігаються в їх специфікаціях. Опитуваний вузол завдяки мапінгу для формування відповіді може використовувати дані інших вузлів. Покладаючись тільки на доступні у вузлах сервіси, вузли системи ефективно надають відповіді на запити в термінах їх схем даних. Конвертація усіх відповідей в єдиний формат з використанням відображення між вузлами може виявитися складною, як і обмін даними і відображеннями даних між вузлами.

Щодо переваг і недоліків систем інтеграції, виділених за ознакою *місця розташування описів джерел даних*. Загалом описи джерел даних (метаописи, каталог для збереження метаданих) містять інформацію, яка дозволяє одним вузлам звертатися до об'єктів баз даних, що містяться на інших вузлах. Перші два варіанти розташування незадовільні за критерієм автономності, оскільки порушують принцип незалежності вузлів від центрального вузла або частково усіх вузлів, адже будь-яке оновлення каталогу повинно виконатися на всіх вузлах системи. Третій варіант зберігання незадовільний за критерієм ефективності, оскільки пошук віддаленого об'єкта вимагатиме звернення в середньому до половини вузлів.

Існують ефективні реалізації поєднання цих підходів. Так, на кожному вузлі однієї з перших промислових федеративних СУБД IBM System R, зберігалася інформація про локальні об'єкти даного вузла і частина інформації загального каталогу – інформація про створені на цьому вузлі об'єкти. Складові системного імені кожного об'єкта однозначно ідентифікували його в федеративній базі даних [9].

Архітектура програмної реалізації інтегрованої інформаційної системи є визначальною для досягнення критеріїв інтеграції. Різноманітні успадковані традиційні архітектури – від файлових систем до пакетів програм – загалом мають незначні інтеграційні можливості і їх включення в множину джерел вимагає багатьох зусиль. Клієнт-серверні трирівневі архітектури з виділеними серверною і клієнтською частинами і сервером баз даних дуже зручні для інтеграції на синтаксичному рівні баз даних, особливо реляційних і об’єктно-реляційних і меншою мірою об’єктних. Web-орієнтовані архітектури зручні саме для інтеграції на семантичному рівні, оскільки мають стандартизовані засоби обміну даними і взаємодії в термінах онтологій.

Системи інтеграції з *синтаксичним і семантичним рівнями* описання і маніпулювати даними відрізняє низка особливостей, які обумовлюють їх переваги і недоліки [10]. Синтаксичні структура даних орієнтована на однорідні набори даних, тоді як семантичні – на зв’язки та відношення між одиницями даних без залежності від їх подібності. Якщо у перших на запит одних джерел вибираються дані з інших джерел і перетворюються у відповідності із заданими вимогами, то у других встановлюються зв’язки між одиницями даних у відповідності до визначень у їх онтологічних описах. До того ж, на відміну від синтаксичної інтеграції, зникає потреба у завантаженні даних у проміжні сховища.

При синтаксичній інтеграції вартість з кожним новим джерелом даних зростає експоненційно, тоді як при семантичній практично не залежить від кількості джерел, оскільки у цьому випадку потрібно лише підготувати описи нових джерел. Традиційні для синтаксичної інтеграції жорстка стандартизація та форматування, порушення яких призводить до втрати контексту, поступаються при семантичній інтеграції гнучкому дотриманню стандартів, можливості введення унікальних типів у межах кожного опису джерела даних.

Насамкінець розглянемо особливості систем інтеграції, виділених за ознакою *метамоделі описання інтегрованих даних*. Оскільки більшість з них широко відомі, стисло охарактеризуємо концепції семантичної інтеграції даних, побудовані на методах визначення семантики предметної області. Для інтеграції гетерогенних інформаційних ресурсів необхідно забезпечити інтероперабельність на семантичному рівні, що породжує задачі порівняння вмісту ресурсів, відшукання відповідностей і розв'язування конфліктів між ними, а також проблему сполучення неоднорідних ресурсів [11].

Повна специфікація ресурсу повинна охоплювати специфікації: власне джерела даних; обмежень цілісності; контексту (області, у якій передбачається використання ресурсу).

Структура і поведінка інформаційного ресурсу визначена семантикою його предметної області. Умовою ефективного використання семантики даних є подання їх специфікації деякою формальною мовою. Теоретичну основу такої мови може скласти мова числення предикатів першого порядку. Розвиток web-технологій сприяв появі мов представлення знань в недвозначному, формалізованому вигляді на основі дескриптивної логіки [12].

Як інструмент створення таких узагальнених специфікацій можна використати онтологічні специфікації, розуміючи під онтологією формалізований опис загальноприйнятого розуміння деякої предметної області, за допомогою якого можуть спілкуватися люди, комп'ютерні системи [13]. Для інформаційного компонента така специфікація становить набір визначень і понять, а також правил (аксіом), пов'язаних з визначеннями і поняттями предметної області.

Онтологія визначає загальний словник для спілкування з використанням інформації предметної області На відміну від метаданих, таких як тип, розмір атрибута, онтології повинні мати виразніші засоби визначення семантики даних, насамперед традиційну ієрархію понять і типів об'єктів, разом з точним описом кожного типу, аксіоми, які задають обмеження на інтерпретації цих понять.

Розроблення сучасних програмних систем, а надто їх інтеграція, має базуватися на стандартних онтологіях, які дозволяють спільно використовувати інформацію відповідних предметних областей людьми і комп'ютерами. Уже розроблено багато онтологій, що уможливлює повторне використання, аналіз і класифікацію знань, явне формулювання припущень в предметній області.

Оскільки розроблення онтологій полягає у визначенні набору даних та їх структури для використання іншими програмами, для кодування онтологій необхідна формальна мова — мова опису онтологій. Обмін програм та програмних агентів даними, описаними в термінах онтологій, дозволить забезпечити їх реальну взаємодію, незалежність від форматів представлення даних. Потреба в онтологічних моделях на основі логік стимулюється реалізацією Semantic Web — нової концепції розвитку Інтернет, підтримуваної World Wide Web Consortium (W3C). Світовою IT-спільнотою розроблено багато мов описання онтологій, наприклад Knowledge Interchange Format, Common Logic, Cycl, Simple HTML Ontology Extension (SHOE), Ontology Markup Language (OML). Вони забезпечують подання зрозумілого для комп'ютерів семантичного знання в документах Web, дозволяючи програмам і програмним агентам збирати значиму інформацію на Web-сторінках і документах, поліпшуючи механізми пошуку і опрацювання знань. Ядерна частина цих мов пов'язана з логічними аспектами, а онтологія визначається як структура, компоненти якої охоплюють декілька рівнів представлень, наприклад концептів, об'єктів, відношень, функцій, аксіоми і онтологій, використовуючи концептуальні графічні можливості. Найбільш поширені такі основні мови:

1. XML (eXtensible Markup Language) — метамова, розроблена W3C для реалізації web-орієнтованих програмних систем і інтероперабельності з Standard General Markup Language (SGML) та HTML. Вона має зручний для аналізу програмами і людського сприйняття синтаксис, який дозволяє визначати складні структури даних, опрацьовувати документи. Це перетворює її у простий і потужний спосіб визначення

мов специфікацій онтологій і засіб обміну онтологіями [14];

2. DARPA Agent Markup Language + Ontology Interchange Language (DAML+OIL) — сформований на стандартах W3C результат злиття мов DAML і OIL, розроблений для надання семантичної інтероперабельності в XML. Вона має розширений набір потужних примітивів моделювання і використовується для формування онтологій з формальною семантикою і сервісами логічного висновку на основі дескриптивної логіки, обміну онтологіями.

3. Ontology Web Language (OWL) [15] — мова описання Web-онтологій, розроблена W3C як розширення Resource Description Framework (RDF) і RDF Schema (RDFS). OWL забезпечує поряд з формальною семантикою додатковий словник, що закладено в основу DAML + OIL, а також вбудовану підтримку відображення онтології. Вона має три діалекти Lite, DL і Full, які відрізняються формальною складністю і відповідно виражальними можливостями: від класифікаційної ієрархії і простих обмежень до максимальної виразності і застосування аксіом і правил для логічного висновку і складними обмеженнями та синтаксичною свободою RDF без гарантій обчислення. На сьогодні вони забезпечують потреби створення програмних систем інтеграції і підтримку взаємодії застосувань і через них бізнес-процесів у семантичних термінах [16].

Результати огляду мов описання онтологій для Semantic Web дозволяють зробити висновок, що вони відповідають потребам створення програмних систем інтеграції. Залишається розглянути засоби семантичного описання інформаційних ресурсів та виконання запитів до них.

Для описання інформаційних ресурсів використовується рекомендована W3C мова RDF [17]. Ця мова є абстрактною моделлю і описує ресурси у вигляді орієнтованого позначеного графа – кожний ресурс може мати властивості, які в свою чергу також можуть бути ресурсами або їх колекціями [18]. RDF, забезпечуючи універсальний, гнучкий метод декомпозиції знань на маленькі частини – триплети і враховуючи їх семантику, дозволяє задати структуру опису джерела (бази даних, XML-документу, Web-сайту тощо). При цьому можна використовувати і розширювати вбудовані поняття RDF-схем, такі як класи, властивості, типи, колекції, успадкування класів і властивостей. Крім опису структури, RDF дозволяє визначати і оперувати твердженнями на основі предикатів, що дозволяє подавати концептуальну інформацію.

Найбільш популярною серед мов запитів до RDF-сховищ на сьогоднішній день є мова SPARQL Protocol and RDF Query Language [19]. Вона визначає тільки запити до інформації, що міститься в моделях, а реалізацію інтерфейсу для мови запитів забезпечує бібліотека для розроблення семантичних веб-застосувань Jena [20]. SPARQL отримує вимоги застосування у формі запиту і повертає цю інформацію у вигляді набору зв'язків або графа RDF.

Тепер розглянемо існуючі рішення для інтеграції інформаційних ресурсів. Розпочнемо із засобів генерації і виконання семантичних запитів до розподілених джерел даних, які можуть надавати дані у форматах абстрактної моделі RDF.

Оброблювач ARQ запитів SPARQL для Jena надає розширення мови запитів для виконання запитів до розподілених джерел, розширюючи SPARQL для передавання шаблону триплету до віддаленої кінцевої точки доступу [20]. Це не є всеосяжним рішенням розподіленої обробки запитів, оскільки клієнт повинен знати джерело, до якого необхідно направляти запит, що неможливо при отриманні єдиного вхідного запиту, згенерованого автономним сервісом.

Проект DARQ пропонує єдиний інтерфейс для виконання запитів до довільної кількості розподілених SPARQL кінцевих точок і здійснює прозорий для клієнта поділ запиту на підзапити та їх виконання (федерування) [21]. Повністю побудований на SPARQL-стандарті, DARQ використовує сервіс-дескриптори для динамічного додавання і вилучення кінцевих точок, а також оптимізує запити на основі статистичної інформації.

Розвиток закладеного в DARQ підходу збагатив його новими ідеями. Так, в проекті SemWIQ [22] підтримуються запити до розподілених джерел даних у розширеному SPARQL-синтаксисі, підтримується робота з багатьма кінцевими точками з використанням конструкції COUNT. Оптимізація здійснюється на основі статистики про наповненість джерел даних, яка отримується моніторинговим компонентом в умовах наявності обмежень автономності, приватності та ін.

У праці [23] розроблено адаптивний алгоритм виконання запитів на SPARQL-кінцевих точках, який на відміну від статичних технік оптимізації застосовує динамічне об'єднання результатів виконання підзапитів лише за наявності зв'язуючих значень.

У проекті FedX [24] запропоновано ряд нових стратегій оброблення SPARQL-запитів до розподілених джерел даних на основі побудови під час виконання запиту єдиного об'єднаного RDF-графа. Якщо джерела даних надають свої SPARQL-кінцеві точки немає потреби у внесенні змін у їх структури даних.

Із реальних проектів варто згадати проект німецьких вчених PANGAEA під керівництвом Майкла Діпенброка [25], російський проект ЕСИМО (Єдина державна система інформації про обстановку в Світовому океані) [26] та ін. Так, перший із них присвячений створенню інформаційної системи PANGAEA за принципом бібліотеки відкритого доступу, призначеної для архівування, публікування і розповсюдження геоцентричних даних системних досліджень нашої планети. Система забезпечує довготерміновий доступ до контенту через інформаційні ресурси низки організацій. Кожний з наборів даних можна ідентифікувати, розділяти, публікувати і цитувати. Дані архівуються як додатки до публікацій або цитовані колекції даних. Цитування доступне через портал German National Library of Science and Technology.

Проект ЕСИМО присвячений створенню інформаційної системи для забезпечення морської діяльності Росії інформацією щодо ситуації у Світовому океані. Система автоматизує функції збору потрібних даних, їх збереженняі опрацювання з широким охопленням системних об’єктів, що перетворює її осередок міждисциплінарних досліджень.

Однак, жоден з цих програмних комплексів не може забезпечити повну інтеграцію різнорідних джерел даних. На жаль, часто недоступні вихідні коди розроблених рішень або в структурі їх даних не завжди може визначитися навіть досвідчений користувач. Крім того, ці проекти спрямовані на використання обмеженого набору моделей даних джерел. Для вирішення проблеми інтеграції потрібен більш гнучкий і універсальний підхід. Однак, справа ускладнюється тим, що існує декілька підходів-претендентів на провідне місце. Серед них досить перспективними видаються підхід, побудований на концепції просторів даних.

Ця концепція запропонована А.Гелеві [27], реалізована в ряді проектів, а в Україні розвивається Н.Б. Шаховською [28]. За умови створення конструктивного формалізму як основи визначення, пошуку і опрацювання даних проміжного шару можна сподіватися, що користувачі отримають ефективну і зручну технологію інтеграції. Але на сьогодні це лише загальна концепція, реалізація якої вимагає багато зусиль.

Отже, кожне з наведених рішень побудоване на цікавих ідеях, вирішує конкретні бізнесові або наукові проблеми. Вони оптимізовані під конкретну задачу, або навпаки, є настільки уніфікованими, що не можуть в повному обсязі бути використані для вирішення поставленої в цій роботі задачі. З іншого боку, всі оглянуті роботи по виконанню запитів до розподілених джерел даних не розраховані на взаємодію з не-RDF даними, не розглядають проблеми їх структурування та класифікації, що дуже важливо при роботі з інформаційними ресурсами Світової системи даних.

Постановка проблеми дослідження

В центрах Світової системи даних є багато формально не пов’язаних джерел даних, побудованих на базі різних інформаційних технологій – бази даних під управлінням СУБД, текстові файли, XML-документи, NoSQL-дані тощо. Необхідно, не вносячи зміни у структури джерел даних, що склалися історично, зробити їх доступними для користувачів за принципом «єдиного вікна». Одночасно необхідно забезпечити можливість виділення пов’язаних полів даних і виконання єдиного запиту до декількох джерел даних. Ще одним важливим аспектом проблеми є забезпечення можливості семантичного «розфарбування» існуючих даних, що зробить їх придатними для машинного оброблення із застосуванням технологій семантичного рівня і більш ефективного пошуку.

Загальний підхід до вирішення проблеми інтеграції джерел даних

Для описання специфікації семантики джерел даних обрані дескриптивні логіки. Як інструмент подання узагальнених специфікацій обрано онтологічні специфікації. Позначимо онтологічний клас і властивість в контексті семантики як C-клас і C-властивість відповідно.

Для реалізації системи інтеграції будемо застосовувати стек технологій Semantic Web – абстрактну модель даних у вигляді орієнтованих графів «суб'єкт-предикат-об'єкт» RDF, мову описання онтологій OWL і мову запитів до RDF-подібних даних SPARQL.

Для високорівневого описання доступних в системі предметних областей необхідно створити центральну онтологію-класифікатор, яка складається з ієрархії С-класів і не залежить від джерел даних. Центральна онтологія може бути розширена іншими онтологіями такої ж структури або парами онтологія-відображення. Кожна така пара забезпечує підключення для конкретного джерела даних. Для цього онтологія описує структуру і зв'язки його С-класів і С-властивостей, а відображення вказує співвідношення його С-класів і С-властивостей до реальної структури джерела даних, а також параметри підключення до нього. Таким чином, запит до джерела даних може формуватися в термінах С-класів та С-властивостей.

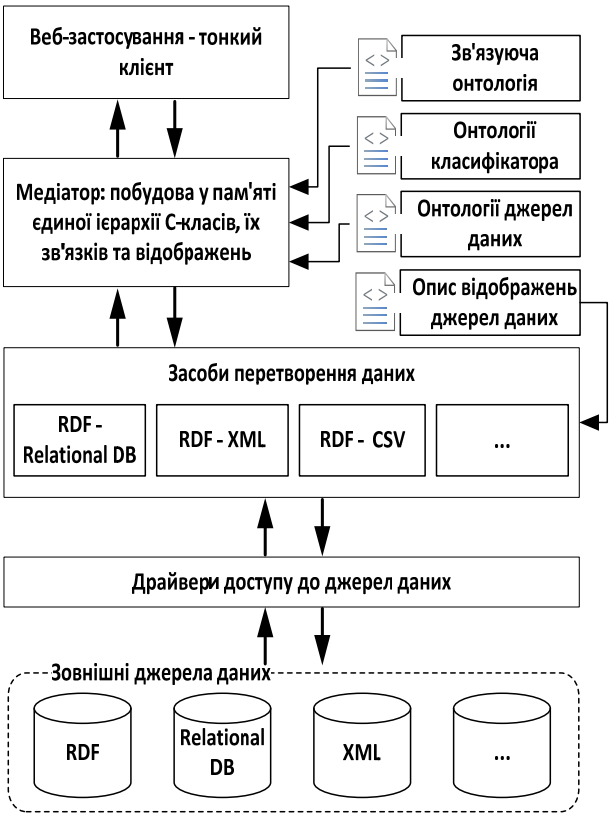


Рис. 2. Загальна схема рішення

Оскільки різні джерела даних можуть мати спільні С-властивості, створимо ще й зв'язуючу онтологію для встановлення зв'язку між такими С-властивостями різних джерел даних за допомогою стандартних предикатів мови OWL.

На рис. 2 наведено загальну схему запропонованого рішення інтеграції джерел даних класу EII з посередником і web-орієнтованою архітектурою. На боці клієнта вибір даних в каталозі, формування запиту до джерел даних, а також отримання відповіді на нього виконується за допомогою web-застосування. Медіатор є завантаженим в пам'ять сервера застосуванням і містить побудоване дерево С-класів з онтологічних специфікацій (саме їх користувач бачить як каталог), їх зв'язки із відображеннями на реальні джерела даних, а також, опціонально, отримані з онтологічних специфікацій та опитування джерел даних обмеження на допустимі діапазони значень даних.

Зв'язування онтологій на рівні застосування здійснюється за рахунок встановлення зв'язків їх С-класів з використанням унікального для кожного С-класу Uniform Resource Identifier (URI).

За прийнятими критеріями (гетерогенність, розподіленість і автономність) запропоноване рішення переважає рішення побудовані на множинних (multidatabase) і федеративних (federated database) базах даних, вже не кажучи про монолітні бази даних. Але за додатковими критеріями перевага не так відчутна.

Оброблення запитів до розподілених неоднорідних джерел даних. Концепція пропонованого рішення, схема оброблення запитів до розподілених джерел даних у якому наведена на рис. 3, позбавляє користувача від необхідності розуміння структур цих джерел, їх характеристик і особливостей мов запитів.

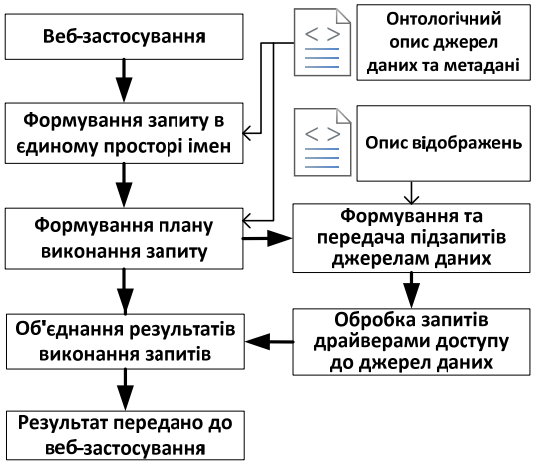


Рис. 3. Схема оброблення запитів

Точкою взаємодії з користувачем є web-застосування – тонкий клієнт. Він дозволяє користувачу формувати запит, оперуючи С-класами, С-властивостями, їх зв'язками та обмеженнями в глобальному просторі імен за допомогою меню-орієнтованого інтерфейсу. Інформація про структуру С-класів і С-властивостей, а також їх зв'язків завантажується з онтологічних описів джерел даних, а загальне дерево С-класів будується з онтології-класифікатора.

Далі застосування формує план виконання запиту, визначаючи, які джерела даних можуть

відповісти на запит або його частини, і чи можливо ці результати об'єднати. При цьому враховуються характеристики джерел даних, а також правила виведення, що визначають можливі операції над С-класами та їх С-властивостями, які отримані з онтологічних описів. Звідти ж береться інформація для формування умов-обмежень для джерел даних і запитів до них.

Після цього запит трансформується в єдиному просторі імен на основі описів відображень між джерелами даних. Він конвертується в запити до окремих джерел даних у відповідному синтаксисі, наприклад, у форматі SQL, і просторі імен.

Сформовані запити передаються у відповідні джерела даних за допомогою стандартних адаптерів, наприклад, JDBC. Обробка запитів адаптером доступу до джерела даних є стандартною практикою при роботі з СУБД. Джерела даних функціонують автономно. Після отримання запиту вони його виконують і повертають результат у вигляді набору записів.

Отримавши від джерел даних результати виконання запитів, застосування дотримуються раніше сформованого плану виконання запиту і виконують необхідні операції з ними – об'єднання, перетин, узагальнення, передача сервісам-обробникам і т.д.

Базова реалізація припускає, що всі джерела даних, застосовуючи стандартизовані адаптери, можуть обробляти SQL-або SPARQL-запити. Для взаємодії з джерелами даних, які надають XML-файли, або використовуються специфічні адаптери, або ці файли приводяться у формат онтологій з метаданими, описаними у XML-подібному синтаксисі. Перший підхід вимагає налаштувань і вартісних програмних продуктів, а другий – втручання у структури джерел даних на боці провайдерів. Наступні реалізації будуть охоплювати більш широке коло джерел.

Розглянемо модель для формування єдиного запиту до множини джерел даних користувачем у єдиному просторі імен. Користувач оперує ієрархією С-класів, зв'язками між ними (описані owl-предикатами SubclassOf і SameAs), а також С-властивостями, які можуть бути у кожного С-класу, і обмеженнями на ці С-властивості. На рівні логіки застосування відомо, які джерела даних пов'язані з якими С-класами і С-властивостями, але для користувача це не має ніякого значення.

Послідовність дій при формуванні запиту:

1. Вибір необхідного С-класу в дереві класифікатора.

2. Встановлення обмежень на значення його С-властивостей (за їх наявності) за допомогою констант або логічних виразів.

3. (Опціонально) встановлення зв'язків з С-властивостями інших С-класів з формуванням Join-запиту.

Таким чином, на виході може бути отримано один або більше одного С-класів, набір С-властивостей і обмежень на них, С-властивості, через які треба об'єднувати значення для отримання кінцевого результату. Виділимо чотири типи стандартних запитів до розподілених джерел даних, які можуть бути оброблені в межах цієї моделі:

1. *Запит до однієї сутності* – часто вживаний запит на отримання підмножини набору даних з одного джерела даних за деякою умовою. Класичним прикладом такого запиту є Select-подібний запит. У випадку запиту до даних з інтегрованою семантикою такий SPARQL-запит буде містити URI потрібного С-класу, за яким закріплене тільки одне джерело даних, і при цьому з інформації про діапазонах допустимих значень (при її наявності) випливає, що запит може бути виконаний на цьому джерелі.

2. *Запит, в якому сутність рознесена в декількох джерелах даних* (*Union*) – запит на отримання за деякою умовою даних результату об'єднання С-властивостей С-класів (які можна представити як таблиці) згідно їх специфікації з різних джерел даних. Тоді йдеться про вхідний запит з одним С-класом і умовою. При цьому у декількох джерелах даних відображено зазначений С-клас і виконується частково або повністю зазначена умова.

3. *Запит з об'єднанням сутностей* (*Join*) – запит на отримання перетину результатів кількох вибірок за С-властивостями різних С-класів, що може бути здійснено за наявності спільних С-властивостей у таких С-класів у зв'язуючій онтології. Зв'язки між зв'язуючими С-властивостями дозволять об'єднати запити для утворення в результаті об'єднаного за допомогою Inner Join набору значень.

4. *Запит на узагальнення* – запит на отримання з джерел даних наборів таблиць або описів С-класів, С-властивостей та їх зв'язків, які не підлягають об'єднанню, але, згідно опису в онтології-класифікаторі, вони є підкласами обраного користувачем С-класу. На практиці

отримання повних таблиць даних з усіх підкласів обраного С-класу може бути недоцільним і створити надмірне навантаження на потужності серверів-провайдерів, і це питання може потребувати окремого вивчення.

План виконання запиту та його оптимізація. Ефективність системи інтеграції даних визначається компонентом планування виконання запиту та його оптимізації. Оскільки джерела даних функціонують автономно, статистичні дані про їх наповнюваність отримати не завжди можливо. До того ж доцільність виконання надскладних запитів до підключених джерел даних, враховуючи їх специфіку у ССД і невелику кількість поєднуваних властивостей, є сумнівною.

Дієвим рішенням проблеми планування та оптимізації запитів є підхід на основі статистичної інформації, яку надають самі постачальники даних. Для неї виділено окреме поле метаданих в онтологічному описі джерела даних, яке заповнюється цілим числом, що вказує на орієнтовну кількість кортежів у джерелі даних в межах описуваної таблиці або описуваного перетину множини таблиць.

Таким чином, за необхідності побудови запиту двох і більше джерел даних, обробник генерує дерево-план виконання запиту, яке відпрацьовує запити до джерел з найменшою кількістю записів (рядків), передаючи в запиті до кожного наступного (пов'язаного з ним) джерела даних їх результат як параметр і одержуючи їх об'єднання на виході. За умови виконання запитів до більше, ніж одного джерела даних – вони виконуються паралельно. Тоді на кожному кроці відкидається певна кількість нерелевантних записів, як при SQL-запиті Inner Join, а на виході – отримуємо єдину таблицю із результатом виконання запиту.

Для побудови плану виконання запиту із врахуванням коефіцієнтів ємності джерел даних необхідно виконати такий алгоритм:

Крок 1. Отримання на вхід запиту у єдиному просторі імен, який визначає множину джерел даних і набір С-властивостей, за якими можуть бути об'єднані результати запитів до даних джерел даних, коефіцієнтів ємності кожного джерела даних;

Крок 2. Знаходження добутку коефіцієнтів ємності кожної пари джерел даних, підзапити до яких підлягають об'єднанню;

Крок 3. Виділення пари джерел даних, підзапити до яких підлягають об'єднанню;

Крок 4. Впорядкування цих пар у порядку зростання добутку коефіцієнтів ємності;

Крок 5. Виділення джерел даних, запити до яких можуть бути виконані паралельно (не потребують вхідних параметрів – результатів виконання запитів з інших джерел) та мають найменші коефіцієнти ємності;

Крок 6. Паралельне виконання запитів до виділених на кроці 5 джерел даних;

Крок 7. Виділення з отриманих на кроці 6 результатів даних, за якими виконується перетин із джерелом-парою та передавання як параметра в запиті до цього джерела-пари;

Крок 8. Представлення отриманого перетину результатів запитів до кожної пари джерел даних як віртуального джерела даних, коефіцієнт ємності якого дорівнює кількості кортежів отриманого результату, а запит вже було виконано;

Крок 9. Виконання кроків 3 – 8 доки не буде отримано єдиний кортеж, який і стане результатом виконання запиту користувача.

Формування запитів безпосередньо до джерел даних. Поля таблиць джерела даних приводяться у абстрактну модель даних у вигляді графу «суб'єкт-предикат-об'єкт» і навпаки за допомогою опису відображень (мапінг). Представимо опис відображення як

$$f: R \leftrightarrow S,$$

де R – вихідна структура джерела даних, S – абстрактне семантичне представлення у вигляді графу.

Для конвертації запитів, виділених до конкретних джерел даних, в SQL-формат з урахуванням властивостей і параметрів цих джерел згідно опису відображень використаємо алгоритм безпосередньої конвертації запитів, наведений у праці [29].

Побудова зв'язаної структури даних із онтологій та підключень до джерел даних. Важливою задачею є побудова у пам'яті застосування зв'язної ієрархії із онтології-класифікатора, розширюючих онтологій, онтологій джерел даних, онтологій, що зв'язуюсь С-властивості джерел, а також їх співставлення з описом відображення джерел даних. Для цього будемо використовувати такий алгоритм:

Крок 1. Завантаження онтології-класифікатора, побудова відповідного головного дерева С-класів.

Крок 2. Послідовне завантаження розширюючих онтологій з під'єднанням їх у пам'яті до головного дерева. Послідовність завантаження розширюючих онтологій може бути визначена

чергою (це необхідно, наприклад, для випадку, коли одна розширююча онтологія розширює іншу).

Крок 3. Послідовне завантаження онтологій-описів джерел даних. При цьому такі онтології можуть мати декілька рівнів підкласів, що робить їх порівнюваними з розширюючими онтологіями.

Крок 4. Приєднання завантажених онтологій-описів джерел даних до головного дерева, співставлення співпадаючих С-класів і їх С-властивостей та визначення їх (за наявності) як таких, що доступні у декількох джерелах даних.

Крок 5. Завантаження звя'зуючої онтології, проекція її на збудоване головне дерево С-класів і С-властивостей. Результатом є граф, який пов'язує деякі листки (С-властивості) збудованого головного дерева.

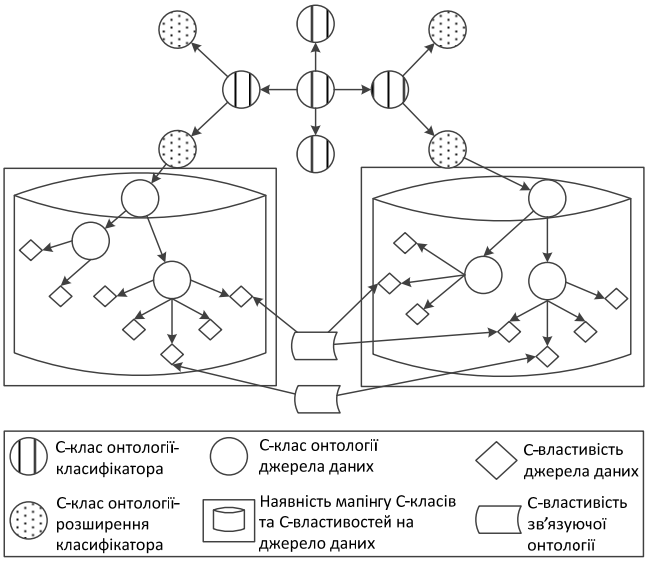


Рис. 4. Приклад об'єктної моделі об'єднаних онтологій та описів джерел даних

Крок 6. Завантаження описів відображень джерел даних і встановлення їх зв'язків з С-класами та С-властивостями.

Приклад побудованого в пам'яті застосування дерева С-класів, С-властивостей, їх зв'язків між собою та з джерелами даних наведений на рис. 4.

Структура і принципи функціонування системи. Структура системи визначається потребою в інтеграції багатьох джерел даних при розв'язанні міждисциплінарних наукових задач. Використання дескриптивних логік для описання джерел даних, модульність, готовність до приєднання ланцюжків сервісів для додаткового оброблення даних приводять до розподіленої системи з центральним вузлом, який працює на сервері застосувань як Java Enterprise компонент с налаштованою точкою доступу. Для приєднання кожного джерела даних добавляються OWL-файл з семантичною маскою метаданих і файл властвостей для безпосереднього підключення зовнішнього джерела з уточненням типу драйвера, IP-адреси тощо. Зрозумілою також стає роль користувацького інтерфейсу і системи конструювання запиту.

Компоненти системи функціонують на основі моделей, визначених нижче. Послідовність роботи компонентів:

1. Генерування SPARQL-запитів за допомогою конструктора або отримання готового запиту від користувача.
2. Парсинг, переписування і оптимізація запиту.
3. Разбиття запиту на підзапити.
4. Виконання підзапитів на вибраних джерелах даних.
5. Очікування отримання відповідей від усіх джерел даних.
6. Об'єднання отриманих результатів за допомогою Inner Join (опціонально).
7. Повернення користувачу єдиної таблиці або множини таблиць (якщо об'єднання неможливе або користувач вирішив його не виконувати) як результат виконання запиту.

Моделі і методи для реалізації окремих етапів

Моделі визначення семантики предметної області. Створення інформаційної системи для інтеграції неоднорідних інформаційних ресурсів, забезпечення їх семантичної інтероперабельності вимагає вирішення проблем порівняння вмісту цих ресурсів, встановлення відповідностей і запобігання конфліктів між ними, а також проблему сполучення різнорідних ресурсів. Як зазначалося вище, повна специфікація інформаційного ресурсу буде охоплювати специфікації власне джерела даних, обмежень цілісності і предметної області. Використання концепції медіатора вимагає описання структури і поведінка кожного інформаційного ресурсу. Вони визначаються семантикою відповідних предметних областей, специфікації якої описуватимемо на основі дескриптивної логіки. Її базові елементи:

- множина класів NC;
- множина індивідуумів NI;
- множина відношень NR.

Для описання інформаційних ресурсів будемо використовувати RDF і RDFS. Засновані на

ідеях представлення знань, таких як семантичні мережі, фрейми і логіка предикатів, прості в реалізації вони дозволяють визначати класи, ресурси і властивості, ієрархії й обмеження типів, концепти, відношення й екземпляри. Для визначення функцій і аксіом будемо використовувати інші мови.

Для описання онтологій будемо використовувати мову OWL, яка повністю відповідає вимогам до розроблюваних систем інтеграції інформаційних ресурсів.

Машина логічного висновку і перетворення даних у формат RDF. Розпочнемо з бібліотеки для організації мапінгу не RDF-даних на онтологію. Для інтеграції семантики предметної області з джерелом даних, необхідно реалізувати механізм мапінгу концептів онтології відповідним об'єктам джерела даних.

Таку задачу здатна здійснити бібліотека Java D2R, яка забезпечує доступ до даних через SPARQL, Jena API і Sesame API у вигляді RDF-графу. Бібліотека D2R забезпечує зв'язок з джерелом даних через JDBC драйвер, що дозволяє використовувати в якості джерела практично будь-який ресурс – від СУБД до плоских файлів і HTML-документів, а при необхідності реалізувати специфічний драйвер. Роль носія семантики предметної області відіграє OWL-онтологія. Файл мапінгу будується в вигляді RDF-документа в форматі N3, доступ до якого може бути реалізовано методами інструменту Jena, а отже – автоматизовано його створення.

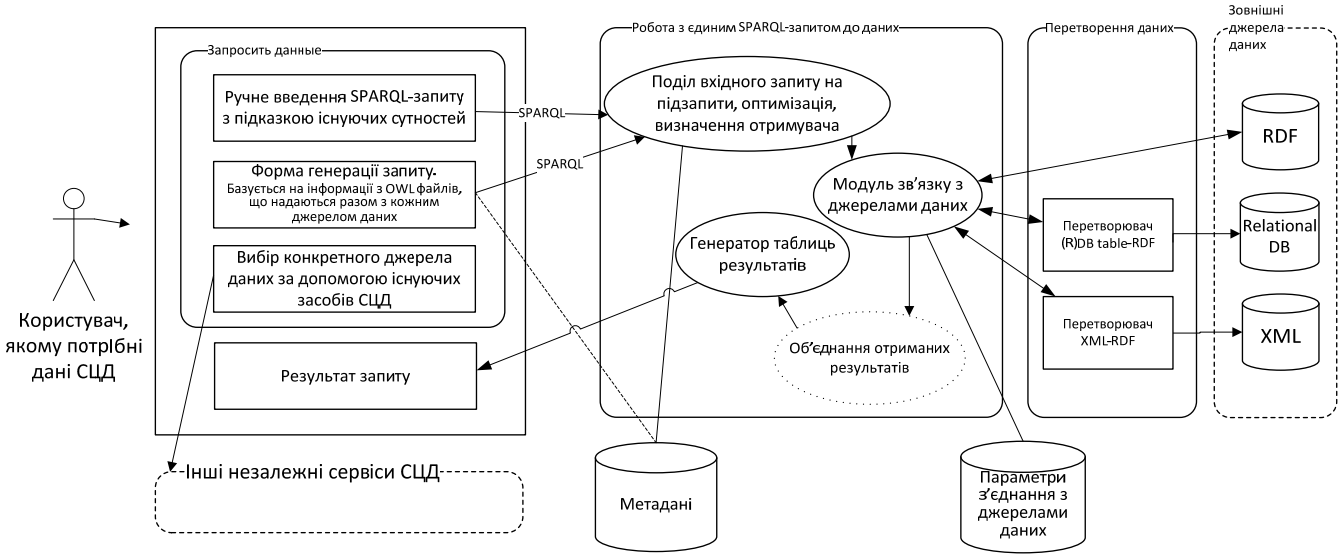


Рис. 5. Загальна схема

Іншим важливим аспектом створюваної системи є представлення користувачу єдиного доступу до метаданих, що включають семантику предметної області та інформацію з джерел даних. Зазначимо важливість простоти в роботі з клієнтською частиною системи. Зокрема, її реалізація в вигляді web-застосування виключить необхідність встановлення додаткового програмного забезпечення та його налаштування.

Наступним необхідним кроком є вибір засобу для створення і маніпулювання RDF-графами. Таким засобом вибрано Jena [30], що є Java API застосуванням. Jena має класи об'єктів для представлення графів, ресурсів, властивостей і літералів. Інтерфейси представлення ресурсів, властивості і літералів називаються відповідно Resource, Property та Literal. У Jena, граф називається моделлю і представлений інтерфейсом Model.

Таким чином, каркас Jena надає високорівневий доступ до графів RDF-документу і тим самим простий і гнучкий спосіб завантаження онтології, маніпулювання триплетами, що в ній містяться, об'єднання декількох графів, наприклад онтології та RDF-документу.

Для ефективного використання семантики предметної області для відповіді на запити користувача, необхідно реалізувати логічний висновок на базі побудованого графу.

Для розв'язання цієї задачі було вибрано машину логічного висновку Pellet [31], що є системою логічного висновку над RDF-даними з OWL DL (дескриптивна логіка). Ця система включає в себе оптимізацію для номіналів, відповіді на кон'юнктивні запити, а також послідовний висновок над новими графами.

Pellet підтримує висновок і надає функції перевірки цілісності, когерентності класу, класифікації для відповіді на запити, такі як отримання всіх або тільки прямих підкласів класу,

визначення прямого типу для індивідів. Важливо, що Pellet має API для спільної роботи з інструментом Jena.

Генерація єдиного запиту на семантизовані дані. Розглянемо генерацію первинного запиту на дані, сформульованого користувачем. Для генерування семантичних запитів використовуються такі підходи:

1. Користувач самостійно вводить запит на мові запитів SPARQL.
2. Запит конструюється за допомогою візуальних засобів, головною функцією яких є надання інтерфейсу для побудови семантичних графів-трійок «суб'єкт-предикат-об'єкт» і обмежень на них.
3. Користувач вводить запит природною мовою, а система надає засоби його розпізнавання і переведення в формат SPARQL.
4. Пошук за ключовим словом і його розвиток – інкрементарне конструювання запиту.

У запропонованій системі інтеграції використовується перший із цих підходів, але самостійне введення SPARQL-запиту до наявних джерел даних передбачає високий рівень компетентності користувача. Тому додатково розроблений наступний композитний алгоритм:

Крок 1. Користувач вибирає сутності і обмеження на них. Щоб уточнити цей крок, введемо необхідні позначення. Нехай $A = \{A_1,\ldots,A_m\}$ – множина доступних в системі сутностей в рамках глобального простору імен, $B = \{B_1,\ldots ,B_m\}$ – множина обмежень на діапазони даних в A_i, i=1,…,m, B_i, i=1,…,m, може бути порожнім, m — кількість доступних в системі сутностей в рамках глобального простору імен, $C \subset A$ — підмножина сутностей із множини A, яка визначається користувачем в процесі формування запиту і уточнює сутності, інформація про які має бути результатом пошуку.

Також можуть бути введені додаткові множини D, G, E (…) параметрів операторів опрацювання вибірки даних, наприклад LIMIT, ORDER BY, OFFSET і т.д.

Тоді запит на дані можна подати у вигляді кортежу:

<A[],B[],C[], *опціональні параметри*>

На практиці такий опис може бути реалізованим у вигляді веб-форми з вибором відповідних даних, а отримана із цієї форми інформація очевидним чином конвертується в SPARQL-запит, оскільки сутності кортежу є одночасно і сутностями цього запиту.

Крок 2. Оброблювач запитів отримує на вході запит, згенерований на кроці 1, і аналізує кількість джерел даних, в яких є інформація про потрібні сутності. Якщо таких джерел даних декілька, то користувачу пропонується уточнити запит, вибравши відповідне джерело даних. У цьому випадку користувач отримує стислу інформацію щодо кожного із можливих джерел даних для цього запиту. Ця інформація може витягуватися безпосередньо із джерела даних або із OWL-файлу зі стандартизованим форматом метаданих. Якщо користувач не може уточнити джерело даних, запит буде виконаний кожним можливим джерелом даних.

Моделі поділу запиту на розподілені дані на підзапити і його виконання

Саме ці моделі дозволяють побудувати ефективні алгоритми оброблення запитів до розподілених джерел даних, які, у свою чергу, визначають ефективність інтеграції джерел даних. За умови перетворення всіх імен даних у запиті у відповідності з глобальним простором імен для опрацювання єдиного запиту до розподілених джерел даних будемо використовувати описані в праці [4] рішення, які вже стали класичними.

Розроблені як частина проекту IBM Starburst вони закріплюють за оброблювачем, який отримав запит на вході три основних операції: аналіз і трансформацію запиту у план його виконання на основі наявної інформації щодо джерела (джерел) даних; багатокрокову оптимізацію запиту; виконання плану, щоб отримати результати запиту. Розглянемо моделі для кожної з наведених операцій.

Запропонований підхід до виконання запитів до розподілених джерел семантизованих даних узагальнює напрацьовані раніше рішення. Для виконання запитів до розподілених RDF- джерел даних, які отримують на вході SPARQL-запити, найбільш прийнятна система інтеграції D2RQ [29] і моделі, наведені у праці [32]. Система інтеграції D2RQ пропонує єдиний інтерфейс для опитування множини розподілених SPARQL-кінцевих точок і здійснює федерування запиту прозоро для клієнта. Виконувана D2RQ оптимізація SPARQL-запитів, побудована на статистичній інформації щодо кількості триплетів у джерелі даних, також досить ефективна. Але D2RQ погано документований і не здатний самостійно (без участі адміністратора) завантажувати інформацію щодо типів даних (дескрипторів) джерела, які мають специфічний формат.

Тому в запропонованій системі інтеграції впроваджені моделі федерування запитів до розподілених джерел даних, деякі алгоритми якої мають відмінності від розроблених для системи D2RQ:

1. Застосування як інформації щодо типів даних для кожного з підключених джерел даних файлів OWL і N3, які також застосовуються для уточнення семантики предметної області і додавання метаданих.

2. Забезпечення можливості виконання запитів не тільки на SPARQL-кінцевих точках, а й із застосуванням інших засобів комунікації, насамперед, обміну повідомленнями JMS або FIPA ACL.

3. Функціонування в умовах відсутності можливості оптимізувати запит, коли відповідні статистичні дані з віддалених джерел не доступні або змінилися.

4. Забезпечення можливості задавати цілий набір джерел даних, на які будуть направлятися згенеровані запити, що уможливлюється запропонованим алгоритмом генерації запиту, коли користувач уточнив доступні для виконання запиту джерела даних.

Федерування запиту на семантизовані дані. Розпочнемо з описання даних й адаптуємо класичну модель поділу, запропоновану в D2RQ. Ємністю джерела даних D є множина C_D кортежів $\boldsymbol{c} = (\boldsymbol{p}, \boldsymbol{r}) \in \boldsymbol{C_D}$, де p – наявний в D предикат, r – обмеження на суб'єкти і об'єкти. Ця зв'язка є регулярним filter-виразом мови SPARQL, яка робить вибір джерела даних більш точним, наприклад, можна уточнити, що джерело даних зберігає інформацію тільки про специфічні типи ресурсів. Позначимо обмеження r як функцію *r(subject, object)* з $\boldsymbol{r}: (\boldsymbol{RDF_T} \cup \boldsymbol{V}) \times (\boldsymbol{RDF_T} \cup \boldsymbol{V}) \to \{\boldsymbol{true}, \boldsymbol{false}\}$.

Застосовувати такі обмеження можна, наприклад, у тому випадку, коли в одному джерелі даних зберігаються рядкові дані, які починаються з букв від A до R, тоді як в іншому розташовані дані, які починаються з букв від Q до Z.

Обмеження на шаблони доступу. Деякі джерела даних мають обмеження на шаблони доступу. Наприклад, джерело даних може вимагати, щоб в запиті завжди були наведені назва міста, в якому виконується пошук, або його поштовий індекс.

Застосуємо рішення, запропоноване в [29]. Оскільки предикати повинні бути обмеженими, вони використані як основа для складання шаблону. Предикат не може бути змінною. Нехай L_D становить множину обмежень на шаблони доступу для джерела даних D і $(S, O) \in L_D$ буде одним шаблоном з S і O, які повинні мати зв'язані суб'єкти (S) або зв'язані об'єкти (O).

Джерело D може відповідати на запит з графовим шаблоном P, якщо він задовольняє щонайменше один із позначених шаблонів доступу для D. Нехай *bound(x)* буде функцією, яка повертає *false*, якщо x є змінною, і *true* в іншому випадку. Тоді шаблон доступу *(S, O)* задовольняєсь, якщо

$$\left(\forall p_s \in S \backslash 0: \exists (s, p_s, o) \in P: bound(s)\right) \wedge \left(\forall p_o \in O \backslash \exists: (s, p_o, o) \in P: bound(o)\right) \wedge (\forall p_b \in S \cap O: \exists (s, p_b, o) \in P: bound(s) \wedge bound(o))$$

Статистична інформація. Наявність статистичної інформації щодо доступних даних допомагає оптимізатору запитів знайти план виконання запиту, ефективний за витратами. Але враховуючи незалежність і розподіленість джерел даних, збір статистичної інформації не завжди можливий.

Наповнення каталогу інформацією. На центральному вузлі системи (оброблювачі вхідних запитів) є каталог, який зберігає інформацію щодо всіх підключених сервісів.

Первинна інформація щодо даних у кожному джерелі вибирається із його опису – відповідних файлів N3 і OWL, які також використовуються для додавання семантичного забарвлення даним.

Планування виконання запиту. При формуванні запиту до розподілених джерел даних необхідно вирішити, яке із джерел здатне на цей запит відповісти. Процеси знаходження релевантних джерел і генерування виконуваних підзапитів і складають сутність планування запитів. Планування запитів побудоване на інформації, отриманій із Сервіс-дескриптора. Нехай $\boldsymbol{R} = \{(\boldsymbol{d_1}, \boldsymbol{C_1}), \dots, (\boldsymbol{d_n}, \boldsymbol{C_n})\}$, тут $\boldsymbol{d_1}..\boldsymbol{d_n}$ - множина джерел даних і $\boldsymbol{C_1}..\boldsymbol{C_n}$ будуть їх вмістом, де $\boldsymbol{C_i} = \{(\boldsymbol{p_{i,1}}, \boldsymbol{r_{i,1}})..(\boldsymbol{p_{i,m}}, \boldsymbol{r_{i,m}})\}$.

Також на вибір доступних джерел даних впливає можлива наявність специфікованих користувачем додаткових обмежень при інкрементарному генеруванні запиту – на яких саме джерелах даних необхідно виконувати запит.

Вибір джерела даних. Запит на мові SPARQL містить не менше одного фільтрованих базових графових шаблонів (FBGP), кожний із яких містить шаблони триплетів. Планування запиту здійснюється окремо для кожного FBGP. Алгоритм пошуку релевантних джерел даних для запиту базується на простому порів-

нянні на відповідність шаблону триплета з доступними шаблонами триплетів в джерелах даних. Відповідність визначається рівністю предиката в шаблоні триплета предикату в джерелі даних, а також відповідністю обмежень на суб'єкт і об'єкт.

Нехай BGP становить множину шаблонів триплетів в FBGP. Результатом вибору джерела даних буде множина джерел даних D_j для кожного шаблону триплетів $t_j = (s_j, p_j, o_j) \in BGP$, де $D_j = \{d | (d, C) \in R \wedge \exists (p_j, r) \in C: r(s_j, o_j) = true\}$

Побудова підзапитів. Результати вибору джерел використовуються для побудови підзапитів, на які кожне джерело може дати відповідь. Підзапити повинні складатися з одного FBGB на одне джерело даних. Подамо підзапит як триплет (T, C, d), де T – множина шаблонів триплетів, C – множина обмежень значень, d – джерело даних, яке може відповісти на підзапит.

Оптимізація запитів. За результатами планування запиту план виконання запиту складається з декількох підзапитів. Задача оптимізатора запиту полягає в тому, щоб побудувати виконуваний і ефективний план виконання запиту, який враховує обмеження на шаблони доступу. Для побудови таких запитів використовується логічна і фізична оптимізація запиту.

Логічна оптимізація. Ця оптимізація запиту використовує еквівалентність виразів запиту для перетворення логічного плану запиту в еквівалентний план, який буде виконаний швидше або з меншими витратами. Рішення, запропоноване D2RQ, здійснює логічну оптимізацію двома шляхами. Спочатку, використовуються правила, описані у праці [21] для переписування оригінального запиту перед плануванням запиту таким чином, щоб об'єднати BGP там, де це можливо і де це можливо змінні заміняються константами із виразів з фільтрами.

Потім можливі обмеження значень переміщуються в підзапити для зменшення кількості проміжних результатів настільки рано, наскільки це можливо. нехай $\mathrm{Q} = (\mathrm{T}, \mathrm{C}, \mathrm{d})$ буде підзапитом і $\mathrm{FBGP} = (\mathrm{T}', \mathrm{C}')$ – фільтрований базовий графовий шаблон. Обмеження значень C' може бути переміщене в під запит, якщо всі змінні цього обмеження також використовуються в шаблонах триплета в підзапиті. Фільтри, які містять змінні із більш ніж одного підзапиту і які не можуть бути розділені з використанням обмеженої множини правил, повинні виконуватися локально в системі федеративних запитів.

Фізична оптимізація. Ця оптимізація запиту призначена для пошуку «кращого» плану виконання запиту серед всіх доступних планів і використовує модель вартості для порівняння різних планів. Вартість запиту визначається на основі інформації щодо кількості доступних в кожному джерелі даних.

Виконання запиту. На цьому етапі можна широко застосовувати багатопоточність: скільки буде підзапитів до різних джерел даних, стільки буде виділено багатопоточних процесів. Розпаралелювання забезпечується технологіями Java Multithreading.

Об'єднання результатів запиту. Тут можна використовувати відомі результати. Дещо окремим питанням залишається реалізація *Nested Loop Join*.

Але навіть виконання цього етапу користувачем власними засобами буде виправданим, оскільки доцільність об'єднання результатів запиту не завжди легко визначити, оскільки дані дуже різнорідні.

Висновки

Запропоновано підхід до інтеграції інформаційних ресурсів, який можна застосовувати для взаємодії СЦД. Цей підхід дозволяє структурувати джерела даних і встановлювати зв'язки між ними, не вносячи змін до самих даних, структур їх зберігання та механізми супроводу. Запропонована багаторівнева архітектура системи інформаційного обслуговування користувачів за їх запитами забезпечує доступ до різнорідних джерел даних і представлення результатів у зручному вигляді. Інформація про джерела даних зберігається в онтологічних специфікаціях, з їх же допомогою описано зв'язки між властивостями джерел даних.

Рішення базується на інтеграції параметрів джерел даних з семантикою предметної області і є легко розширюваним шляхом додавання нових семантичних специфікацій і параметрів підключення джерел даних. Розроблені специфікації, що описують семантику джерел даних, можуть бути застосовані для розв'язання й інших наукових задач.

Перелік посилань

1. M. Zgurovsky, A. Gvishiani, K. Yefremov, A. Pasichny. Integration of the Ukrainian science into the world data system // Cybernetics and Systems Analysis: Volume 46, Issue 2 (2010). – P. 211.

2. Теленик С.Ф. Логічний підхід до інтеграції програмних застосувань підтримки міждисциплінарних наукових досліджень / С.Ф.Теленик, О.А.Амонс, К.В.Єфремов, В.Т.Лиско // Наукові вісті НТУУ «КПІ». – 2013. – № 5. – С. 53-72.
3. Guide To The World Data Center System - General Principles, World Data Centers, Data Services [Електронний ресурс]. Режим доступу: http://www.wdc.rl.ac.uk/wdc/guide/wdcguide.pdf.
4. Kossmann D. The state of the art in distributed query processing. – New York: ACM Computing Surveys vol.32, «ACM», 2000. – P. 422-469.
5. Тузовский А.Ф. Интеграция баз данных на основе онтологий [Електронний ресурс]. Режим доступу: ontology.ipi.ac.ru/files/d/dc/DBIntegration.ppt — Назва з екрану.
6. Имхофф К. Аналитические решения: понимание трех составляющих интеграции - EAI, EII, ETL (перевод с английского) [Електронний ресурс]. Режим доступу: http://citforum.edunet.kz/consulting/BI/integration/.
7. Wu T. EII – ETL – EAI - What, Why, and How! [Електронний ресурс]. Режим доступу: https://www6.software.ibm.com/developerworks/- tw/events/20051028/db2_6b.pdf.
8. Calvanese D., De Giacomo G., Lembo D., Lenzerini M., Rosati R. Data Management in Peer-to-Peer Data Integration Systems [Електронний ресурс]. Режим доступу: http://www.dis.uniroma1.it/~degiacom /papers/2006/calv-etal-GDM-book-06.pdf.
9. Кузнецов С.Д. Основы современных баз данных [Електронний ресурс]. Режим доступу: http://www.citforum.ru/database/osbd/- contents.shtml.
10. Интеграция данных: синтаксис и семантика / Л.Черняк // Открытые системы. – 2009. –№10. [Електронний ресурс]. Режим доступу: http://www.osp.ru/os/2009/10/ 11170978/.
11. Розробка онтології матеріалознавства засобами Protégé-OWL / Д.Г. Досин, Р.Р. Даревич, Н.В. Шкутяк // Штучний інтелект. – 2008. – № 3. – с. 70-77.
12. Д.И. Муромцев. Онтологический инжиниринг знаний в системе Protégé. – СПб: СПбГУ ИТМО, 2007. – 62 с.
13. Шаповалова С.И., Ефремов К.В., Глуханик А.И. Организация интегрированного доступа к информационным ресурсам [Текст] / С.И. Шаповалова, К.В. Ефремов, А.И. Глуханик / – Сборник трудов конференции ИАИ-2011 – 2011. – С.102-108.
14. Пол Спенсер. XML. Проектирование и реализация. М.: Лори. – 2001.– 510 с.
15. OWL Web Ontology Language W3C Recommendation [Електронний ресурс]. Режим доступу: http://www.w3.org/TR/owl-features/.
16. I. Horrocks et al. From SHIQ and RDF to OWL: the making of a Web Ontology Language / Web Semantics: Science, Services and Agents on the World Wide Web 1 (2003), p. 7–26.
17. RDF Primer W3C Recommendation [Електронний ресурс]. Режим доступу: http:// www.w3.org/TR/rdf-primer/.
18. Жигалов В. А. Технология построения естественно-языковых интерфейсов к структурированным источникам данных – М.: Диссертация на соискание ученой степени кандидата технических наук, 2000. – 172 с.
19. SPARQL Query Language for RDF [Електронний ресурс]. Режим доступу: http://www.w3.org/TR/rdf-sparql-query/.
20. ARQ - A SPARQL Processor for Jena [Електронний ресурс]. Режим доступу: http://jena.sourceforge.net/ARQ/.
21. Quilitz B., Leser U. Querying Distributed RDF Data Sources with SPARQL – Berlin: Lecture Notes in Computer Science vol. 5021, «Springer Berlin Heidelberg», 2008., – с. 524-538.
22. Langegger A., Wöß W., Blöchl M. SemWIQ – Semantic Web Integrator and Query Engine, – München: Informatik 2008 Beherrschbare Systeme dank Informatik, 2008. – с. 718-722.
23. Kikuchi S., Sachdeva S., Bhalla S. и др. Adaptive Integration of Distributed Semantic Web Data – Berlin: Databases in Networked Information Systems Volume: 5999, «Springer Berlin Heidelberg», 2010. с. 174-193.
24. A. Schwarte, P. Haase, K. Hose, R. Schenkel, M. Schmidt. FedX: A Federation Layer for Distributed Query Processing on Linked Open Data. In ESWC Poster and Demo Session Proceedings. Springer, 2011.
25. http://www.pangaea.de/
26. http://www.esimo.ru/
27. Franklin M. Fpom Database to Dataspace: A New Abstraction for Information Management / M.Franklin, A.Halevy, D.Maier // ACM SIGMOD. – 2005. – December (Record 34, no. 4). Режим доступу: http://portal.acm.org/wdc/citation.cfm7id-1107502.
28. Шаховська Н.Б. Методи опрацювання консолідованих даних за допомогою просторів даних / Н.Б.Шаховська // Проблеми програмування. – 2011. – № 4. – С. 72-84.
29. The D2RQ Plattform: Accessing Relational Databases as Virtual RDF Graphs [Електронний ресурс]. Режим доступу: http://d2rq.org/.
30. Jena Toolkit For Semantic Web Applications [Електронний ресурс]. Режим доступу: http://jena.apache.org/.
31. Evren Sirin, Bijan Parsia and oth. Pellet: A practical OWL-DL reasoner — Web Semantics: Science, Services and Agents on the World Wide Web, Volume 5, Issue 2, June, 2007, P. 51-53.
32. Бездушный А.А. Математическая модель системы интеграции данных на основе онтологий // Вестн. НГУ: Сер. Информационные технологии. Новосибирск, 2008. Т. 6. Вып. 2. с. 15–40.

УДК 681.3, 658.115.6

ANNA KYSELOVA

CONTEXT DATA ANALYSIS FOR MICROGRID CONTROL SYSTEM

За останні кілька десятиліть попит на електроенергію зростає, тому в останні роки, швидко розвиваються системи керування в області мікрогрід, що включають в себе керування як поновлювальними джерелами енергії так електричними навантаженнями. Системи керування у мікрогрід повинні мати змогу опрацьовувати великі обсяги даних, що пов'язані як з енергетикою, так і бути у змозі реагувати на зовнішні зміни контекстних даних. У статті описується модуль аналізу контекстних даних, що входить до системи керування мікрогрід.

Over the past few decades the energy demand has been continuously increasing (both in industrialized and in emerging countries) and the control systems for renewable sources and electrical loads have become much more sophisticated. Through the last years we have seen increased interest in the area of Microgrid control system. Today they are highly distributed, can manage large amounts of energy related data and have to be able to react rapidly (but smartly) when conditions change. The paper describes these challenges and presents data management solutions as a module of context data analysis for Microgrid control system. These solutions include time series forecasting, managing subscriptions for prediction energy supply and demand and analytical query processing past and future context data.

1. Introduction

A microgrid comprises medium- and/or low-voltage distribution systems with distributed energy sources, storage devices and controllable loads. They can operate either if connected to the main power network or if isolated (islanded) in a controlled and coordinated way. Frequently we refer to a self-sufficient interconnection of distributed generation, residential and industrial load in a low-voltage network without a persistent connection to a larger grid [1, 2].

The main goal of the microgrid concept is to create a control system that will be able to prediction and react smartly to the actions of all electrical facilities (loads and generators) connected to it, by means of converters in the unified information environment. The system should be able to rationally utilize energy, efficiently control normal and emergency conditions and to take into account the user comfort. Microgrid control system as a complex information processing system integrates such organization levels as: Renewable sources of energy; Electrical devices; Digital sensors; User tasks [3].

Microgrid is tightly interconnected with sensor networking, allowing behavior prediction and fault detection. There must be a constant balance between user comfort and energy efficiency. Viewed as an energy system, i.e., one with energy input/output and internal energy flows, a microgrid control system presents an example of a deeply coupled system of energy usage, comfort and performed work. Currently, efforts are aimed at the improvement of energy efficiency to decrease energy costs [4].

The paper focuses on the development of the module of context data analysis for Microgrid control system, which is designed to minimize user's costs and ensure the most comfortable conditions in microgrid. It is engineered with the use of a new method which consists of context prediction obtained from sensors of Microgrid.

2. Model of context

Load and generators management in microgrid is connected with the difficulty of analyzing the current situation due to its continuous alteration and the necessity for additional information-processing equipment to process the huge amount of heterogeneous sources [5]. One of the artificial intelligence (AI) tools introducing handy and understandable model representation is the context model.

Context can been defined as any information that can be used to characterize the situation of an entity, where the entity is a person, place, or object that is considered relevant to the interaction between a user and an microgrid control system, including the user and application themselves [6, 7].

Integration into the context of the information received from heterogeneous sources allows obtaining the model of the real microgrid present state (Fig. 1).

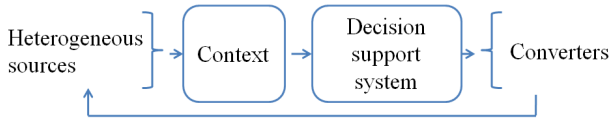


Fig. 1. Context conversion structure.

Context is formed as a result of an event generated by microgrid sensor data. Conversion of the measured data to context is shown in Table 1.

Table 1. Conversion of the Measured Data to Context

Context source (from sensors)	Raw data	Context
thermometer	00100111	14 °C
microcontroller (current sensor)	00110010	2 A

Context attributes of microgrid:

- Physical (Lighting, temperature, acceleration etc.);
- Computing entity (connectivity, network bandwidth, network latency etc.);
- Equipment (loads, energy sours, sensors etc.);
- Time of synchronization (time of day, weekday, month, season of year etc.);
- Action at previous action time (get, set, subscribe, unsubscribe);
- Architecture (zone or location).

Context prediction requires the consideration of the time dimension. A set of observations $t_1 ... t_n$ with t_1 being recorded at a specific time interval t_i is called a time series [9]. A discrete time series is one in which the observations t_1 are taken at discrete time intervals. Despite a sample quantity of publications about time series processing methods, the problem of practical application of these methods under the conditions of dynamic operational conditions of the converters is actual.

3. An approach to context parameter prediction

Predictions require the specification of a stochastic model that captures the dependency of future or past values. We will refer to such models as the prediction models. Usually the creation of the prediction models is computationally expensive and often involves numerical optimization schemes to estimate model parameters. Once the model has been created and the parameters estimated, it can efficiently be used over and over again to predict future values of the time series. As the new data arrives, the prediction model might require maintenance in form of parameter re-estimation. This is computationally expensive while most parameters cannot maintain constant step-size [10].

The following algorithm is provided for module of context data analysis. It's peculiarity lies in an improved performance and additional functionality inside the input parameters of the time series database.

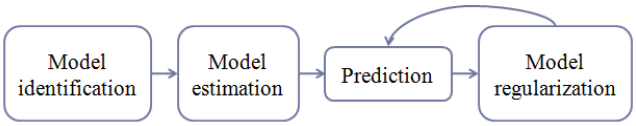


Fig. 2. Formalized procedural path of the prediction process.

Model Identification.

Preliminary time series processing consists in the detection of the series values anomalous values and series smoothing. The randomness of the commutation, though, leads to the disturbances in power consumption characteristics [11, 12]. Keeping record of time points and the value of the disturbances complicates the forecasting process and can lead to erroneous results. Filtration or smoothing of context time series is the necessary preliminary prediction stage for obtain trends [13, 14]. Thus, the first step of the module of context data analysis is the filtration and the second step is the prediction.

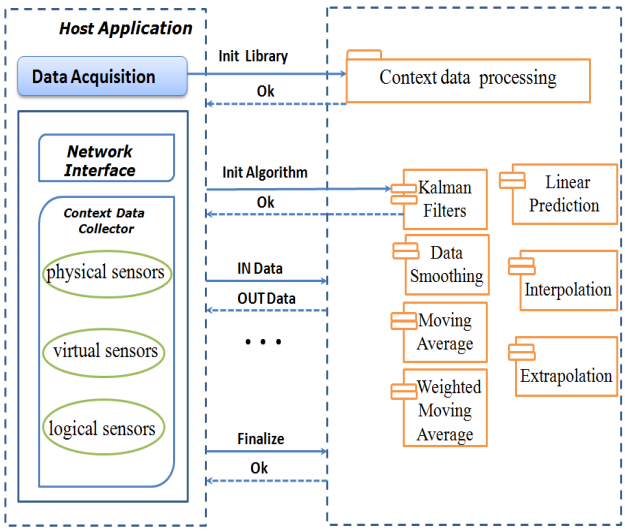


Fig. 3. Raw sensor context data analysis.

There are three distinct groups of smoothing:

- Averaging Methods – moving average, weighted moving average;

- Exponential Smoothing Methods – simple, weighted, exponential, double;
- Kalman filter.
- And three group of prediction:
- Interpolation – linear, polynomial, spline;
- Extrapolation – linear, polynomial, French curve, conic;
- Linear prediction.

Model Estimation.

The Table 2 consist of the lists initial parameters for prediction algorithms.

Table 2. Parameters for Prediction Algorithms

No.	Letter symbol	Description
1	γ	The number of series values to be forecasted
2	t_{max}	The right margin of the time series forecasting interval
3	n	Sample (a number of time series values, used for a single forecasting)
4	t_k	Initial time point of the forecasting
5	n_{min}	Minimal sample value
6	n_{max}	Maximal sample value

Forecasting of the time series without noise is carried out by the following algorithm (Fig.4).

1) Assign γ, n and n values;

2) Read the value of a current time series value X(tk). If $t_k = t_{max}$ - algorithm stops;

3) If $t_k<t_{max}$, for the current time series value at a time point tk the task of single forecasting on the interval γ is solved. Predicted time series value will be Y(tk+γ) as in (1)

$$E(t) = Y(t_k) - X(t_k + \gamma)\,. \qquad (1)$$

4) $Y(t_k+\gamma)$ is entered in the dynamic list of the time series values being predicted on the interval up to t_{max}.

5) If $t_k<t_{max}$, k=k+1 and the dynamic list of the time series values being predicted is checked for an element with a timestamp $X(t_k+\gamma)$. If there is such an element, where Q(tk+γ) is a root-mean-square error (RMSE) of a time series values being predicted from the real $X(t_k+\gamma)$ as in (2)

$$Q(t_k) = M\left[E^2(t)\right] = M\left[(Y(t_k) - X(t_k))^2\right]. \qquad (2)$$

Sample n regularizing is performed and the element $Y(t_k+\gamma)$ is deleted from the dynamic list of the time series values being predicted. If the list doesn`t contain an element with a timestamp tk, the algorithm goes to step 2).

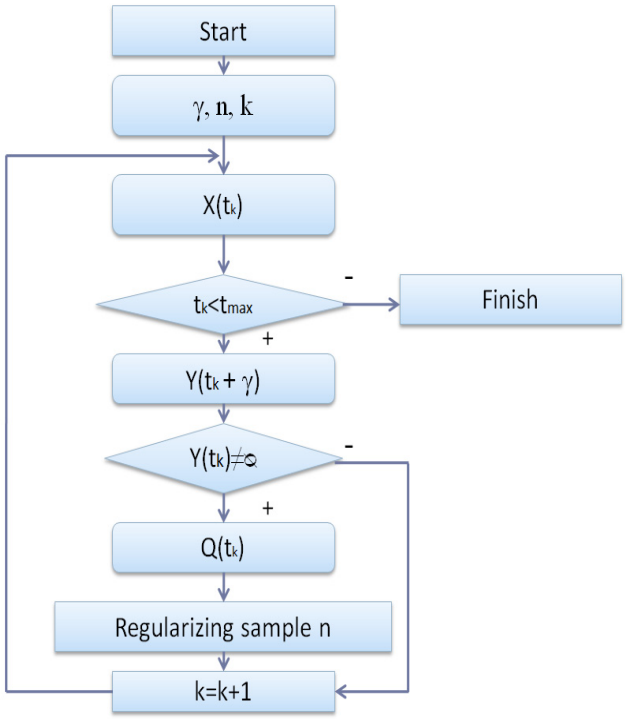


Fig. 4. General prediction algorithm.

The algorithm repeats itself until forecasting error value won`t be received for every time series value on the time interval up to t_{max}.

Model Regularization.

If the value $Q(t_k+\gamma)$ falls outside the confidence range of prediction errors, the task of regularizing sample *n* of the prediction method is performed. By sample regularizing we understand sample value alteration up to the value which provides the transition of $Q(t_k+\gamma)$ to the area of confidence range (Fig.5).

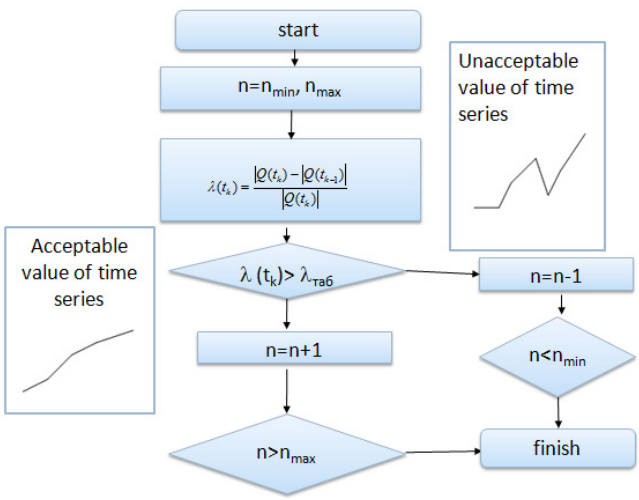


Fig. 5. Regularization algorithm.

1) Assign n= n_{min} , n_{max}=max;

2) Read the RMSE $Q(t_k+\gamma)$ from dynamic list of time series being forecasting $Q(t_{k-1}+\gamma)$. The find unacceptable values of time series we used Irwin criterion as in (3):

$$\lambda(t_k+\gamma)=\frac{|Q(t_k+\gamma)-|Q(t_{k-1}+\gamma)|}{|Q(t_k+\gamma)|}. \quad (3)$$

3) If $n < n_{max}$, then n=n+1 and algorithm goes to step 2).

This prediction approach with regularizing sample, it is an important step in enabling the processing of larger amounts of heterogeneous data from distributed and renewable energy sources into the Microgrid.

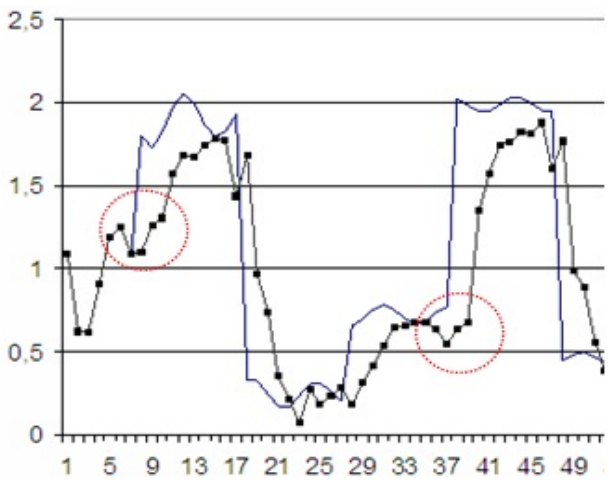


Fig. 6. a) Linear prediction. The RMSE of the time series without model regularization.

Figure 6 (b) shows an example of the result of the proposed module regularization depending on the values of the time series (acceptable or unacceptable values). Based on the optimization of the sample size, changes the order of the polynomial of the prediction time series method.

After module of context data analysis, control algorithms can generate a variety of control decisions coming on the microgrid converter assembly. Therefore, microgrid control system is a smart decision support system, which adjusts to the context alteration by means of its operation algorithm alteration.

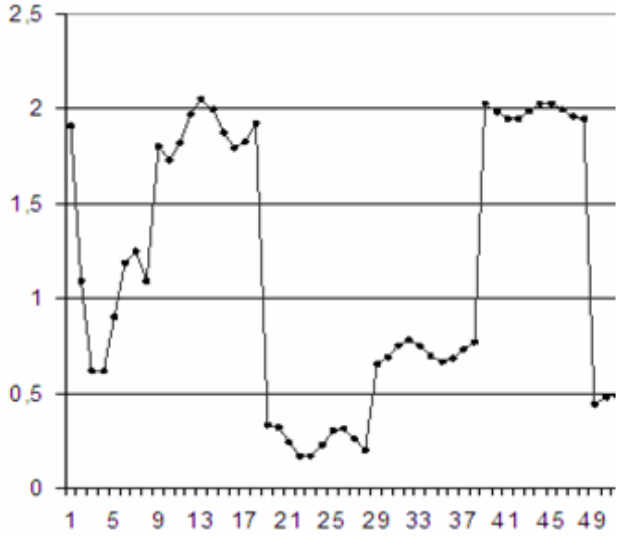


Fig. 6. b) Linear prediction. The RMSE of the time series with model regularization.

4. Conclusion

The integration of prediction inside a Microgrid control system is a rising topic in the research community. We have described the module of context data analysis for Microgrid control system that facilitates a more efficient utilization of renewable energy sources by taking benefits of energy flexibilities.

The proposed module regularization (adaptation) of time series for prediction method allows reducing forecasting error from 6-5% to 2-1.5%, as the test results showed. Program application approach of regularization its initial version so that there is still the room for improvement. Firstly, the current approach can be extended to support other forecasting methods (e.g., ARMA, ARIMA). Secondly, it can increase the accuracy of future data forecasts. Finally, we obtain faster approximate results of historical data and increased performance of data loading.

References

1. B. Kroposki, T. Basso, and R. DeBlasio, "Microgrid Standards and Technologies," in Power and Energy Society General Meeting – Conversion and Delivery of Electrical Energy in the 21st Century, Pittsburgh, PA, 2008, pp. 1-4.
2. Mohamed, "Microgrid Modeling and Online Management," PhD thesis, Helsinki University of Technology, Helsinki, Finland, 2008.
3. Dicorato, M., Forte, G., and Trovato, M., A procedure for evaluating technical and economic feasibility issues of MicroGrids, IEEE Bucharest Power Tech Conference, 2009.

4. C. Lima, V. Gomes, J. Lima, J. Martins, J. Barata, L. Ribeiro, and G. Candido, “A standard-based software infrastructure to support energy efficiency using renewable energy sources,” in Industrial Electronics (ISIE), 2011 IEEE International Symposium on, Gda´nsk, Poland, june 2011, pp. 1175 –1180.
5. Pichkalov, I.; Kyselova, A.; Kyselova, O.; Kyselov, G., "Optimal control of uninterruptible power system with diesel generator", Electronics and Nanotechnology (ELNANO), 2013 IEEE XXXIII International Scientific Conference, pp.378-380, 16-19 April 2013.
6. Kyselova A., Shalaginov Intelligent model for Dealing with Contexts Application // CAD in Machinery Design. Implementation and Educational Problems: 18-th Ukranian–Polish Conference «CADMD’2010», October 14-16, 2010, Lviv, Ukraine: Proc. – Lviv: Vezha& Co , 2010. – P. 20–22.
7. Beale, R., Lonsdale, P. Mobile Context-aware Systems: The Intelligence to Support Tasks and Effectively Utillise Resources. MobileHCI (2004).
8. Zhuikov, V.; Kyselova, A., "Integration of context-aware control system in microgrid", Electronics and Nanotechnology (ELNANO), 2013 IEEE XXXIII International Scientific Conference, pp.386-390, 16-19 April 2013.
9. Firat A. Information Integration Using Contextual Knowledge and Ontology Merging [Electronic resource]: PhD Thesis: 27.08.03: defended Sep. 2003. MIT, 2003. 151p. URL: http://www.mit.edu/~bgrosof/paps/phd-thesis-aykut-firat.pdf (access date: 29.05.2009).
10. G. Box, G. Jenkins, and G. Reinsel. Time Series Analysis: Forecasting and Control. Wiley, 2008.
11. R. J. Hyndman, A. B. Koehler, R. D. Snyder, and S. Grose. A state space framework for automatic forecasting using exponential smoothing methods. International Journal of Forecasting, 18, 2000.
12. Zellner and J. Tobias. A note on aggregation, disaggregation and forecasting performance. Journalof Forecasting, 19:457{469, 2000.
13. Kyselova A.G., Kyselov G.D., Sergeev A.A., Shaloginov “Data processing for multimodal application” “Electronics and Communications”, №2, 2011, pp. 86-92.
14. S. Duan and S. Babu. Processing forecasting queries. In VLDB, 2007.

УДК 004.932.72'1

ДЬЯКОНОВА С.В.,
ЗАЙЧЕНКО Ю.П.

ПОДХОД К РЕШЕНИЮ ЗАДАЧИ АВТОМАТИЗИРОВАННОГО ОБНАРУЖЕНИЯ ЗДАНИЙ НА СПУТНИКОВЫХ ИЗОБРАЖЕНИЯХ

В статье рассматривается разработанный подход к обнаружению зданий на спутниковых изображениях. Описывается модификация структуры нечеткого классификатора NEFCLASS, позволяющая определить степень соответствия входного образца выходным классам. Рассматривается информативный признак, основанный на цветовой модели HSV и использующий показатель насыщенности цвета для характеристики объектов спутниковых изображений.

In the article discusses an approach to building extraction from high-resolution satellite images. It considers a modification of the fuzzy classifier NEFCLASS structure, which allows determining the degree of compliance of input sample and outputs. It proposes informative patterns, based on the HSV color model, and using the color saturation for characteristics of objects in satellite imageries.

Введение

Задача автоматического распознавания зданий на спутниковых изображениях сверхвысокого разрешения является важной частью в решении задачи автоматической интерпретации данных, получаемых из систем дистанционного зондирования Земли (ДЗЗ). Для сбора, хранения, анализа и графической визуализации пространственной информации используется ГИС. Ручная оцифровка и ввод пространственных данных в базу данных ГИС имеет недостатки в виде больших временных и материальных затрат, а также в виде высокого риска допущения ошибок, по причине человеческого фактора. Автоматическая система распознавания зданий на спутниковых изображениях позволит не только сократить временные и материальные затраты на обновление базы данных ГИС, а также повысить точность вводимых данных.

За последние годы количество исследований, касательно темы обнаружения зданий на спутниковых изображениях, значительно выросло. Большую часть работ можно классифицировать относительно того требуется ли для них фаза обучения или нет. Наиболее распространенными подходами являются: линейные классификаторы [1], методы анализа с использованием множества самоорганизующихся карт [2], алгоритмы на основе искусственных нейронных сетей [3], подходы на основе геометрических признаков [4] [5].

Настоящая работа построена на принципах объектно-ориентированного подхода к анализу спутниковых изображений, в рамках которого происходит анализ и классификация изображений на уровне объектов, т.е. групп пикселей, объединенных на основе определенной совокупности критериев. В процедуре объектно-ориентированного анализа можно выделить два основных этапа: первый этап направлен на выделение объектов, т.е. их сегментацию, второй этап - на распознавание полученных сегментов.

Для решения задачи распознавания зданий на спутниковых снимках используется направление интеллектуального анализа данных, объединившее в себе нечеткую логику и нейронные сети под названием «мягкие вычисления» (Soft Computing).

В качестве основы для метода распознавания зданий был выбран нечеткий классификатор NEFCLASS, поскольку поставленная задача решается в условиях неопределенности. Источниками неопределенности могут служить неполнота знаний об исследуемых объектах, неполнота обучающей выборки, сильная зашумленность образцов. Для обучения параметров функций принадлежности были выбраны три различных алгоритма обучения: градиентный, сопряженных градиентов и генетический.

Целью статьи является описание разработанного подхода на основе технологии мягких вычислений к обнаружению зданий на спутниковых изображениях, позволяющего повысить эффективность извлечения пространственных данных из спутниковых изображений.

Применение нечеткого классификатора NEFCLASS к задаче распознавания зданий

В качестве основы для метода распознавания зданий был выбран нечеткий классификатор NEFCLASS [6], поскольку поставленная задача

решается в условиях неопределенности. Источниками неопределенности могут служить неполнота знаний об исследуемых объектах, неполнота обучающей выборки, сильная зашумленность образцов.

Здания были разделены на три основных вида: здания с треугольной, плоской и круглой крышами, поскольку каждый из перечисленных видов зданий имеет различные геометрические признаки. Такое разделение дает неоспоримое преимущество при создании наиболее точной и полной базы правил для классификатора. Таким образом разделение одного большого класса «здание» на три непересекающихся подкласса «здание с треугольной крышей», «здание с круглой крышей» и «здание с плоской крышей» позволит получить наилучший результат при распознавании.

Одним из результатов зрительного восприятия человеком какого-либо изображения является глобальная оценка свойств этого изображения. В [7] таким свойствам дается название «интегральных». Набор значений интегральных характеристик объекта отражает его индивидуальность. Следовательно, для формирования базы знаний нечеткого классификатора NEFCLASS необходимо определить систему интегральных характеристик объектов (информативных признаков), принадлежащих каждому рассматриваемому классу. В случае нечеткой нейронной сети характеристики объектов задаются в виде нечетких множеств, которые описывают классы образов.

Система геометрических особенностей различных видов зданий позволяет выразить сходства и различия между ними. Множество геометрических особенностей образует следующий набор признаков:

1. Округлость крыши здания для каждого сегмента рассчитать, как отношение площади к периметру. Площадь сегмента рассчитать, как количество принадлежащих сегменту пикселей, а Периметр сегмента рассчитать, как количество пикселей лежащих на границе области.

2. Среднее значение углов определить по следующей схеме [1]. Проводится главная ось между двумя точками контура, которые находятся на максимальном расстоянии друг от друга. После определяются две вспомогательные оси с каждой стороны главной оси, для этого построить перпендикулярные линии к главной оси, которые имеют максимальное расстояние от контура до главной оси. Провести поперечную ось, это должен быть перпендикуляр к главной оси, который соединяет две точки контура находящиеся на максимальном расстоянии друг от друга. Соединить линиями полученные шесть точек в шестиугольник. В качестве признака возьмем среднее значение двух углов, лежащих по обе стороны главной оси.

3. Длина крыши здания определяется как отношение длины главной оси к длине поперечной оси, полученного выше шестиугольника.

4. Полнота вычисляется как отношение общего количества пикселей, которые лежат внутри шестиугольника и не принадлежат к исследуемой области к количеству пикселей, которые лежат за пределами шестиугольника и принадлежат исследуемому сегменту.

Применение цветовой модели HSV для определения интенсивности оттенка крыш зданий

Предлагаемый в работе признак для определения оттенка крыш использует понятие цветовой модели HSV, в которой координатами цвета являются:

Цветовой тон (Hue). Тон определяется характером распределения излучения в спектре видимого света. Именно тон определяет название цвета. Варьируется в пределах от 0° до 360°.

Насыщенность (Saturation). Может варьироваться в пределах от 0 до 1. Чем больше этот параметр, тем чище цвет.

Значение цвета (Value) или Яркость (Brightness). Задаётся в пределах от 0 до 1.

Опытным путем была найдена закономерность показывающая, что насыщенность цвета крыш зданий имеющих цветовой тон в пределах от 60° до 280° значительно ниже, чем насыщенность других объектов, например водных объектов, лесных зон, которые также находятся в этом пределе цветового тона. Разработанный информативный признак представляет собой отношение среднего значения цветового тона изображения к среднему значению насыщенности его цвета. Параметр яркости используется для того, чтобы отсеять слишком темные и светлые пиксели на крыше (например, белый и черный цвет) и не включать их для подсчета среднего значения. Таким образом, случайные тени на крыше от соседних зданий или от растущих рядом деревьев не создадут помех при подсчете признака. Предложенный признак позволяет распознавать здания с любым цветом крыши, и может отличить прямоугольное здание с зеленой крышей или круглой здание с синей крышей от схо-

жих по геометрической форме объектов, имеющих растительную или водную поверхность.

На Рис.1 представлен пример нахождения значений информативных признаков.

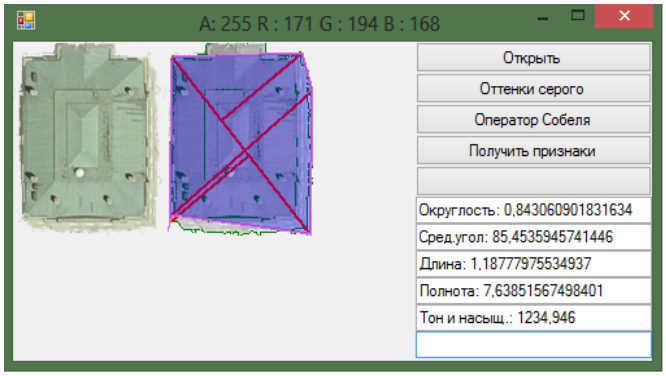


Рис. 1 Пример нахождения значений информативных признаков

Предлагается модификация структуры нечеткого классификатора NEFCLASS, позволяющая определять степень соответствия входного образца каждому из выходных классов и в случае, если отклонение между значениями меньшеε, то ранжировать возможные решения по предпочтительности. В существующей системе выходной класс определяется как максимальное значение из всех значений функций активации нейронов выходного слоя, т.е. берется самый сильный сигнал. Преимущество предложенной модификации заключается в возможности нахождения приоритетных классов, на Рис. 2 представлена архитектура модифицированной ННС NEFCLASS.

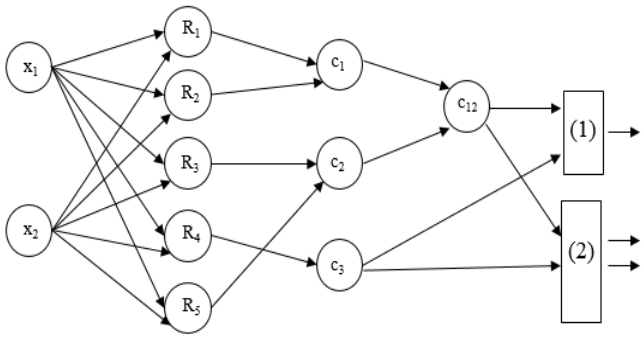


Рис.2 Архитектура модифицированной ННС NEFCLASS

Активация нейронов правил:

$$a_R^{(p)} = \min_{x \in U_1} \{W(x,R)(a_x^{(p)})\}$$

Активация нейронов выходного слоя:

$$a_c^{(p)} = \sum_{R \in U_2} W(R,c) a_R^{(p)}$$

$x = (x_1, \ldots, x_n) \in$ классу c_m, если

$$a_{c_m}^{(p)}(x) = \max_i a_{c_i}^{(p)}(x) \qquad (1)$$

Степень соответствия P_{c_i} входного образца $x = (x_1, \ldots, x_n)$ классу c_i

$$P_{ci} = \frac{c_i}{\sum_{i=1}^{m} c_i}$$

Критерий: $\forall i, \forall j \quad |P_{ci} - P_{cj}| \le \varepsilon$ (2)

Экспериментальные исследования

В работе использовались спутниковые изображения IKONOS и GeoEye-1. Пространственное разрешение спутниковых данных составляет 0.8 м и 0.5 м соответственно. Входное изображение представляет собой растровые данные в цветовой модели RGB, которые преобразуются в оттенки серого с последующей сегментацией.

Выходам нейронной сети NEFCLASS соответствуют четыре класса $\{c_1, c_2, c_3, c_4\}$, входам – пять информативных признаками $\{x_1, x_2, x_3, x_4, x_5\}$

c_1 - здание с треугольной крышей;
c_2 - здание с круглой крышей;
c_3 - здание с плоской крышей;
c_4 - не здание.
x_1 – округлость крыши;
x_2 - среднее значение углов;
x_3 - длина крыши;
x_4 - полнота;
x_5 - среднее значение интенсивности оттенка крыши.

Рассмотрим способ формирования базы правил. Каждое правило имеет следующий вид:

ЕСЛИ x_1 является μ_1, …, x_n является μ_n,
ТО образец $(x_1, \ldots, x_5)$ принадлежит классу i;
где i название класса образа,
n количество информативных признаков.

Для обучения параметров функций принадлежности были выбраны три различных алгоритма обучения: градиентный, сопряженных градиентов и генетический [8]. Для обучения используется гауссова функция принадлежности.

Обучающая выборка состояла из 700 образцов, выборка для тестирования включала 400 образцов.

На Рис.3 показана зависимость погрешности от количества пройденных итераций по каждому методу обучения для нейронной сети NEFCLASS.

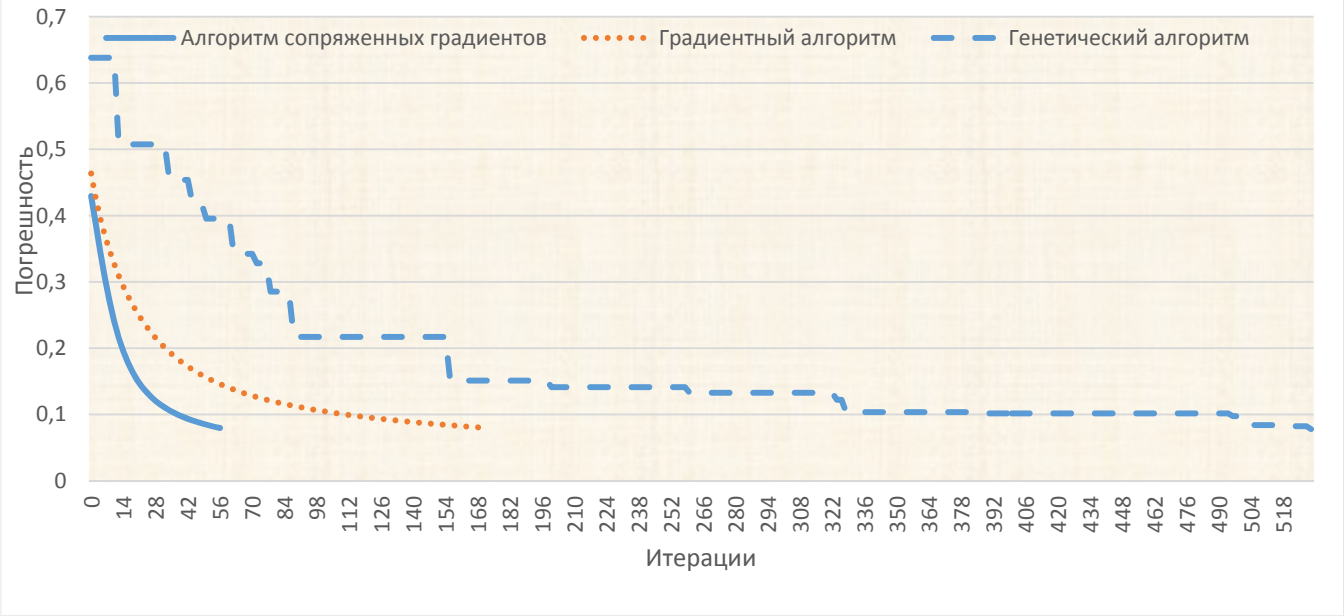


Рис.3 Зависимость погрешности от количества пройденных итераций для различных алгоритмов обучения

На Рис.4 приведены результаты классификации.

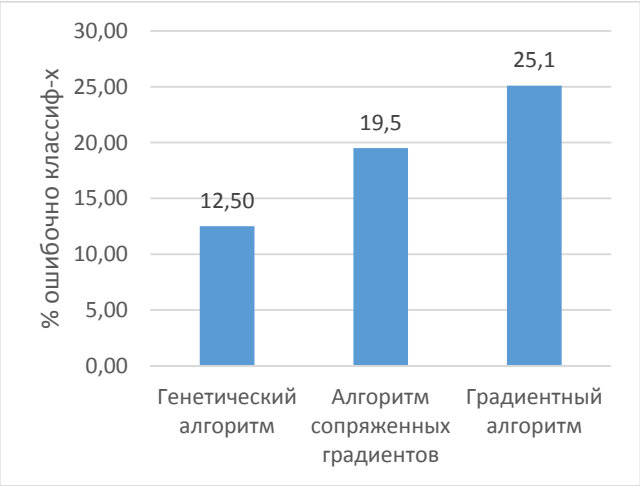


Рис.4 Процент ошибочно классифицированных образцов

По результатам экспериментов наилучшее качество распознавания было получено при обучении нейронной сети с помощью генетического алгоритма обучения. Процент ошибочно классифицированных образцов составил 12,50%.

Выводы

Описана модификация структуры нечеткого классификатора NEFCLASS, позволяющая определить степень соответствия входного образца выходным классам, что повышает эффективность распознавания и обеспечивает ранжирование возможных решений по предпочтительности. Предложен информативный признак, основанный на цветовой модели HSV и использующий показатель насыщенности цвета для характеристики объектов спутниковых изображений. Данный информативный признак, позволяет отличить верхнюю конструкцию зданий от теней и объектов, имеющих растительную и водную поверхность.

Построен модифицированный нечеткий классификатор NEFCLASS с тремя алгоритмами обучения: генетическим, градиентным и алгоритмом сопряженных градиентов. По результатам экспериментов наилучшее качество распознавания было получено при обучении нейронной сети с помощью генетического алгоритма обучения, 12,50% ошибочно классифицированных образцов.

Список литературы

1. S. M¨uller, D. W. Zaum, Robust building detection from aerial images. - IAPRS, Vol. XXXVI, Vienna, 2005. – pp. 143-148.
2. Persson M., Sandvall M., Duckett T. Automatic Building Detection from Aerial Images for Mobile Robot Mapping. – International Symposium on In Computational Intelligence in Robotics and Automation, 2005. - pp. 273 - 278.

3. Benediktsson J.A., Pesaresi M., Amason K. Classification and feature extraction for remote sensing images from urban areas based on morphological transformations. – Geoscience and Remote Sensing, 2003. - pp. 1940–1949.
4. Sirmacek B., Unsalan C. Building Detection from Aerial Images Using Invariant Color Features and Shadow Information. – International Symposium on Computer and Information Sciences, 2008. - pp. 1-5.
5. Wei Liu. Building detection from high-resolution satellite image using probability model. – IGARSS, 2005. - pp. 3888 - 3891.
6. Detlef Nauck, Rudolf Kruse. NEFCLASS – A neuro-fuzzy approach for the classification of data. – Applied Computing, 1995. – pp. 1-5.
7. В. А. Ганзен. Восприятие целостных объектов. - Л.: Издательство Ленинградского университета, 1974. - 152 с.
8. Зайченко Ю. П. Нечеткие модели и методы в интеллектуальных системах. – К.: Издательский дом «Слово», 2008. – 333 с.

УДК 004.932.75'1

НАДЕРАН Э.,
ЗАЙЧЕНКО Ю.П.

ПОДХОД К РАЗРАБОТКЕ СИСТЕМЫ РАСПОЗНАВАНИЯ РУКОПИСНЫХ МАТЕМАТИЧЕСКИХ ВЫРАЖЕНИЙ ВВОДИМЫХ В ЭВМ В РЕЖИМЕ РЕАЛЬНОГО ВРЕМЕНИ

Предложен подход к структурному анализу рукописных математических выражений, основанный на синтаксических правилах и пространственных отношениях между объектами в выражении. Предложена динамическая база эвристических правил. Проведены экспериментальные исследования, позволяющие оценить качество распознавания рукописных математических выражений вводимых в ЭВМ в режиме реального времени.

An approach to structural analysis of handwritten mathematical expressions, based on syntactic rules and spatial relations between objects in an expression. An experimental study to assess the quality of heuristic rules-based recognition of handwritten mathematical expressions in real time.

Введение

Сегодняшнее начало эры пост-ПК, указывает на наличие глобальной тенденции роста использования вычислительных устройств с сенсорными экранами. Появляется необходимость ввода данных в ЭВМ без использования традиционных устройств ввода, таких как клавиатура и мышь. Возможность взаимодействия пользователя с ПК с помощью сенсорной функциональности станет безусловно самой естественной, удобной и быстрой альтернативой для ввода математических выражений в ЭВМ. Центральной задачей проблемы онлайн распознавания рукописных математических выражений является создание эффективного метода классификации рукописных математических символов, цифр и букв. Высокая вариативность в написании одних и тех же символов, и сложность определения структуры математических выражений затрудняют процесс распознавания.

Сегодня наиболее распространенными подходами для классификации рукописных символов являются: алгоритмы сопоставления с образцом [1], скрытые Марковские модели [2] [3], алгоритмы на основе искусственных нейронных сетей [4]. В последнее время начинает активно использоваться аппарат искусственных нейронных сетей с целью решения различных задач классификации. Преимуществом искусственных нейронных сетей является их способность к обобщению полученной информации, таким образом обученная на ограниченном множестве выборочных данных нейронная сеть может вернуть верный результат применительно к данным, которые не участвовали в процессе обучения. Поскольку решение об идентификации рукописных символов принимается системой распознавания в условиях неполной и неточной информации, можно считать подходящим разработку метода распознавания на основе нечеткого нейронного классификатора, сочетающего в себе преимущества нейронных сетей и нечетких систем логического вывода.

Целью данной статьи является рассмотрение подхода к структурному анализу математических выражений, разработанного с целью повышения качества распознавания рукописных математических выражений вводимых в ЭВМ в режиме реального времени.

Обзор процесса распознавания рукописных математических выражений

Для того чтобы правильно распознать математическое выражение необходимо точно знать значение каждого из символов, которые в него вошли, и определить расположение этих символов относительно друг друга.

Следовательно, задачу распознавания рукописных математических выражений будет целесообразным рассматриваться с точки зрения следующих двух процессов, которые были описаны в [5]:

1. Распознавание символов;
2. Определение пространственных отношений между символами.

В предлагаемой системе распознавания рукописных математических выражений, используется набор символов, который содержит строчные буквы латинского алфавита, наиболее часто встречаемые в формулах строчные буквы греческого алфавита, цифры и специальные математические символы. Отобранные символы позволяют писать тригонометрические и логарифмические функции, интегралы, арифметические корни различной степени и др. Все символы были разделены на две группы:

1. В первую группу вошли символы, которые могут быть написаны без отрыва пера, т.е. с помощью одного отрезка, как 0 и 1. Эти символы участвуют в этапе распознавания.
2. Во вторую группу вошли символы, которые могут быть получены из комбинации символов первой группы. Такие символы будут реконструированы на этапе структурного анализа из нескольких распознанных отрезков.

Для решения задачи распознавания рукописных символов был выбран нечеткий классификатор NEFCLASS [6], поскольку решение об идентификации рукописных символов принимается системой распознавания в условиях неполной и неточной информации. Преимуществом искусственных нейронных сетей является их способность к обобщению полученной информации, таким образом обученная на ограниченном множестве выборочных данных нейронная сеть может вернуть верный результат применительно к данным, которые не участвовали в процессе обучения. Подробно предлагаемый подход к решению задачи распознавания символов был рассмотрен в [7].

Структурный анализ математических выражений

После этапа распознавания символов главной задачей становится определение пространственных отношений между составляющими математического выражения. Предлагается подход к структурному анализу математических выражений, состоящий из трех этапов и позволяющий определить расположение распознанных нейронной сетью символов относительно друг друга. Подход включает в себя этап размещения, этап реконструкции и этап группировки символов, как показано на Рис.1. Написанное пользователем математическое выражение просматривается после каждого внесенного изменения, что дает возможность записывать составляющие математического выражения в любом порядке, а также вносить изменения в уже написанное выражение.

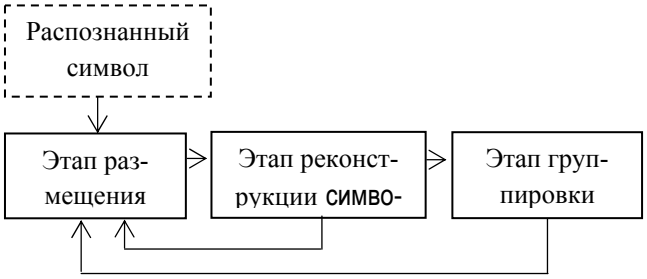


Рис.1 Схема структурного анализа математических выражений

Символы разделены на группы, в зависимости от допустимых и обязательных позиций размещения других символов относительно них. На Рисунке 2 схематически изображены все восемь возможных позиций. Под допустимой позицией расположения символов понимается возможная, но не обязательная к заполнению символом позиция. Под обязательной позицией понимается позиция обязательная к заполнению символом.

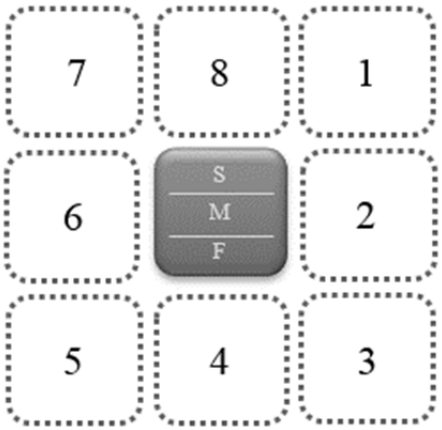


Рис.2 Схематическое изображение допустимых и обязательных позиций размещения символов относительно текущего рассматриваемого символа

На этапе размещения символов наилучшая позиция для размещения символа определяется по формуле:

$$NP = P*k$$

где P – процент попадания символа

k – коэффициент позиции, принимающий следующие значения:

0-для недопустимых позиций,

1-для допустимых позиций,

1,5- для обязательных позиций.

Этап размещения символов работает по следующему принципу:

- Если наилучшая позиция размещения для данного символа принимает значение ноль, то найти наилучшую позицию размещения для соседних символов относительно данного.
- Если ноль, то поместить данный символ на месте ближайшей допустимой или обязательной позиции.

На Рис.3 показан пример определения наилучшей позиции для размещения символов «2» и «3».

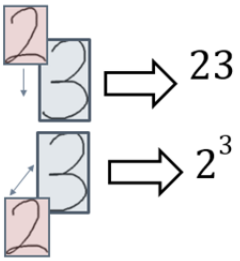


Рис. 3 Пример определения наилучшей позиции для размещения символов

Этап реконструкции символов решает следующие задачи:

1. Преобразование распознанных последовательных отрезков в соответствующий символ. Например, если отрезок «\» пересекает отрезок «/» в позиции M, то отрезки преобразуются в символ «x».
2. Преобразование распознанных последовательных отрезков в соответствующую комбинацию символов, имеющую семантическое значение.
3. Коррекция ошибочно распознанных отрезков, путем нахождения семантического значения комбинации символов в которую они входят.

Для решения поставленных задач используется динамическая база эвристических правил, пример которой приведен на Рис.4. Данная таблица основана на последовательности записи каждого последующего символа, и расположении символов относительно друг друга.

Текущий символ i	Символ S_{i-1}	Позиция s_i относительно s_{i-1}	Символ S_{i-2}	Позиция s_{i-1} относительно s_{i-2}	Результат
-	∨	M			∀
-	+	4			±
.	⎩	8			i
-	⎩	S			t
-	⎧	S			f
.	⎭	8			j
/	\	M			x
\	/	M			x
>	-	2 or F			→
/	-	S or M	c	M	∉
-	C	M			∈
-	Z	M			≠
o	O	7	/	7	%
|	-	F or M	|	S	π
⎩	-	F or M	⎭	S	π
⎩	~	F or M	⎭	S	π
n	1	6	s	6	sin
s	0	6	c	6	cos
m	I	6	l	6	lim

Рис. 4 Пример базы эвристических правил

Рассмотрим этап группировки. Некоторые символы позволяют сгруппировать несколько отдельных символов в одну группу. К таким символам относятся:

- Различного вида скобки,
- Математические аббревиатуры,
- Дробная черта,
- Сумма ∑,
- Произведение ∏,
- Интеграл,
- Арифметический корень,
- Точка, запятая.

Группа символов рассматривается как единое объединение, для нее существуют допустимые и обязательные позиции размещения соседних символов. На Рис. 5 приведен пример группировки символов с помощью круглых скобок.

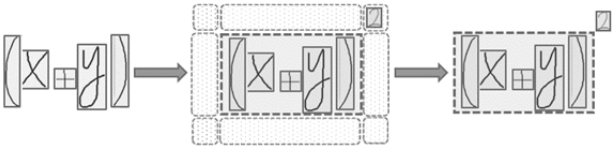


Рис. 5 Пример группировки символов с помощью круглых скобок

Для проверки эффективности работы предложенных подходов были проведены экспериментальные исследования, позволяющие оценить качество распознавания рукописных математических выражений. Программная часть системы реализована на языке C# на базе платформы Microsoft .NET. Преимущество использования платформы .NET Framework заключается в возможности реализации исполняемого кода независимого от аппаратной части, поскольку исходный код переводится компилятором в промежуточный байт-код Common Intermediate Language (CIL) и затем код исполняется виртуальной машиной Common Language Runtime (CLR), которая с помощью встроенного в неё JIT-компилятор преобразует промежуточный байт-код в машинные коды нужного процессора. Кроме того CLR заботится о базовой безопасности, управлении памятью и системе исключений. Необходимо отметить, что использование современной технологии динамической компиляции позволяет достигнуть высокого уровня быстродействия. Результат работы можно сохранить в формате изображения .png или в виде кода языка TEX [8].

Для обучения нечеткого классификатора NEFCLASS была использована выборка, состоящая из 260 рукописных математических выражений. Для генерации базы правил каждому образцу ставился в соответствие подходящий рукописный символ. После чего нейронная сеть была обучена с помощью генетического алгоритма обучения. Таким образом, были найдены значения весовых коэффициентов необходимые для инициализации нейронной сети.

Тестирование проводилось путем рукописного ввода математических выражений в режиме реального времени в планшетный ПК с помощью стилуса. Общая выборка для тестирования составила 140 математических выражений. Среднее время распознавания каждого символа составило 150 миллисекунд на компьютере со следующими характеристиками: CPU Intel Core 2 Quad с частотой 2.4 ГГц, RAM 4 Гб. Результаты тестирования отражены на Рис.6 и отражают качество распознавания и структурирования математических выражений.

Распознавание отрезков	Реконструкция символов (для правильно распознанных отрезков)	Структурный анализ математических выражений
91,54%	95,32%	71,29%

Рис.6 Результаты тестирования

Поле «Распознавание отрезков» показывает процент правильно классифицированных символов, написанных без отрыва пера с помощью одного штриха, и позволяет оценить эффективность распознавания. Поле «Реконструкция символов» отображает процент верно реконструированных символов из распознанных правильно отрезков. Поле «Структурный анализ математических выражений» показывает процент правильно структурированных выражений.

Выводы

В статье был описан предлагаемый подход к структурному анализу рукописных математических выражений вводимых в ЭВМ в режиме реального времени. Рассмотрена разработанная динамическая база эвристических правил, основанная на знаниях о порядке записи и пространственных отношениях между символами, позволяющая проводить реконструкцию символов, коррекцию неправильно распознанных символов в последовательности путем нахождения ее семантического значения. Для проверки эффективности работы предложенного подхода были проведены экспериментальные исследования, позволяющие оценить качество распознавания рукописных математических выражений. По результатам тестирования были получены следующие результаты: процент правильно классифицированных символов, написанных без отрыва пера с помощью одного штриха составил 91,54%, процент правильно структурированных выражений составил 71,29%.

Список литературы

1. Nakayama Y. A prototype pen-input mathematical formula editor. – EDMEDIA, 1993. - pp. 400-407.
2. Winkler H.-J., Fahrner H., Lang M. A soft-decision approach for structural analysis of handwritten mathematical expressions. – ICASSP, 1995. - pp. 2459-2462.
3. Kosmala A., Rigoll G. On-line handwritten formula recognition using statistical methods. - International Conference on Pattern Recognition, 1998. – pp. 1306-1308.

4. Dimitriadis Y. A new interactive mathematical editor, using on-line handwritten symbol recognition, and error detection-correction with an attribute grammar. – ICDAR, 1991. - pp. 885-893.
5. Dorothea Blostein. General Diagram-Recognition Methodologies. - Queen's University, Canada, 1995. – pp. 1-13.
6. Зайченко Ю.П. Нечёткие модели и методы в интеллектуальных системах. Учебное пособие для студентов высших учебных заведений. – К.: «Издательский Дом «Слово»», 2008. – с. 344.
7. Вісник НТУУ «КПІ» Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – К.:Век+, - 2012. - №57. – с.124-128.
8. Кнут Д. Э. Все про TEX. / Пер. с англ. М. В. Лисиной — Протвино: РДТЕХ, 1993. – с. 592

УДК 004.2

МОЛЧАНОВСЬКИЙ О.І.,
ЗНАХУРЕНКО В.П.

МЕТОД ІНДУКТИВНОГО НАВЧАННЯ В ОСНОВІ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ ПОДАРУНКІВ

В цій статті розглянута задача надання рекомендацій щодо вибору подарунків. Метод розв'язку даної задачі відноситься до методів індуктивного навчання. Розроблено метод, який будує нечітке дерево прийняття рішень, використовуючи алгоритм FuzzyID3. Запропоновано метод надання рекомендації подарунків з фактором впевненості в кожній рекомендації.

The subject of the article is the task of making recommendations on the choice of gifts. Method for solving this problem is one of the methods of inductive learning. The method was developed for generating a fuzzy decision tree by using the algorithm FuzzyID3. The method that recommends some gifts with membership value of each one was proposed.

1. Вступ

Кожна людина стикалася з проблемою вибору подарунків. Для того, щоб допомогти людям, можна запропонувати систему, яка б за певними критеріями та знанням про вподобання людини-адресата, висувала припущення щодо подарунку, який би був бажаний і корисний [1]. Таку систему будемо називати рекомендаційною. Потенційні користувачі подібної системи – це будь-які користувачі мережі Internet, які мають складності з вибором подарунку якійсь іншій людині. Рекомендаційні системи є активною областю досліджень в галузі інтелектуального аналізу даних та машинного навчання. Ця тематика є ключовою для таких міжнародних наукових конференцій як RecSys (recsys.acm.org), SIGIR (sigir.org), KDD (kdd.org).

Методи розв'язку даної задачі відносяться до методів індуктивного навчання. Таких алгоритмів існує багато, тому важливим кроком розв'язання поставленої задачі є вибір алгоритму, який би якнайкраще відповідав запропонованій галузі досліджень.

В рамках цієї статті розглядаються алгоритми побудови дерева прийняття рішень та множини правил виведення. Необхідно зазначити, що з дерева прийняття рішень можна отримати правила виведення та навпаки.

Один з перших алгоритмів індуктивного навчання був розроблений Quinlan. Це алгоритм ID3 [2], а також його наступна модифікація J48 [3]. Окрім того слід відзначити методи Prism, запропонований Cendrowska [4], та PART, розглянутий Frank та Witten [5].

Важливим аспектом індуктивного навчання та функціонування рекомендаційних систем є використання нечіткої логіки для опрацювання неоднозначностей у вхідних та вихідних даних. Одним з важливих екземплярів цього класу алгоритмів є метод FuzzyID3, запропонований Umano та ін. [6]. Метод, який розглядається в цій роботі, ґрунтується якраз на модифікації алгоритму FuzzyID3.

Попередньо також був проведений аналіз існуючих рішень, тобто систем, які розв'язують подібну задачу. Були розглянуті наступні ресурси:

1. gifts.com.ua;
2. tutdar.com.ua;
3. market.yandex.ua/gifts.xml;
4. millionpodarkov.ru.

За результатами аналізу виявилось, що всі перелічені системи не орієнтуються на вибір подарунку під конкретну людину, а більше спрямовані на урахування приводу для подарунку. Окрім того, можна зауважити, що в цих системах використовуються експертні, жорстко визначені оцінки щодо того, яким категоріям користувачів варто рекомендувати конкретні подарунки. Даний факт не дозволяє говорити про гнучкість функціонування та ефективність надання рекомендацій існуючими системами.

2. Постановка задачі

Призначенням системи рекомендацій подарунків в Інтернет є надання рекомендації подарунків користувачам.

Нехай $A = \{(a_i, v_i, p_i)\}$ – множина входів, де a_i – це атрибут, v_i – значення атрибуту a_i, p_i – це фактор впевненості щодо того, що атрибут a_i має значення v_i. Для нашої предметної області атрибути a_i – це стать, вік, вартість подарунку, хобі, привід подарунків тощо. Значення фактору впевненості p_i лежить в проміжку від 0 до 1.

Множина виходів $B = (m_i, q_i)$, де m_i – це рекомендований товар/об'єкт, q_i – степінь впевненості в цьому рекомендованому товарі. Аналогічно до p_i, q_i також лежить в проміжку від 0 до 1.

В якості m_i в нашій предметній області виступають типи подарунків, наприклад, офіційний, косметика, побутова техніка, комп'ютерна техніка, квитки в театр тощо.

Необхідно розробити такий метод, який може побудувати відображення $f : A \rightarrow B$, яке якнайкраще відповідає індивідуальним очікуванням користувача.

3. Побудова дерева рекомендації

Оскільки кожен з алгоритмів [1-4] може надати тільки правила виведення чи дерево прийняття рішень, листками якого буде лише один варіант подарунку, втрачається ідея альтернативи вибору користувача. Наявність рекомендації лише одного подарунку позбавляє дарувальника різноманітності вибору, тобто якщо він вже такий подарунок зробив, то нічого іншого вибрати він не зможе. Саме тому у нагоді може стати метод, який будує дерево прийняття рішень, листки якого містять множину пар: «подарунок – фактор впевненості».

Одним з таких методів є алгоритм FuzzyID3 [5]. Він є модифікацією алгоритму ID3. Проте ID3 вибирає тестовий атрибут на основі приросту інформації, що обчислює ймовірність звичайних даних, однак FuzzyID3 обчислює ймовірність приналежності даних до деякої множини.

3.1 Нечіткі атрибути

Нехай маємо набір даних D, який складається з окремих кортежів. Кожен кортеж містить s значень для атрибутів $A_1, A_2, \ldots, A_s$, один клас з множини $C = \{C_1, C_2, \ldots, C_n\}$ та число μ, яке відповідає впевненості тому, що обраний клас C_i відповідає зазначеним атрибутам. Для нашої предметної області атрибути відповідають характеристикам адресата подарунку, а клас – це окрема категорія подарунків: C_1 – офіційний, C_2 – предмет гардероба, C_3 – гастрономічний, C_4 – для курця, C_5 – ювелірний виріб, C_6 – живий подарунок (тварина), C_7 – косметика, C_8 – побутова техніка, C_9 – комп'ютерна техніка, C_{10} – квитки в театр, кіно, виставку, C_{11} – сувенір, C_{12} – гроші.

Нехай є нечіткі множини $F_{i1}, F_{i2}, \ldots, F_{im}$ для кожного атрибуту A_i (значення m змінюється для кожного атрибуту).

Наприклад, в нашій предметної області маємо атрибут "вік" із можливими значеннями: «до18», «від 18 до 24», «від 25 до 34», «від 35 до 44», «від 45 до 54», «від 55 до 64», «старше 64». Коли користувач вказує певний вік адресату подарунку, то система повинна враховувати нечіткість цих даних. Таким чином ми створюємо нечіткі множини (fuzzy sets) для кожного можливого значення атрибуту «вік». Нижче наводяться приклади цих нечітких множин, де поруч з кожним елементом множини вказується його степінь приналежності цій множині.

- «до 18» = {1/до18, 0.3/18_24, 0.1/25_34};
- «від 18 до 24» = {0.4/до18, 1/18_24, 0.5/25_34, 0.1/35_44};
- «від 25 до 34» = {0.1/до18, 0.5/18_24, 1/25_34, 0.4/35_44};
- «від 35 до 44» = {0.1/18_24, 0.5/25_34, 1/35_44, 0.4/45_54};
- «від 45 до 54» = {0.1/25_34, 0.5/35_44, 1/45_54, 0.4/55_64};
- «від 55 до 64» = {0.1/35_44, 0.5/45_54, 1/55_64, 0.4/ старше64};
- «старше 64» = {0.2/45_54, 0.6/55_64, 1/старше64}

Нечіткі множини «близькі», «середні», «дальні» для атрибуту "стосунки" мають вигляд:

- «близькі» = {1/кохана людина, 0.9/член сім'ї, 0.8/близький друг, 0.4/родич, 0.2/ колега з роботи};
- «середні» = {0.4/кохана людина, 0.5/член сім'ї, 0.6/близький друг, 1/родич, 0.8/колега з роботі, 0.6/приятель, 0.3/інше};
- «дальні» = {0.7/родич, 0.9/колега з роботі, 1/приятель, 0.8/інше}.

Аналогічно будуються нечіткі множини значень й для інших атрибутів.

3.2 Карти атрибутів

Проте в описану вище схему важко вписати деякі атрибути. Наприклад, розглянемо атрибути, які послуговуються для визначення вподобань адресата подарунку. Ми дозволяємо користувачу обрати декілька вподобань (наприклад, спорт, йога, музика, відвідування театрів, допінг тощо), а також вказати наскільки це вподобання притаманне адресату подарунка (за шкалою від 1 до 5, де 1 – зовсім не притаманне, 5 – дуже притаманне).

Можна відзначити, що запропонована схема має два суттєвих недоліки. По-перше, кількість таких вподобань може бути дуже великою і це може вплинути на час побудови дерева прийняття рішень та надання подальших рекомендацій. По-друге, підхід, в якому кожне вподобання розглядається як окремий атрибут, не може бути ефективно масштабований (з точки зору додавання нових вподобань в систему пізніше).

Тому було запропоновано об'єднати усі вподобання в один атрибут, який ми назвали «хобі». Всі можливі вподобання ми розглянули з точки зору відповідності чотирьом критеріям: фізична чи розумова діяльність, пасивна чи активна діяльність. Причому ці критерії утворюють дві пари, які пов'язують між собою протилежні сутності (рис. 1).

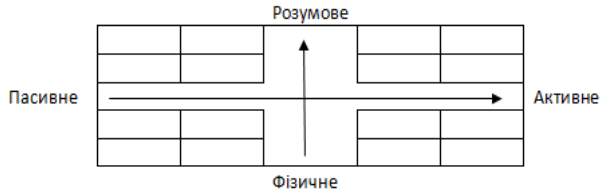


Рисунок 1 – Карта хобі

Тепер кожне вподобання адресата буде відображене на двовимірний простір, який визначається вимірами: «фізична-розумова діяльність» та «активна-пасивна діяльність». Наприклад, відвідування тренажерного залу бути тяжіти до фізичної та пасивної діяльності, а заняття екстремальними видами спорту – до фізичної та активної діяльності.

Для формалізації даного розподілення вподобань по критеріям були введені так звані «карти хобі» (рис. 1). Вони уявляють собою двовимірну матрицю розміром 4×4, де елементом матриці є значення від 0 до 1, що відповідає тому, наскільки сильно конкретне вподобання «присутнє» в певній точці двовимірного простору хобі.

Нарешті, всі вподобання адресата подарунку об'єднуються в одну матрицю шляхом підсумовування карт окремих хобі із врахуванням степеню притаманності (який ми позначимо через α_i; α_i набуває значень в проміжку від 0 до 1). Через H_i позначимо матрицю, яка відповідає карті i-го хобі. Тоді загальна карта хобі адресата вираховується за формулою (1).

$$H = \frac{1}{c}\sum_{i=1}^{n}\alpha_i \cdot H_i \qquad (1)$$

де $\alpha_i \in [0;5]$ – ранг i-ого хобі адресата, $i \in [1;n]$ (в нашій системі наразі $n = 19$ – загальна кількість вподобань), c – кількість вподобань, для яких $\alpha_i>0$.

Аналогічно будується карта для атрибуту «привод подарунку» (рис. 2). Тут теж маємо дві пари протилежних критеріїв: приватний-публічний і традиційний-офіційний привід.

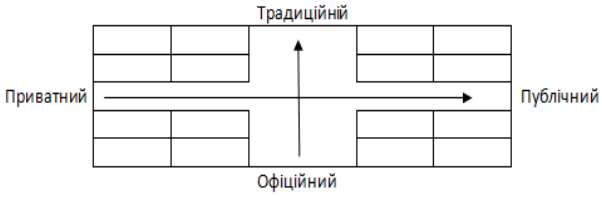


Рисунок 2 - Карта приводів

3.3 Алгоритм побудови дерева прийняття рішень

Алгоритм для побудови нечіткого дерева прийняття рішень для надання рекомендацій подарунків приймає на вхід множину навчальних даних D. Розглянемо кроки алгоритму.

Крок 1. Згенерувати корінь дерева як набір всіх даних D, задати ступінь впевненості μ для нечіткої множини, яка відповідає оцінці подарунка адресатом у навчальній вибірці. Обрати корінь як поточний вузол.

Крок 2. Якщо поточний вузол t з набором нечіткої множини даних D задовольняє одній з наступних умов:

- частка набору даних класу C_k більша або рівна θ_r:

$$\frac{\left|D^{C_k}\right|}{\left|D^{C}\right|} \geq \theta_r; \qquad (2)$$

- кількість наборів даних менше θ_n:

$$\left|D\right| < \theta_n; \qquad (3)$$

- немає більше атрибутів для класифікації,

то вважати цей вузол листком і розрахувати значення для ступенів впевненості для кожного класу з множини C (у (2) та (3) θ_r та θ_n є пара-

метрами керування, про які мова буде йти пізніше). Інакше

Крок 3.1. Для атрибутів A_1, A_2, …, A_s розрахувати прирости інформації $G(A_i, D)$ за формулою (4) та вибрати атрибут A_{max}, що максимізує їх.

$$G(A_i, D) = I(D) - E(A_i, D) \quad (4)$$

де $I(D)$ – міра інформації, що розраховується за формулою (5), та $E(A_i, D)$ – очікувана інформація за формулою (7).

$$I(D) = -\sum_{k=1}^{n} p_k \cdot \log_2 p_k \quad (5)$$

де p_k – фактор приналежності даних класу C_k в множині D:

$$p_k = \frac{\left|D^{C_k}\right|}{|D|} \quad (6)$$

$$E(A_i, D) = \sum_{j=1}^{m} (p_{ij} \cdot I(D_{F_{ij}})) \quad (7)$$

де p_{ij} – фактор приналежності значення j (нечіткого набору) серед усіх можливих значень атрибуту i:

$$p_{ij} = \frac{\left|D_{F_{ij}}\right|}{\sum_{j=1}^{m} \left|D_{F_{ij}}\right|} \quad (8)$$

Крок 3.2. Розбити D на нечіткі множини D_1, D_2, …, D_m згідно з обраним атрибутом A_{max}, де значення ступеню впевненості буде розраховуватися як добуток ступеню приналежності множені D на значення $F_{max,j}$ для A_{max} в D. При цьому для всіх D_i атрибут A_{max} буде видалений з подальшого розгляду.

Деякі атрибути, а саме хобі та приводи подарунків використовують двовимірний масив в якості елементів нечітких множин (див. розділ 3.2). Це спричиняє деякі складності, зокрема, як перевести матрицю приналежності, наприклад, деякого хобі до окремого числа-значення впевненості μ_a (яке лежить в проміжку від 0 до 1). Було запропоновано використання зведення матриці (9).

$$\mu_{a_1} = \frac{1}{16} \sum_{i}^{n} \sum_{j}^{n} h_{ij} f_{ij}^{(a_1)} \in [0;1] \quad (9)$$

де $H = \left\| h_{ij} \right\|$ – матриця хобі, $F_{a_1} = \left\| f_{ij}^{(a_1)} \right\|$ – матриця типізації хобі. Число 16 в знаменнику відповідає кількості комірок матриці хобі (нагадуємо, що вона має розмірність 4×4).

Крок 3.3. Згенерувати нові вузли t_1, t_2, ..., t_m для нечітких множин D_1, D_2, …, D_m та позначити нечіткою множиною $F_{max,j}$ ребро, що з'єднує вузли t_j та t.

Крок 3.4. Замінити D на D_j (j = 1, 2, …, m) та перейти на пункт 2.

3.3 Результати побудови дерева

Для побудови дерева можна використовувати різні значення параметрів θ_n та θ_r, відповідно до формул (2) та (3), які відповідають за деталізацію побудови дерева.

Приклад побудованого дерева зі значення $\theta_r = 0{,}7$ та $\theta_n = 100$ представлено на рисунку 3. Кожен листок дерева містить інформацію про тип подарунку та степінь приналежності вподобань опитуваних, що наведено у таблиці 1.

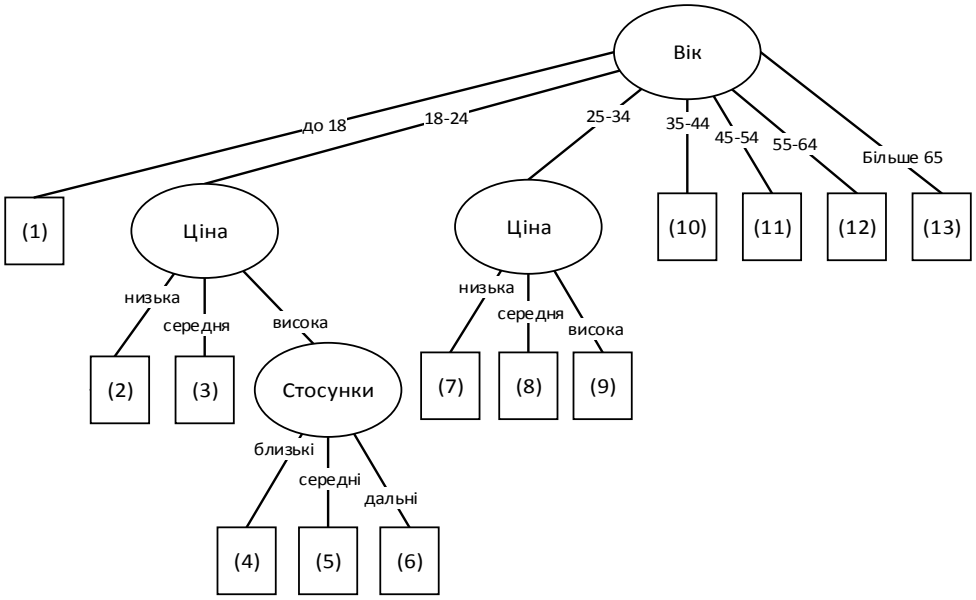


Рисунок 3 - Приклад нечіткого дерева прийняття рішень

Таблица 1 - Листки нечіткого дерева прийняття рішень

Клас / Листок	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
(1)	0,01	0,12	0,04	0,01	0,17	0,07	0,02	0,21	0,02	0,04	0,12	0,04
(2)	0,02	0,12	0,11	0,01	0,12	0,11	0,01	0,08	0,01	0,10	0,12	0,06
(3)	0,01	0,18	0,03	0,02	0,17	0,12	0,02	0,13	0,02	0,06	0,12	0,05
(4)	0,02	0,14	0,01	0,01	0,25	0,06	0,03	0,22	0,02	0,02	0,10	0,08
(5)	0,02	0,14	0,01	0	0,19	0,08	0,03	0,27	0,02	0,03	0,13	0,06
(6)	0,01	0,09	0	0	0,12	0,18	0,03	0,40	0	0	0,14	0
(7)	0,02	0,13	0,08	0,02	0,12	0,15	0,01	0,08	0,02	0,08	0,11	0,06
(8)	0,01	0,18	0,02	0,02	0,17	0,13	0,02	0,13	0,02	0,05	0,11	0,06
(9)	0,01	0,14	0	0,01	0,25	0,08	0,05	0,22	0,01	0,02	0,09	0,07
(10)	0,05	0,15	0,01	0,01	0,24	0,11	0,06	0,16	0,01	0,03	0,08	0,05
(11)	0,09	0,14	0	0,01	0,24	0,11	0,10	0,12	0,02	0,03	0,09	0,04
(12)	0,08	0,14	0	0,03	0,25	0,12	0,11	0,11	0,01	0,03	0,08	0,03
(13)	0,07	0,14	0	0,04	0,25	0,12	0,14	0,10	0,01	0,03	0,07	0,03

Були проведені експерименти зі значенням $\theta_r = 0{,}7$ та різними значеннями для $\theta_n = 100$, 70, 50, 40, 30, 25, 20, 10 (табл. 2).

Таблиця 2 - Результати експериментів

θ_n	Глибина дерева	Кількість вузлів	Максимальна кількість елементів в черзі
10	7	13530	8580
20	7	4356	2402
25	7	2565	1362
30	5	257	196
40	5	187	132
50	5	145	93
70	5	81	41
100	3	17	10

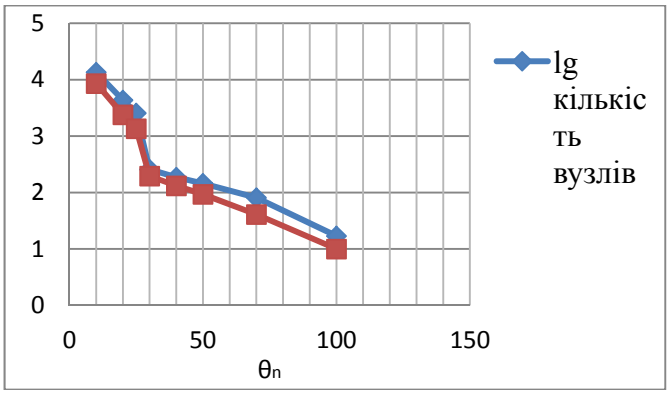


Рисунок 4 - Графіки залежності розмірності дерева від деталізації

Кількість вузлів дерева, як бачимо, зростає експоненціально зі зростанням деталізації, яке є обернено пропорційним до θ_n. З цього слідує, що чим детальнішим та більш точним має бути нечітке дерево прийняття рішень, тим більше часу необхідно витратити на його обрахунок.

4. Надання рекомендацій

Для надання рекомендацій на основі побудованого нечіткого дерева прийняття рішень, розглянемо наступний процес. На вхід подається один кортеж зі значеннями нечітких атрибутів (як описано в розділі 3.1). Для того, щоб отримати рекомендацію для нечіткого дерева прийняття рішень необхідно, починаючи з кореневого вузла перейти до тестового атрибуту у відповідному вузлі та повторити цю операцію допоки не дійшли до листового вузла. На відміну від звичайних дерев прийняття рішень, у нечітких деревах необхідно зробити переходи більше, ніж по одному ребру, враховуючи ступінь приналежності цього ребра.

Тепер потрібно вирішити три операції. По-перше, для об'єднання значень ступенів приналежності по ребрах, ми використовуємо добуток ступенів приналежності по ребрах одного шляху від кореня до листка. По-друге, отриманий ступінь на шляху помножується на фактор впевненості в листку для кожного з класів C_i. Нарешті, об'єднання ступенів приналежності кожного класу в нечіткому дереві прийняття рішень відбувається шляхом обрахунку суми ступенів для конкретного класу в усіх листках дерева. Додатково можна провести нормалізацію отриманих факторів приналежності для кожного класу.

Розглянемо приклад для нашої предметної області для побудованого дерева на рис. 3. Нехай користувач увів наступні значення атрибутів: стать = «жіноча», вік = «від 25 до 34», стосунки = «близький друг», вартість = «100 - 300» (інші атрибути не впливають на отримання рекомендації для цього дерева). На рис. 5 наведені ступені приналежності для кожного значення атрибутів (див. підписи до ребер дерева).

Для листка (2) дерева для класу C_1 отримуємо наступне значення фактору: $0{,}5\times0{,}4\times0{,}02 = 0{,}004$. Значення для всіх класів по всіх вузлах наведені в таблиці 3.

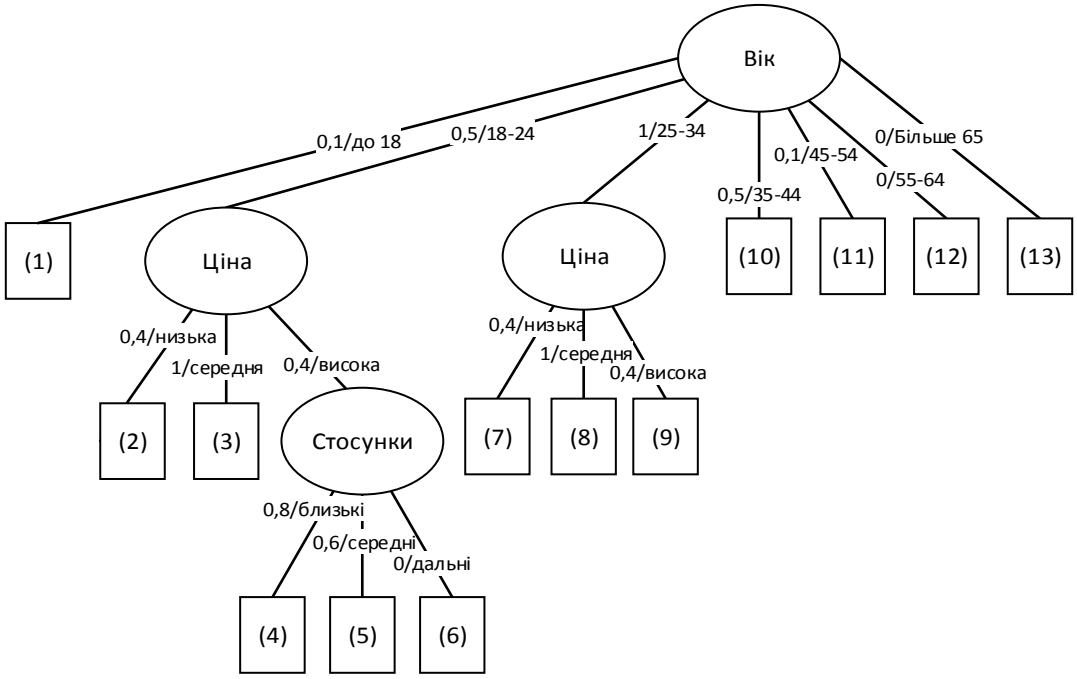


Рисунок 5 – Дерево прийняття рішень із факторами впевненості для вхідного кортежу

Таблиця 3 – Значення факторів впевненості по класах для дерева (рис. 5)

Клас / Листок	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
(1)	0,001	0,012	0,004	0,001	0,017	0,007	0,002	0,021	0,002	0,004	0,012	0,004
(2)	0,004	0,024	0,022	0,002	0,024	0,022	0,002	0,016	0,002	0,02	0,024	0,012
(3)	0,005	0,09	0,015	0,01	0,085	0,06	0,01	0,065	0,01	0,03	0,06	0,025
(4)	0,0032	0,0224	0,0016	0,0016	0,04	0,0096	0,0048	0,0352	0,0032	0,0032	0,016	0,0128
(5)	0,0024	0,0168	0,0012	0	0,0228	0,0096	0,0036	0,0324	0,0024	0,0036	0,0156	0,0072
(6)	0	0	0	0	0	0	0	0	0	0	0	0
(7)	0,008	0,052	0,032	0,008	0,048	0,06	0,004	0,032	0,008	0,032	0,044	0,024
(8)	0,01	0,18	0,02	0,02	0,17	0,13	0,02	0,13	0,02	0,05	0,11	0,06
(9)	0,004	0,056	0	0,004	0,1	0,032	0,02	0,088	0,004	0,008	0,036	0,028
(10)	0,025	0,075	0,005	0,005	0,12	0,055	0,03	0,08	0,005	0,015	0,04	0,025
(11)	0,009	0,014	0	0,001	0,024	0,011	0,01	0,012	0,002	0,003	0,009	0,004
(12)	0	0	0	0	0	0	0	0	0	0	0	0
(13)	0	0	0	0	0	0	0	0	0	0	0	0
Сума	0,0716	0,5422	0,1008	0,0526	0,6508	0,3962	0,1064	0,5116	0,0586	0,1688	0,3666	0,202
Нормалізовані суми	0,0222	0,1680	0,0312	0,0163	0,2016	0,1227	0,0330	0,1585	0,0182	0,0523	0,1136	0,0626

Отже, для вхідного кортежу даних найкращими рекомендаціями подарунків є ювелірний виріб (20%), предмети гардеробу (16,8%), побутова техніка (15,9%), тварина (12%) та сувеніри (11%). Окрім того, даний підхід дозволяє виявити найбільш небажані подарунки: подарунки для курця (1,6%), комп'ютерна техніка (1,8%), офіційний подарунок (2%).

5. Висновки

Була запропонована система, яка за певними критеріями та знанням про вподобання людини, висуває припущення щодо подарунку, який буде бажано отримати.

Зроблено аналіз наявних аналогів за результатами яких була обґрунтована необхідність створення такої системи, яка би спрощувала вибір подарунків користувачами тематичних веб сайтів (інтернет-магазинів).

Проведений аналіз відомих алгоритмів засвідчив необхідність розробки власного математичного методу. Було запропоновано модифікацію методу FuzzyID3 для урахування комплексних атрибутів (наприклад, вподобання та привід подарунку). Запропонований метод дозволяє легке масштабування - врахування додаткових атрибутів.

Поза межами уваги лишились два важливих питання. По-перше, запропонований метод має суттєвий недолік, який полягає у підсиленні типових рекомендацій. Для нашої предметної області це означає неможливість рекомендування оригінальних подарунків. По-друге, при практичному застосуванні системи ми стикнемось із великою кількістю категорій подарунків, які варто було би об'єднати у деяку таксономію. Тож необхідно модифікувати запропонований алгоритм для можливості роботи із класами категорій подарунків.

Список літератури

1. Знахуренко В.П. Система надання рекомендацій щодо вибору подарунків. Тези IV Всеукраїнської заочної науково-практичної конференції «Сучасні інформаційні технології» Київ. 2013. – 115 с.
2. R. Quinlan (1986). Induction of decision trees. Machine Learning. 1(1):81-106.
3. Ross Quinlan (1993). C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, CA.
4. J. Cendrowska (1987). PRISM: An algorithm for inducing modular rules. International Journal of Man-Machine Studies. 27(4):349-370.
5. Eibe Frank, Ian H. Witten: Generating Accurate Rule Sets Without Global Optimization. In: Fifteenth International Conference on Machine Learning, 144-151, 1998.
6. Umano M., Okamoto H., Hatono I., Tamura H., Kawachi F., Umedzu S., Kinoshita J. Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems. - Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on, 2113 - 2118 vol.3

УДК 621.396.946.2

КУЛАКОВ Ю. А.,
ВОРОТНИКОВ В. В.,
ГУМЕНЮК И. В.

АНАЛИЗ СТРУКТУРЫ ТЕЛЕКОММУНИКАЦИОННОЙ СЕТИ ПУТЕМ ПРЕДСТАВЛЕНИЯ ЕЕ ТОПОЛОГИИ ПРЕДФРАКТАЛЬНЫМ ГРАФОМ

В работе проведен анализ формирования предфрактальных графов телекоммуникационных сетей с использованием свойства самоподобия, а также рассмотрен подход для создания моделей таких сетей. Показано, что различие между топологией реальной сети и её модели в виде предфрактального графа может быть оценено при помощи удаление новых рёбер l - ранговой затравки. Полученные результаты позволяют выделять на структуре реального графа сети фрагменты, которые хорошо описываются предфрактальными графами, а значит, имеют масштабно-инвариантные характеристики.

The paper analyzes the formation prefractal graphs telecommunication networks using the self-similarity property and considered approach to create models of such networks. It is shown that the difference between the actual network topology and its model as predfractal graph can be estimated using the removal of new edges l-rank seed. The results are used to distinguish the structure of the real network graph fragments, which are well described prefractal graph, and thus have a scale-invariant characteristics.

1. Постановка проблемы

Современные вычислительные системы и телекоммуникационные сети могут состоять из большого количества узлов и связей. В связи с этим возрастают трудности проектирования топологии таких систем, что связано с недостаточно развитой математической и инструментальной базой для поиска оптимальных решений. Проблема состоит в том, что при решении такого рода задач оптимизацию приходится выполнять на весьма непростых комбинаторных объектах – графах связей с большим количеством вершин и ребер, а комбинаторно-переборный характер задач не позволяет получить точное решение, возникающих при этом задач большой размерности [1].

Если представить сложную сеть в виде фрактала с циклически повторяющимися структурами, то задача моделирования их структуры сводится к определению момента фазового перехода диссипативных топологических структур сети, с последующим созданием модельных объектов с фрактальной структурой и изучением их геометрических характеристик [1, 2].

В связи с этим задача создания математических методов и программных инструментальных средств построения больших графов с прогнозируемыми параметрами становится все более актуальной.

2. Анализ последних исследований и публикаций

Существует большой класс задач для решения которых требуется обработка сложных дискретных структур. Задачи, связанные со сложными структурами, возникают в теории перколяции [3], в теории многоуровневых иерархических систем [4], [5], в моделировании глобальных структур [6] и т. д. Структуры, которые не проектируются специальным образом, в большинстве своем обладают свойством самоподобия [8]. Такие структуры называют фрактальными [9].

Целью работы является представление топологии реальной сети с помощью графа, имеющего свойства самоподобия фрактала.

3. Формулировка задачи исследования

Пусть $G=(V,E)$ - неориентированный граф, который описывает топологию заданной сети, где V - вершины графа, E -ребра графа.

Для построения фрактального (предфрактального) графа определено множество структур графа, состоящих из различного числа элементов, которые могут быть использованы в качестве затравки $H=(W,Q)$, W - вершины сегмента графа затравки, Q - ребра сегмента графа затравки.

Необходимо:

1) у намеченной для замещения вершины $\tilde{v} \in V$ выделить множество, смежных ей вершин $\tilde{V} = \{\tilde{v}_j\} \subseteq V$;

2) каждую вершину $\tilde{v}_j \in \tilde{V}$ соединить ребром с одной из вершин затравки $H = (W, Q)$;

3) используя рекурсивно операцию замены вершин затравкой H в l этапов, необходимо сформировать фрактальный граф.

Введем ограничения:

1) размеры элементов структур $H = (W, Q)$ различны;

2) вершины соединяются случайным образом или по определенному принципу самоподобия.

Отличие топологии реальной сети и ее предфрактального графа необходимо оценить как разницу структур: графа $G = (V, E)$ реальной сети и полученного фрактального графа $G_L = (V_L, E_L)$ ее модели.

4. Изложения основного материала

Фрактальные графы определяются как объединение графа и фрактала, с присущими свойствами фракталов: самоподобием, дробной размерностью, масштабной инвариантностью и т. д. [4, 13].

Фрактальные графы строятся иерархически: вершина графа вышестоящего уровня раскрывается в виде графа, подобного вышестоящему [10]. Процесс раскрытия уровня еще называют затравкой вершины графа.

В литературе термином «затравка» принято называть какой-либо связный граф $H = (W, Q)$. Для определения предфрактального графа [10] выполняют операцию замены вершины затравкой, суть которой заключается в следующем.

Предфрактальный граф обозначим через $G_L = (V_L, E_L)$, где V_L – множество вершин графа, а E_L – множество его ребер. Определим его рекуррентно, заменяя в построенном на предыдущем этапе графе $G_l = (V_l, E_l)$ каждую его вершину затравкой $H = (W, Q)$.

На этапе $l = 1$ предфрактальному графу соответствует затравка $G_1 = H$. Фрактальный граф $G = (V, E)$, порожденный затравкой $H = (W, Q)$, определяется бесконечной траекторией [10].

Использование операции замены вершины затравкой в процессе формирования предфрактального графа G_L, для элементов $G_l = (V_l, E_l)$, $l \in \{1..L-1\}$, его траектории позволяет ввести в общем виде отображение:

$$\varphi^t(V_l) = V_{l+t}, \qquad (1)$$

где множество V_l – прообраз множества V_{l+t}.

Для предфрактального графа $G_L = (V_L, E_L)$, ребра, появившиеся на l-ом этапе порождения, будем называть ребрами ранга l.

Если из предфрактального графа $G_L = (V_L, E_L)$, порожденного n-вершинной затравкой $H = (W, Q)$, последовательно удалить все старые ребра (ребра ранга l), то исходный граф распадется на множество связных компонент $\{B_L^{(1)}\}$, каждая из которых изоморфна [11] затравке $H = (W, Q)$.

При удалении из предфрактального графа $G_L = (V_L, E_L)$ всех ребер рангов $l = 1..L-r$ получим множество $\{B_{L,i}^{(r)}\}$. Очевидно, что всякий блок $B_L^{(r)} = (U_L^{(r)}, M_L^{(r)})$ является предфрактальным графом $B_r = (U_r, M_r)$, порожденным затравкой $H = (W, Q)$.

Для любой вершины из траектории предфрактального графа $G_L = (V_L, E_L)$ справедливо:

$$\varphi^t(v_j) = U_{L+t,j}^{(t)}, \ \varphi^t(v_j) = B_{l+t,j}^{(t)}, \qquad (2)$$

где $B_{L+t,j}^{(t)} = (U_{L+t,j}^{(t)}, M_{l+t,j}^{(t)}) \subseteq G_{l+t}$.

Аналогично,

$$\varphi^t(U_{l,i}^{(r)}) = U_{l+t,i}^{(r+t)}, \ \varphi^t(B_{L,i}^{(r)}) = B_{L+t,i}^{(r+t)}, \qquad (3)$$

где $r \in \{1..L-t\}$, $i \in \{1..n^{L-r}\}$.

Обобщением описанного процесса формирования предфрактального графа $G_L = (V_L, E_L)$ является такой случай, когда вместо единственной затравки $H = (W, Q)$ используется множество затравок.

На рис. 1 жирными линиями обозначены ребра затравки 1-го ранга. Линии средней толщины - ребра затравок 2-го ранга, а тонкими линиями соответственно изображены ребра затравок 3-го ранга.

При формирования предфрактального графа $G_L = (V_L, E_L)$ производится ровно $\frac{n^L - 1}{n - 1}$ операций замены вершины затравкой. При необходимости можно убедиться в том, что каждая операция приносит графу $G_L = (V_L, E_L)$ по $(k_H - 1)$-ой внутренней грани.

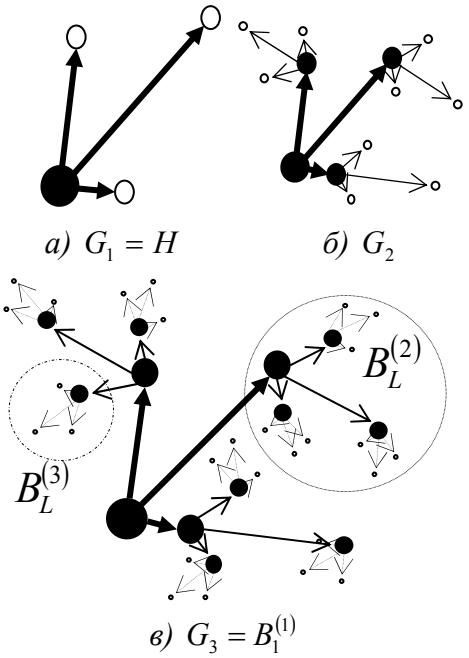


Рис. 1. Предфрактальные графы, порожденные затравкой 3-го ранга

Поэтому общее число $(k_H - 1)\dfrac{n^L - 1}{n - 1}$ внутренних граней предфрактального графа $G_L = (V_L, E_L)$ в сумме с одной внешней гранью дают формулу:

$$k_{G_L} = 1 + (k_H - 1)\frac{n^L - 1}{n - 1}. \qquad (4)$$

Для предфрактального графа $G_L = (V_L, E_L)$, порожденного множеством затравок, через $k_{\min} = \min\limits_{t=2,T} k_{H_t}$, $k_{\max} = \max\limits_{t=2,T} k_{H_t}$ обозначим минимальное и максимальное число граней среди затравок из множества $\mathrm{H} = \{H_t\}$, где t - количество затравок, применяемых для покрытия моделируемого графа.

Число граней k_{G_L} предфрактального графа $G_L = (V_L, E_L)$, порожденного множеством затравок, ограничено двойным неравенством:

$$(k_{\min} - 1)\frac{n^L - 1}{n - 1} \le k_{G_L} - 1 \le (k_{\max} - 1)\frac{n^L - 1}{n - 1}. \qquad (5)$$

Отметим, что правая часть неравенства (5) достигается, когда при формировании предфрактального графа $G_L = (V_L, E_L)$ участвует только одна затравка из множества $\mathrm{H} = \{H_t\}$ с числом граней равным $k_{\max}$. Левая часть (5) достигается, когда предфрактальный граф $G_L = (V_L, E_L)$ порожден затравкой $\mathrm{H} = \{H_t\}$ с минимальным числом граней $k_{\min}$.

Пример. Построим модель топологии сети в виде предфрактального графа. Пусть сеть задана графом (рис. 2), которой не является канонически фрактальным.

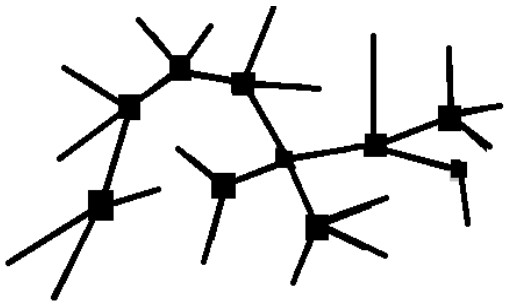


Рис. 2. Пример топологии сети

Выделим связный граф, который будет затравкой для исходного предфрактального графа (рис. 3).

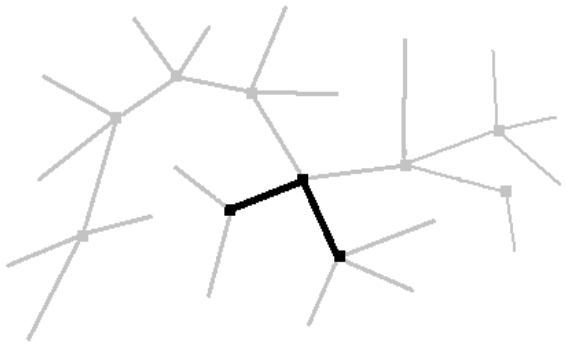


Рис. 3. Затравка предфрактального графа

Используя операцию замены вершин затравкой, суть которой описана выше, по принципу самоподобия получаем предфрактальный граф соответствующего ранга.

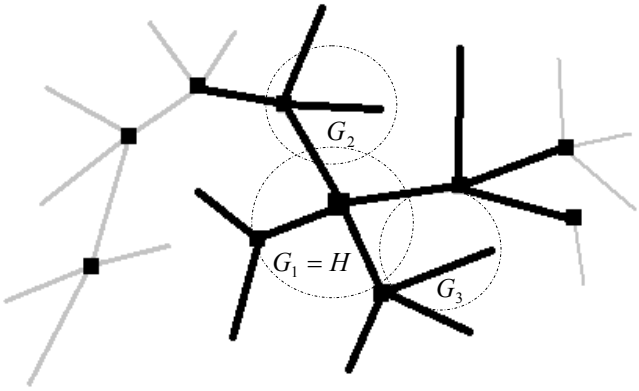


Рис. 4. Схема формирования предфрактального графа

Стоит отметить, что при порождении 3-ранговой затравкой предфрактального графа последовательно удаляются старые ребра. Это видно из области G_3. Для порождения конечно-

го предфрактального графа необходимо провести 4 операций замены вершин затравкой.

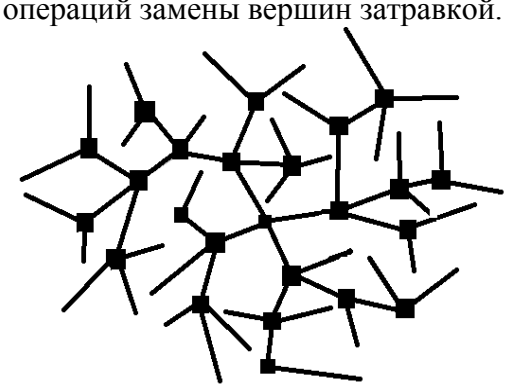


Рис. 5. Результат операций замены вершин графа затравкой H_t

В соответствии с (4-6) было получено значения минимального и максимального количества граней предфрактального графа, которое можно получить путем замены вершин реального графа затравкой из $H = (W, Q)$. Для нашего случая при $k_{\min} = 2$ и $k_{\max} = 4$ зависимость количества граней k_{G_L} от ранга l приведена на рис. 6.

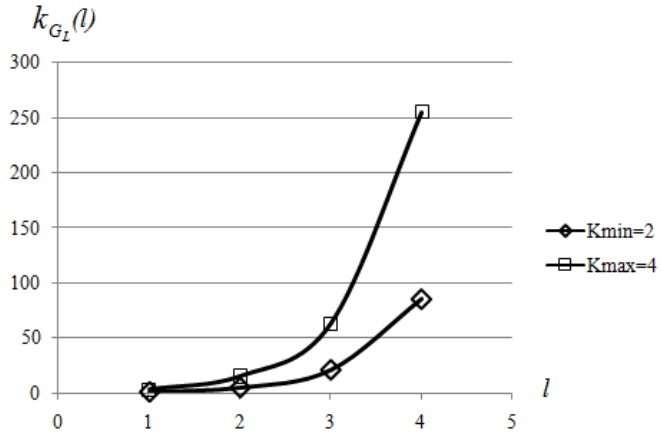


Рис. 6. Зависимость количества граней k_G от ранга l

Анализ зависимости (рис. 6) свидетельствует о том, что для реального графа сети (рис. 2), состоящего из 9 узлов и 29 ребер, при использовании трехузловой затравки (рис. 3), для построения модели топологии в виде предфрактального графа необходимо выполнение четырех рекурсивных процедур замещения узлов реальной сети затравкой H_1.

Построенный предфрактальный граф имеет 25 узлов, соединенных 62 ребрами.

Для приведения предфрактального графа к модели топологии реальной сети (рис. 7) из его структуры удалены лишние узлы и ребра.

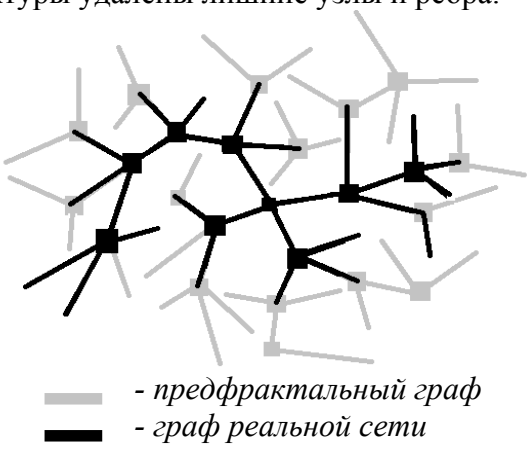


Рис. 7. Предфрактальный граф

5. Выводы

Анализ сложной сети путем представления ее структуры в виде фрактала с циклически повторяющимися структурами существенно снижает трудоемкость решения задач, связанных со сложными структурами и системами в целом.

В качестве элементов предфрактального графа можно рассматривать его подграфы-затравки или, в зависимости от постановки задачи, блоки различных рангов. Причем установленные правила соединения подграфов-затравок при помощи старых ребер, позволяют говорить о наличии (или об отсутствии) у моделируемых графов фрактальных свойств.

Список литературы

1. Корнеев В. В. Архитектура вычислительных систем с программируемой структурой / В. В. Корнеев. – Новосибирск: Наука. – 1985. – 168 с.
2. Корнеев В.В. Параллельные вычислительные системы / Корнеев В.В. – М.: Нолидж. – 1999. – 320 с.
3. Charles J. Colbourn. Combinatorial Models and Network Reliability // [Электронный ресурс]:/ J. Colburn – 6-th International WorkShop on Design of Reliable Communication Networks: DRCN, La Roshelle, France. – 09 October 2007. – Режим доступу до ресурсу: http://www.drcn.org/drcn07/ keynotes/ colburn.pdf.
4. С. Уэлстид. Фракталы и вейвлеты для сжатия изображений. Учебное пособие: – М.: Изд-во Триумф. – 2003. – 320 с.
5. Месарович М. Теория иерархических многоуровневых систем. / М. Месарович, Д. Мако, И. Такахара // – М.: Мир. – 1973. – 346 с.

6. Munzner T. Interactive Visualization of Large Graphs and Networks. /Tamara Munzner [Электронный ресурс]: Ph. D. Thesis Defense. Stanford Univ., 10 april 2000. – http://www.graphics.stanford.edu/~munzner/talks/defense/sld001.htm
7. Сомик К. В. Связные информационные структуры. /К. В. Сомик – М.: Финансы и статистика. – 2005. – 176 с.
8. Бобылева Е.В. Алгоритмы распознавания графов на предфрактальность / Е. В. Бобылева // Питання прикладної математики і математичного моделювання. – 2003. – №11. – С. 9 – 15.
9. Турчин В. Ф. Феномен науки. Кибернетический подход к эволюции. / В. Ф. Турчин. – М.: ЭТС. – 2000. – 420 с.
10. Кочкаров А.А. Предфрактальные графы в проектировании и анализе сложных структур. / А. А. Кочкаров, Р. А. Кочкаров // [Электронный ресурс]: Препринт ИПМ им. М.В. Келдыша РАН. – Москва. – 2003. – Режим доступу до ресурсу: http://www.keldysh.ru/papers/2003/prep10/prep2003_10.html.
11. В. А. Емеличев. Лекции по теории графов / Емеличев В.А., Мельников О.И., Сарванов В.И., Тышкевич Р.И. // – М.: Наука. – 2013. – 392 с. – ISBN: 978-5-397-03289-6
12. Щербина О. А. Методологические аспекты динамического программирования / О. А. Щербина // Динамические системы. – 2007. – № 22. — С. 21 – 36.
13. Роновер Р. М. Фракталы и хаос в динамических системах. Основы теории: пер. с англ. / Р. М. Роновер. – М.: Изд-во Постмаркет. – 2000. – 340 с.

УДК 004.724.4(045)

КУЛАКОВ Ю.А.,
КОГАН А.В.

ОПРЕДЕЛЕНИЕ ОПТИМАЛЬНОГО ЧИСЛА ГРАНИЧНЫХ МАРШРУТИЗАТОРОВ ПРИ ОРГАНИЗАЦИИ БЕЗОПАСНОЙ МНОГОПУТЕВОЙ МАРШРУТИЗАЦИИ В МОБИЛЬНОЙ КОМПЬЮТЕРНОЙ СЕТИ БОЛЬШОЙ РАЗМЕРНОСТИ

В работе проведен анализ эффективности использования протокола BGP в мобильной компьютерной сети при разбиении на зоны маршрутизации. Благодаря модифицированному протоколу BGP значительно улучшается процесс маршрутизации. Для оптимизации нахождения множества непересекающихся путей предложен способ нахождения минимального разреза между двумя вершинами.

The paper analyzes the efficiency of BGP in the mobile network in the division into zones routing. Thanks modifying the BGP routing process is significantly improved. To optimize the finding set disjoint paths provides a method for finding the minimum cut between two vertices.

Введение

Для мобильных компьютерных сетей большой размерности характерна динамическая реконфигурация структуры сети и отсутствие централизованного управления. В рамках данного типа сетей, как правило, используется многоуровневая маршрутизация. Существующие протоколы маршрутизации (*RIP, OSPF*) ориентированы на фиксированную структуру сети, поэтому они являются не эффективными для использования в мобильных компьютерных сетях с динамической реконфигурацией. Кроме того, объем служебного трафика, передаваемого по сети, существенно зависит от реконфигурации сети.

В мобильных компьютерных сетях большой размерности используются протоколы децентрализованной многоуровневой динамической маршрутизации, которые предполагают разбиение сети на зоны маршрутизации, используя протокол *BGP (Border Gateway Protocol)* [1, 2, 3] для передачи информации между сетями.

Беспроводные технологии имеют принципиальный недостаток с точки зрения безопасности - доступ к беспроводной среде передачи данных не составляет особого труда. При этом следует отметить, что известные методы повышения безопасности в основном ориентированы на сети с фиксированной структурой.

В связи с этим возникает необходимость в разработке нового подхода к надежной передачи данных в мобильных компьютерных сетях за счет использования многопутевой маршрутизации внутри доменов и между ними. При этом, количество независимых путей между двумя смежными доменами маршрутизации зависит от количества граничных маршрутизаторов между ними.

Обзор существующих решений

Развитие Internet привело к необходимости создания более гибкого и эффективного протокола маршрутизации для обслуживания крупных сетей. Протоколы состояния канала должны решать характерные для протоколов вектора расстояний проблемы. Однако, в отличие от протоколов вектора расстояния, протоколы состояния канала сложны и требовательны к ресурсам маршрутизаторов.

Сеть провайдера переносит трафик между автономными системами, и очень неплохо, когда многие системы могут общаться через *BGP*. Такие системы способны взаимодействовать друг с другом через внутренние соединения *BGP*. Внешние соединения используются для коммуникации между равными друг другу системами, находящимися в различных автономных системах.

Граничный протокол маршрутизации *BGP* является попыткой решить самую серьезную проблему *EGP*. В отличие от *EGP*, протокол *BGP* предназначен для обнаружения маршрутных петель. *BGP* является протоколом маршрутизации между автономными системами, специально созданным для применения в *Internet. BGP* можно назвать следующим поколением *EGP*. Как и *EGP*, протокол *BGP* относится к классу "междоменных протоколов".

Протокол *EGP* имеет ряд существенных недостатков:

1. Маршрутизатор *EGP* представляет только один путь до каждой сети. Это делает невозможным использование процедур динамического перераспределения нагрузки между параллельными каналами;

2. Маршрутизатор *EGP* не поддерживает внеклассовые сети. Целью *BGP* является поддержка маршрутизации через цепочку автономных систем и предотвращение формирования зацикливания. Для этого системы *BGP* обмениваются информацией о путях к сетям, которых они могут достичь. В отличие от *EGP,* протокол *BGP* показывает всю цепочку автономных систем, которые нужно пройти по пути к заданной сети.

Важно отметить, что существенным отличием *BGP* от других протоколов маршрутизации является способность обмена информацией о маршрутизации с хостами, а не только с маршрутизаторами. Возможна конфигурация, в которой хост возьмет на себя всю работу по общению с внешними системами *BGP* в соседних автономных системах. Хост может использоваться как сервер маршрутизации, пересылая информацию граничному серверу собственной автономной системы.

Серьезным недостатком *BGP* является то, что данный протокол является протоколом однопутевой маршрутизации, и не обеспечивает необходимого качества обслуживания.

Одним из эффективных способом повышения безопасности передачи данных в мобильных компьютерных сетях является многопутевая маршрутизация.

В работе [4] рассмотрен пример беспроводной сети, состоящей из узлов и связей, соединяющих их, представленную в виде графа *G=(V,E)*, (рис. 1), где $V=\{v_i \mid i=1\ldots N\}$ множество вершина, $E=\{E_j \mid j=1\ldots M\}$ – множество связей между вершинами. Надежность $Q_{i,j}$ пути $L_{i,j}$ между двумя произвольными вершинами V_i и V_j в общем случае определяется надежностью q_k вершин, входящих в этот путь, и определяется следующим образом:

$$Q_{i,j} = \prod_{q_k \in Li,j} q_k$$

Вероятность q_i изменяется в пределах 1÷ 0 и характеризует уровень защиты *i*-го узла. Максимальной надежности соответствует значение q_i=1.

Рассмотрим путь $L_{S,T}$ (рис. 1.) между вершинами *S* и *T*, включающий множество вершин $\{S, V_3, V_6, V_{10}, T\}$. В данном случае вероятность перехвата сообщения по пути $L_{S,T}$ равна:

$$p_{S,T} = 1 - q_3 q_6 q_{10}.$$

Так как рассматривается безопасность доставки сообщения, то предполагается, что источник и адресат надежны, т.е. $q_S = q_T = 1$.

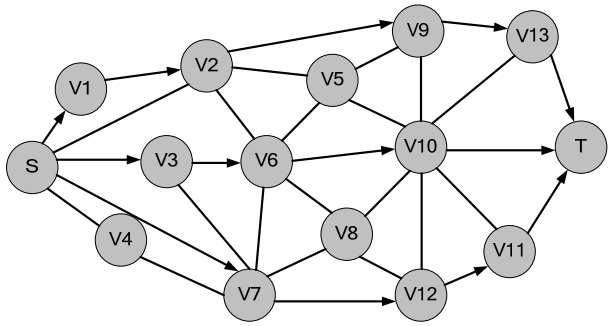


Рис. 1. Граф беспроводной сети

В рамках многопутевой маршрутизации в компьютерной сети необходимо найти оптимальный набор путей при котором вероятность перехвата сообщения P_{msg} была минимальной.

Многопутевая безопасная маршрутизация предполагает разделение сообщения на части с последующей пересылкой частей сообщения по разным путям. При разбиении сообщения на части вероятность перехвата сообщения равна:

$$P_{msg}(n) = \prod_{i=1}^{M} p_i ,$$

где p_i – вероятность перехвата части сообщения. Чем больше частей p_i и больше путей, по которым передаются части сообщения, тем меньше вероятность перехвата всего сообщения и лучше защита. Таким образом, цель алгоритма поиска путей состоит в том, чтобы найти как можно больше путей, которые в то же время будут наиболее безопасными.

Постановка задачи

Эффективность использования существующих способов и алгоритмов маршрутизации в современных компьютерных сетях в существенной степени зависит от структуры и размерности сети. Кроме того, также необходимо учитывать требования интеллектуальных протоколов маршрутизации, обеспечивающих передачу информации с заданными

параметрами качества обслуживания при минимальном объеме служебного трафика независимо от динамической реконфигурации сети.

В связи с мобильностью узлов, топология сети меняется динамически, что делает невозможным использование протоколов централизованной табличной таблиц маршрутизации. Поэтому в мобильных компьютерных сетях большой размерности предлагается использовать протоколы децентрализованной многоуровневой динамической маршрутизации, которые предполагают разбиение сети на домены маршрутизации. В этом случае процедура многопутевой маршрутизации осуществляется отдельно на внутридоменном и междоменном уровнях. При этом, количество независимых маршрутов внутри доменов и между ними зависит от соотношения между количеством доменов и их размерностью

Решение поставленной задачи

При рассмотрении вопросов внутридоменной маршрутизации следует учитывать тот факт, что автономная система очень уязвима к атакам. Следовательно, процесс маршрутизации не будет безопасным. В связи с этим в работах [9, 10, 11, 12] предложен способ нахождения множества непересекающихся путей, которые повышают уровень безопасности при передачи информации внутри автономной системы.

При рассмотрении вопросов междоменной многопутевой маршрутизации одним из основных вопросов является разбиение сети на домены маршрутизации и определение оптимального числа граничных маршрутизаторов.

На рис. 2 при пересечении двух подсетей образуется одни общий узел v_6, который будем считать граничным маршрутизатором *BGP*. В общем случае использование одного граничного маршрутизатора не эффективно, потому что все маршруты между доменами должны будут проходить через один граничный маршрутизатор, а это ставит под угрозу безопасность процесса маршрутизации. В связи с этим предлагается формировать домены маршрутизации таким образом, чтобы количество общих граничных маршрутизаторов равнялось количеству независимых маршрутов для передачи информации.

На рис. 3 представлено разделение сети на два домена маршрутизации с тремя граничными маршрутизаторами, каждый из которых располагается на одном из независимых путей между доменами. Благодаря такому подходу балансируется нагрузка на сеть и повышается безопасность передачи информации.

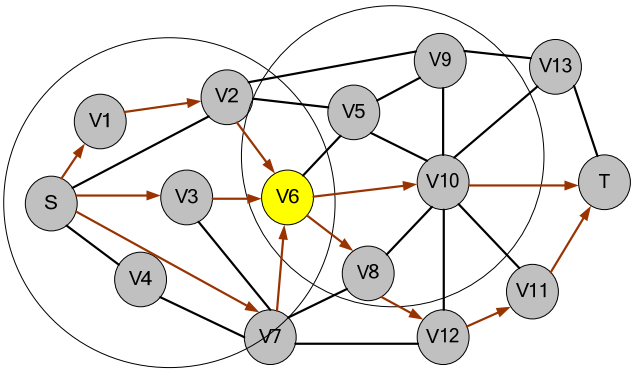


Рис. 2. Пересечение подсетей с одним общим узлом

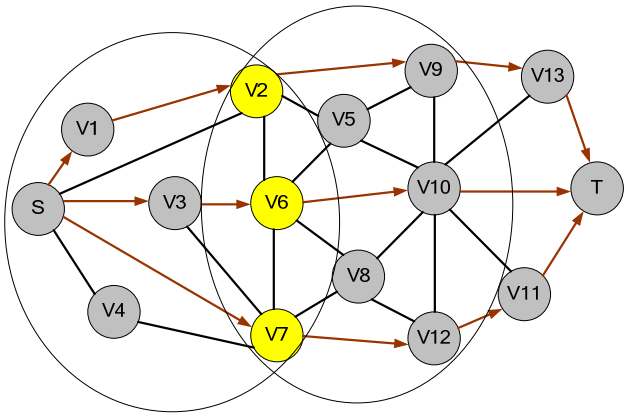


Рис. 3. Пересечение подсетей с несколькими общими узлами

В работах [5, 6] для разбиения сети на домены маршрутизации предлагается, использовать протокол *ZRP*. В результате пересечения доменов маршрутизации образуются общие узлы, которые принадлежат нескольким подсетям одновременно.

Общим правилом является использование максимально возможного числа непересекающихся маршрутов. Это повышает надежность и способствует перераспределению нагрузки между каналами.

Определение максимально возможного числа непересекающегося маршрутов между отправителем и получателем информации можно свести к задаче нахождения минимального разреза между двумя вершинами. Под минимальным разрезом понимается минимальное подмножество вершин графа, удаление которых приводит к разделению графа на два подграфа. В теории графов [7] такое минимальное подмножество вершин принято называть подмножеством сочленения связного неориентированного графа. В работе [8] предложен способ определения минимального подмножества сочленения связного неори-

ентированного графа. Мощность данного подмножества определяет максимальное количество число непересекающихся путей между двумя вершинами, расположенными в смежных подграфах. Применительно к данной задаче способ определения минимального подмножества сочленения заключается в следующем. Происходит разбиения графа G на два подграфа, первый подграф G_S содержит вершину S отправителя информации, а второй G_T включает вершину T получателя информации (рис.1). На начальном этапе множество вершин V_S подграфа G_S включает только одну вершину S, т.е. $V_S = \{S\}$. Затем формируется минимальное подмножество сочленения $V_{\min}$ из вершин, связанных с вершинами подмножества V_S, то есть с вершиной S. В данном случае такими вершинами являются: v_1, v_2, v_3, v_4, v_7. Сформируем новый подграф G_S с множеством вершин $V_S = \{S, v_1, v_2, v_3, v_4, v_7\}$. Вершины S, v_1, v_4 являются внутренними вершинам подграфа G_S , так как не связаны с вершинами подграфа G_T. Вершины v_2, v_3, v_4, v_7 подграфа G_S связаны с вершинами подграфа G_T и образуют минимальное подмножество сочленения $V_{\min} = \{v_2, v_3, v_7\}$ между подграфами G_S и G_T. Процесс формирования минимального подмножества сочленения продолжается пока в подграфе G_T не останется только одна конечная вершина T. При этом в зависимости от топологии графа и среднего значения степени его вершин в графе может существовать несколько минимальных разрезов между двумя произвольными вершинами, например, для графа, представленного на рис. 2, между вершинами S и T существует 4 минимальных разреза графа (рис.4): $C_1 = \{v_2, v_3, v_7\}$; $C_2 = \{v_2, v_6, v_7\}$; $C_3 = \{v_9, v_{10}, v_{12}\}$; $C_4 = \{v_{10}, v_{11}, v_{13}\}$.

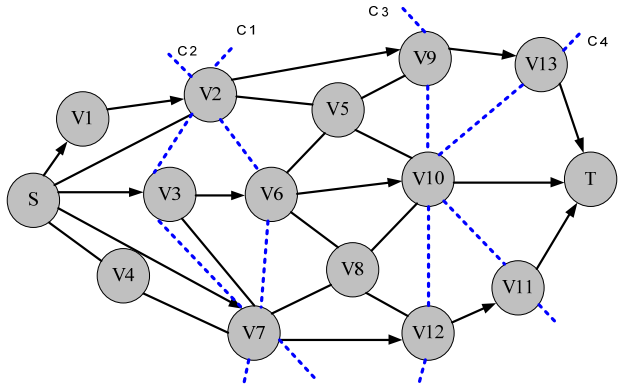


Рис. 4. Минимальные разрезы графа

Каждой вершине $v_i \in V_{\min}$ соотносится граничный маршрутизатор протокола *BGP* с помощью которых осуществляется междоменная маршрутизация.

На рис. 5 представлен алгоритм нахождения граничных маршрутизаторов и определения независимых маршрутов.

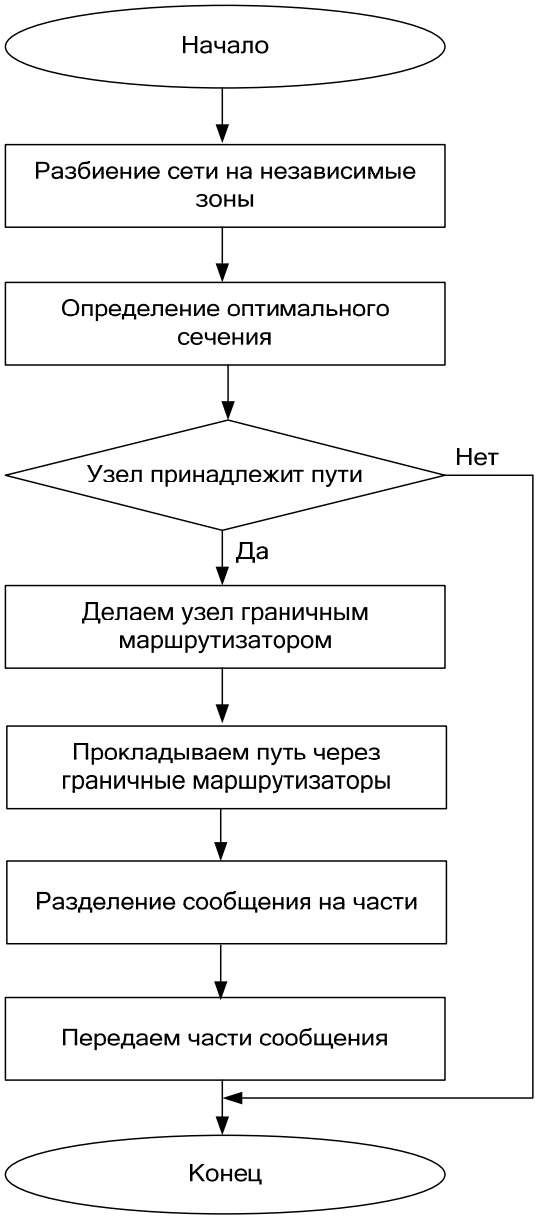


Рис. 5. Алгоритм безопасной междоменной маршрутизации

Выводы

В работе предложен способ формирования структуры мобильной компьютерной сети с оптимальным числом граничных маршрутизаторов, обеспечивающих максимальное количество независимых путей как внутри, так и между доменами. Предложенный в работе алгоритм междоменной маршрутизации позволяет повысить уровень безопасности передаваемой информации и обеспечивает равномерную загрузку мобильной сети.

Список литературы

1. Stephen Kent. Secure Border Gateway Protocol (S-BGP) – Real World Performance and Deployment Issues/ Stephen Kent, Charles Lynn, Joanne Mikkelson, and Karen Seo// BBN Technologies: Appeared in Proceedings of the Network and Distributed System Security Symposium (NDSS 2000), February 2000. – San Diego, California, 2000. – p.1-14.
2. Network Configuration Example Advertising Multiple Paths in BGP: [Электронный ресурс]. – October 2011. – P.1-32. – Режим доступу до журналу: http://www.juniper.net/support/eula.html.
3. Game Theoretic Stochastic Routing for Fault Tolerance and Security in Computer Networks// Stephan Bohacek João Hespanha, Junsoo Lee, Chansook Lim, Katia Obraczka. – 2011. – С. 1-30.
4. Кулаков Ю.А. Безопасная многопутевая маршрутизация в беспроводных сетях большой размерности / Ю.А. Кулаков, В.В. Лукашенко, А.В. Левчук // Защита информации. – 2011. – № 2 (51). – С. 120-126.
5. Левчук А.В. Анализ эффективности алгоритмов разбиения мобильной компьютерной сети на зоны маршрутизации / Левчук А.. В. // Проблеми інформатизації та управління: зб. наук. праць. – К.: Вид. нац. авіац. ун-ту «НАУ-друк», 2010. – № 1(29). – С. 118-121.
6. Лукашенко В.В. Анализ эффективности способов разбиения сети на зоны маршрутизации / В.В. Лукашенко, А.В. Левчук // Електроніка та системи управління. – 2010. – № 1(23). – С. 112-118.
7. А. Кофман. Введение в прикладную комбинаторику. – М.: «Наука». – 1975.
8. Кулаков Ю.А. Определение минимального подмножества сочленения структуры компьютерной сети / Кулаков Ю.А., Эль-Хуссейн Касам Мустафа // Вісник національного технічного університету України «КПІ», Інформатика, управління та обчислювальна техніка. – 1998. – № 31. – С. 128-134.
9. Кулаков Ю.А. Многопутевая маршрутизация в беспроводных сетях / Кулаков Ю.А., А.В. Левчук // Електроніка та системи управління. – 2010. – № 4(26). – С. 142-147.
10. T. Nirmal Raj. Secured Multi Path Routing with Trust Establishment Using Mobile Ad Hoc Networks / T. Nirmal Raj, S. Saranya, S. Arul Murugan, G. Bhuvaneswari // International Journal of Scientific & Engineering Research. – 2012. – Volume 3, Issue 1. – P. 1-5.
11. Haseeb Zafar. QoS-aware Multipath Routing Scheme for Mobile Ad Hoc Networks / Haseeb Zafar, David Harle, Ivan Andonovic, Laiq Hasan, Amjad Khattak // International Journal of Communication Networks and Information Security (IJCNIS). – 2012. – Vol. 4, No. 1. – P. 1-10.
12. Madhusudan G. Novel Technique of Multipath Routing Protocol in Ad hoc Network / Madhusudan G., D.S.Vinod // International Journal of Computer Networks & Communications (IJCNC). – 2012. – Vol.4, No.3. – P. 109-119.

УДК 681.324

РОЛИК А.И.

ДЕКОМПОЗИЦИОННО-КОМПЕНСАЦИОННЫЙ ПОДХОД К УПРАВЛЕНИЮ УРОВНЕМ УСЛУГ В КОРПОРАТИВНЫХ ИТ-ИНФРАСТРУКТУРАХ

Предложен декомпозиционно-компенсационный подход к управлению уровнем ИТ-услуг в корпоративных ИТ-инфраструктурах, предполагающий декомпозицию задач управления уровнем услуг и компенсацию негативного влияния различных факторов выделением дополнительных ресурсов критичным приложениям. Подход основан на интегрированном взаимодействии трех иерархических процессов — согласования уровня услуг, планирования ресурсов и управления уровнем услуг с учетом иерархии ИТ-инфраструктуры.

Decomposition-compensation method of service level management of corporate IT infrastructures was proposed. The method based on the interaction of three integrated hierarchical processes — matching the level of services, resource planning and service level management.

Введение

Бизнес рассматривает информационные технологии (ИТ) в качестве средства повышения своей производительности и улучшения конкурентоспособности. Эффективность выполнения бизнес-процессов существенно зависит от качества ИТ-услуг. Увеличивающееся количество ИТ-услуг, необходимых для автоматизации бизнес-технологий, усложнение приложений и возрастающее количество компонентов ИТ-инфраструктуры приводят к снижению эффективности работы ИТ-подразделений и повышению затрат на поддержание штатного режима функционирования ИТ-инфраструктуры. Обслуживание ИТ-услуг, внесенных в каталог, осуществляет ИТ-подразделение. Предоставление ИТ-услуг регламентируется пакетом соглашений об уровне сервиса (SLA), заключенных между бизнес-подразделениями и ИТ-подразделением. В SLA определяются значения ключевых показателей эффективности (KPI) и качества (KQI) — ограниченный набор объективно измеряемых параметров, позволяющий оценить качество ИТ-услуг [1]. Для поддержания значений KPI и KQI на уровне, зафиксированном в SLA, администраторы обеспечивают непрерывное функционирование ИТ-инфраструктуры, осуществляют обслуживание и ремонт с использованием методов автоматического, автоматизированного и ручного управления

Для повышения производительности ИТ-инфраструктуры и автоматизации процессов обслуживания создаются ИТ управления ИТ-инфраструктурой (УИТИ), на основе которых разрабатываются системы управления (СУ) ИТ-инфраструктурой [2, 3]. Увеличение бизнес-спроса на ИТ-услуги, многообразие ИТ, а также постоянное совершенствование бизнес-процессов и вызванная этим необходимость разработки и внедрения новых ИТ приводят к чрезмерному усложнению СУ, являющейся продуктом системной интеграции многообразных подходов и порой несовместимых решений от различных производителей. Возрастающая сложность управления ИТ-инфраструктурой, сопровождающаяся увеличением затрат на операционную деятельность, вызывает необходимость поиска новых подходов к управлению ИТ-инфраструктурой.

В настоящее время развитие ИТ находится под сильным влиянием таких факторов, как консолидация и виртуализация ресурсов, прогресс в области облачных вычислений, конвергенция телекоммуникационных технологий и услуг. В информационной индустрии под воздействием этих факторов осуществляется активный процесс глобализации информационных и телекоммуникационных технологий, формируется новая ИТ-среда, предоставляющая перспективные средства ведения бизнеса. Революционные преобразования в области ИТ способствуют прогрессу бизнес-технологий, однако эффективность применения ИТ сдерживает отставание в области технологий управления ИТ-инфраструктурой.

Высокая стоимость владения полнофункциональной ИТ-инфраструктурой корпоративного уровня и существенная зависимость успешности бизнеса от качества ИТ-услуг делают актуальной научно-прикладную проблему создания информационной технологии управления кор-

поративной ИТ-инфраструктурой, нацеленной на поддержание согласованного уровня ИТ-услуг при рациональном использовании ресурсов в условиях их виртуализации, кластеризации и консолидации при существенной динамике запросов пользователей.

Постановка проблемы в общем виде

Существенное влияние ИТ на достижение целей бизнеса не только подчеркивает значимость ИТ-сервисов, но и акцентирует необходимость управления ими. Лидирующее положение в области управления ИТ-сервисами принадлежит ITSM [4, 5] — признанному во всем мире и повсеместно реализуемому подходу к управлению ИТ, о чем свидетельствует появление международного стандарта ISO/IEC 20000 [6, 7]. Стандарт ISO/IEC 20000, являясь первым международным стандартом управления ИТ-услугами, содержит: требования к руководству, документированию, компетенции, осведомленности и подготовке персонала; требования к мониторингу, измерению, оценке и улучшению процессов; принципы формирования плана управления услугами и порядка применения цикла Деминга к управлению ИТ-услугами. Стандарт внес изменения и дополнения в процессную модель, перейдя от набора процессов к созданию целостной системы управления ИТ-услугами. Выделяется 13 процессов, разделенных на пять групп, и две области управления верхнего уровня.

Требования к управлению ИТ-услугами и к системе управления определены в ISO/IEC 20000-1, а рекомендации по организации деятельности по управлению ИТ-услугами — в ISO/IEC 20000-2. В соответствии со стандартом СУ, включающая политики и структурированный подход, должна реализовывать внедрение и эффективное управление всеми ИТ-услугами. При этом рассматривается только процессное управление, а вопросы оперативного управления ИТ-инфраструктурой, состояние и функционирование которой оказывает самое непосредственное влияние на уровень услуг, не рассматриваются.

Среди процессов ITSM и ISO/IEC 20000 отсутствуют процессы и область оперативного управления уровнем ИТ-услуг и ИТ-инфраструктурой. Это объясняется тем, что управление ИТ-услугами — сравнительно новая и активно развивающаяся область управления. Возможность и необходимость управления ИТ-услугами уже признана, о чем свидетельствует появление стандарта, в то время как рассмотрение ИТ-инфраструктуры в качестве объекта управления при менеджменте ИТ-услуг до сих пор в полной мере не осознано.

В то же время рассмотрение ИТ-инфраструктуры как объекта управления для поддержания качества предоставления ИТ-услуг на приемлемом уровне при изменении состояния компонентов ИТ-инфраструктуры и с учетом динамики запросов пользователей не только возможно, но и необходимо.

Таким образом, вопросы процессного управления уровнем услуг хорошо проработаны и стандартизованы, а вопросам управления уровнем услуг посредством оперативного управления ИТ-инфраструктурой до сих пор не уделялось достаточного внимания. Поэтому в данной статье, учитывая важность для бизнеса получения без перебоев ИТ-услуг со стабильно высоким качеством, внимание уделено именно этому аспекту управления ИТ-инфраструктурой.

Целью данной работы является разработка такого подхода к управлению ИТ-инфраструктурой, чтобы качество предоставляемых ею ИТ-услуг соответствовало заданному, согласованному с бизнес-подразделением, уровню, при рациональном использовании ресурсов.

Постановка задачи управления ИТ-инфраструктурой при управлении уровнем услуг

Для такого объекта управления, как ИТ-инфраструктура, очень сложно определить и формализовать единую задачу управления. Поэтому производится декомпозиция общей задачи управления ИТ-инфраструктурой [8], заключающейся в выборе такого допустимого управления, которое максимизирует значение эффективности управления $(\mathrm{K}(U) \to \max\limits_{U \in \mathcal{U}})$, на отдельные задачи с последующей их формализацией.

ИТ-инфраструктура предназначена для предоставления пользователям ИТ-сервисов. Эффективность управления в этом случае может быть оценена по качеству Q предоставляемых сервисов и затратам на управление. При оперативном управлении ИТ-инфраструктурой задача управления качеством сервисов состоит в поддержании заданного уровня качества сервисов с минимальным количеством используе-

мых для этого ресурсов. Тогда максимальная эффективность управления достигается выбором такого управления, при котором фактический уровень сервисов соответствует согласованному с бизнес-подразделением Q_{cor} и достигается с минимальными затратами:

$$\text{Затраты}(Q(U) \to Q_{\text{cor}}) \to \min_{U \in \mathcal{U}}. \quad (5)$$

Качество Q сервисов определяется качеством $Q_j, j = \overline{1, N}$ всех ИТ-услуг:

$$Q = f(Q_1, ..., Q_N), \quad (6)$$

следовательно, управляющие воздействия должны поддерживать заданный уровень качества каждой услуги с использованием для этих целей минимального количества ресурсов:

$$\text{Затраты}(Q_j(U) \to Q_j^{\text{cor}}) \to \min_{U \in \mathcal{U}}, \quad \forall j = \overline{1, N}, \quad (7)$$

где $Q_j^{\text{cor}}, j = \overline{1, N}$ — согласованный уровень j-й услуги.

В отличие от процессного управления, перед которым стоит задача постоянного повышения качества услуг, оперативное управление нацелено на поддержание наиболее дешевым способом качества услуг на согласованном уровне, при этом управление должно быть таким, чтобы обеспечивалось

$$q_{k,j} - q_{k,j}^{*} \to 0, \forall j, k, \quad (8)$$

где $q_{k,j}$ и $q_{k,j}^{*}$ — соответственно, целевое и фактическое значения k-го показателя качества j-й услуги.

С точки зрения бизнеса критерием эффективности управления ИТ-инфраструктурой при обеспечении качества j-й услуги может быть выбор такого управления $U \in \mathcal{U}$, при котором достигается минимальное фактическое время обработки i-го запроса к приложению $A_j, j = \overline{1, N}$, N — количество приложений

$$\min_{\forall i, j}(T_{\Phi_{i,j}} = (t_{R_{i,j}} - t_{A_{i,j}})), \quad (9)$$

где $t_{A_{i,j}}$ — время поступления i-го запроса от пользователя j-й услуги, $t_{R_{i,j}}$ — время поступления ответа пользователю на i-й запрос к приложению A_j.

Что касается качества предоставления услуг, то ИТ УИТИ должна осуществлять управление, руководствуясь не критерием (9), поскольку невозможно достичь времени обработки запросов, равного 0, а стремиться минимизировать разницу между заданным $T_{З_j}$ и фактическим T_{Φ_j} временем выполнения запросов к j-му приложению, измеренным на стороне пользователя, используя наименьшее количество ресурсов

$$\min_{\forall j}(T_{З_j} - T_{\Phi_j}) = \min_{\forall j}(\Delta T_j), \text{ при } T_{\Phi_j} > T_{З_j}. \quad (10)$$

В случае, когда $T_{\Phi_j} - T_{З_j} > 0$, причиной чему может быть увеличение количества запросов пользователей, выполнение критерия (9) или (10) возможно путем выделения приложению A_j дополнительных информационно-вычислительных ресурсов и/или приоритетного прохождения данных пользователей приложения A_j по телекоммуникационной сети.

Подход к оперативному управлению уровнем ИТ-услуг

Суть предлагаемого подхода к оперативному управлению уровнем услуг. Основная цель бизнеса — получение максимальной прибыли. Максимальная прибыль за счет ИТ достигается тогда, когда бизнесу предоставляется множество $S = \{s_i\}$, $i = \overline{1, K}$ необходимых ИТ-услуг с максимальным качеством Q и минимальными затратами C.

Управление уровнем услуг в корпоративных ИТ-инфраструктурах предлагается осуществлять интегрированным взаимодействием трех процессов: согласования уровня услуг, планирования ресурсов и управления уровнем услуг (рис. 1).

Процесс согласования уровня услуг запускается по инициативе бизнес-менеджеров и заканчивается формированием или обновлением элементов множества S и матрицы $Q = \|q_{ki}\|$, элемент q_{ki}, $k = \overline{1, M_i}$, $i = \overline{1, K}$ которой соответствует согласованному значению k-го показателя качества i-й услуги. Бизнес выделяет услугам S количество r ресурсов. Полученное в результате ресурсное обеспечение в виде системы

$$< Q, r > \quad (11)$$

является основой для решения задач на нижерасположенном уровне.

Процесс планирования заключается в выделении и закреплении за каждой услугой s_i, $i = \overline{1, K}$ части из ресурсов $R_1,...,R_m$ ИТ-инфраструктуры, выделенных для поддержания услуг, при этом $r_1,...,r_m$ — количество ресурсов $R_1,...,R_m$, а $c_1,...,c_m$ — стоимость единицы ресурса $R_1,...,R_m$ соответственно.

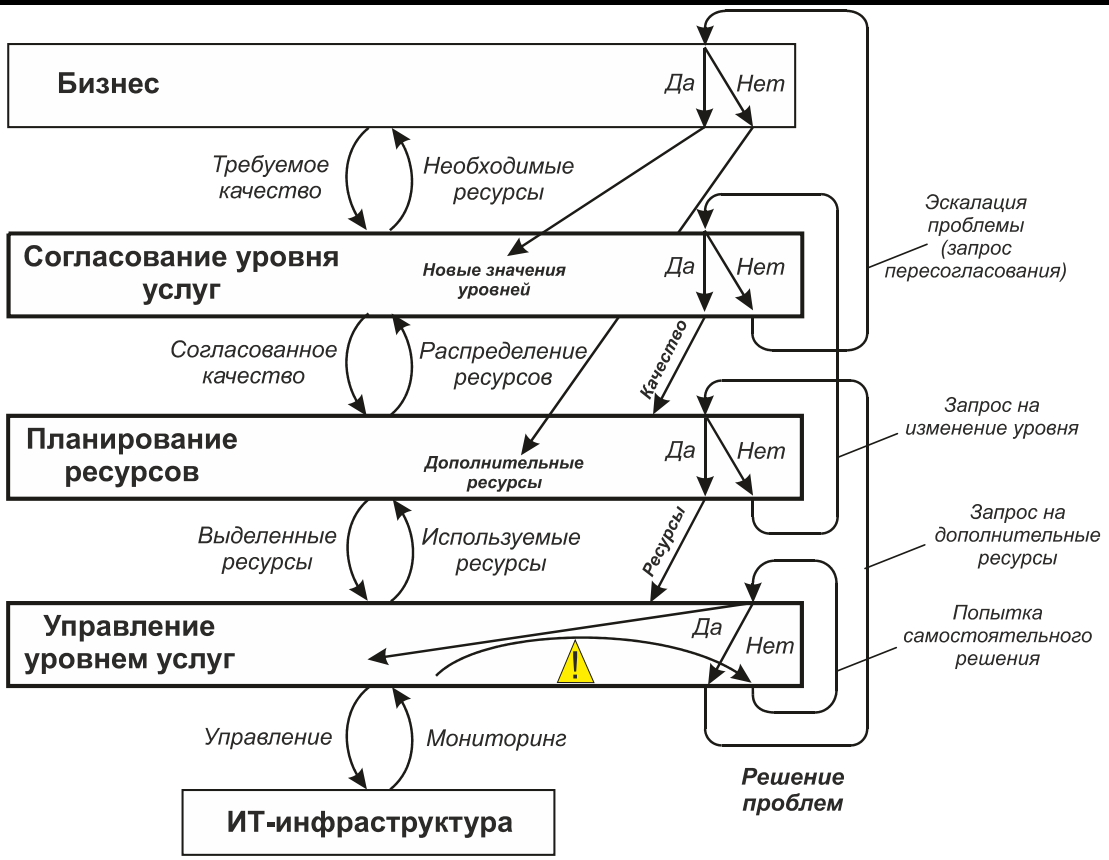


Рис. 1. Взаимодействие процессов при управлении уровнем услуг

Тогда количество всех ресурсов вычисляется как:

$$r = \sum_{j=1}^{m} r_j \,, \qquad (12)$$

а стоимость c ресурсов определяется следующим образом:

$$c = \sum_{j=1}^{m} r_j \cdot c_j \,. \qquad (13)$$

Использование услугами ресурсов задается матрицей $\mathrm{P} = \left\| \rho_{ij} \right\|$, где ρ_{ij} равно количеству выделенного услуге s_i ресурса R_j, $j = \overline{1,m}$ или 0, если ресурс не требуется.

Процесс управления уровнем услуг осуществляет управление ИТ-инфраструктурой так, чтобы фактические значения q_{ki}^*, $k = \overline{1,M_i}$, $i = \overline{1,K}$ показателей качества услуг соответствовали согласованным значениям из матрицы Q, т. е. чтобы выполнялось равенство

$$q_{ki} - q_{ki}^* = 0, \quad \forall k = \overline{1,M_i}, i = \overline{1,K} \,. \qquad (14)$$

Суть предлагаемого подхода к управлению уровнем услуг заключается в следующем.

При невыполнении условия (14) выявляются элементы матрицы фактических значений показателей качества $Q^* = \left\| q_{ki}^* \right\|$, для которых $q_{ki}^* < q_{ki}, \forall k = \overline{1,M_i}, i = \overline{1,K}$. ИТ УИТИ пытается решить проблему на нижнем уровне (см. рис. 1), изменяя значения параметров функционирования элементов ИТ-инфраструктуры либо перераспределяя ресурсы между приложениями так, чтобы увеличить значение q_{ki}^*. Если в результате восстановительных мероприятий удалось обеспечить выполнение равенства (14), то функционирование ИТ-инфраструктуры продолжается с новыми настройками. Если полномочий нижнего уровня недостаточно для достижения (14), то осуществляется эскалация проблемы на уровень планирования ресурсов.

В ходе процесса планирования ресурсов осуществляются попытки решить проблему выделением дополнительных ресурсов $R_1,...,R_m$ услуге s_i, для которой выполняется условие $q_{ki}^* < q_{ki}$. Если дополнительные ресурсы выделяются, то формируется матрица $\mathrm{P}' = \left\| \rho'_{ij} \right\|$ с новыми значениями элементов, причем $\rho'_{ij} > \rho_{ij}$, $j = \overline{1,m}$ или 0, если j-й ресурс не требуется. Если дополнительные ресурсы отсутствуют, то на уровне планирования ресурсов предпринимаются попытки произвести перераспределение ресурсов между услугами, отда-

вая ресурсы более важным услугам за счет менее важных. Если проблема решается, то значения матрицы $P' = \|\rho'_{ij}\|$ с новым планом закрепления ресурсов поступают на нижний уровень. При невозможности разрешения проблемы на уровне планирования ресурсов производится эскалация проблемы на вышерасположенный уровень.

Процесс согласования уровня услуг по инициативе процесса планирования осуществляет пересмотр сначала значения q_{ki}, для которого $q^*_{ki} < q_{ki}$, а затем, возможно, и значений всех элементов q_{ki}, $k = \overline{1, M_i}$, $i = \overline{1, K}$ матрицы качества услуг Q в сторону уменьшения. Если удается сформировать матрицу $Q' = \|q'_{ki}\|$ с новыми значениями показателей качества услуг, то она передается на уровень ниже, где производится высвобождение ресурсов и выделение их услугам, для которых выполняется условие $q^*_{ki} < q_{ki}$, $k = \overline{1, M_i}$, $i = \overline{1, K}$. Если процесс согласования уровня услуг не имеет полномочий для формирования матрицы $Q' = \|q'_{ki}\|$, то производится эскалация проблемы на уровень бизнеса, который вынужден либо сгенерировать матрицу $Q' = \|q'_{ki}\|$ с новыми значениями, либо увеличить общий объем ресурсов, что приводит к увеличению значений $r_1,...,r_m$, либо довольствоваться фактическим уровнем услуг.

Рассмотрим подробнее процессы, реализуемые при согласовании уровня услуг, планировании ресурсов и управлении уровнем услуг.

Процесс согласования уровня услуг в корпоративных ИТ-инфраструктурах. Требования максимизации ($\max Q$) качества услуг и минимизации связанных с этим затрат ($\min C$) находятся в естественном противоречии, что приводит к необходимости установления экономически обоснованного уровня услуг с учетом как возможностей компании, так и достигнутого отраслью уровня и ожиданий клиентов.

Если $\hat{D}$ — доходы бизнеса от ИТ, а $\hat{C}$ — затраты на ИТ-инфраструктуру, то бизнес совместно с процессом согласования уровня услуг пытается добиться

$$\max(\hat{D} - \hat{C}). \qquad (15)$$

Стремясь к постоянному повышению качества, бизнес- и ИТ-подразделения должны в рамках процессного управления ITSM выполнять аналитическую оценку зависимостей качества сервиса от стоимости ресурсов $\hat{Q} = f_1(c)$, потерь бизнеса $\hat{L}$ от некачественного сервиса с учетом рисков M_0 и неопределенностей N_0 (рис. 2), после чего с учетом уровня качества, достигнутого отраслью, определяются допустимые значения уровня услуг $Q = \|q_{ki}\|$, которые еще не приводят к потерям при выполнении бизнес-операций. Таким образом, система

$$< \hat{D}, \hat{C}, \hat{L}, \hat{Q}, M_o, N_0 > \qquad (16)$$

определяет задачу по формированию двойки (11).

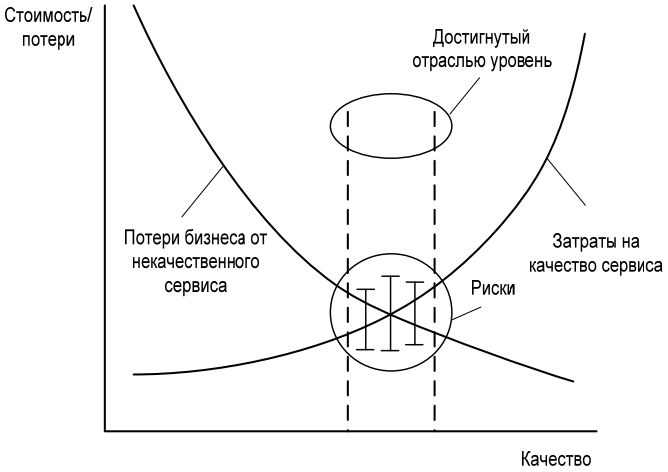


Рис. 2. Нахождение оптимального соотношения уровня качества сервиса и стоимости достижения этого качества

Если полученные с учетом рисков значения уровня услуг превышают достигнутый отраслью уровень, то есть повод для оптимизма и развития бизнеса. Иное говорит о проблемах и является основой для сворачивания бизнеса в соответствующих направлениях.

Бизнесу предоставляются варианты реализации услуги со стоимостными, техническими, организационными и временными показателями (рис. 3). Бизнес определяет оптимальное отношение доходность/(качество ИТ-услуг). Для этого оцениваются потери в зависимости от падения уровня услуг и стоимость предоставления услуг высокого качества, после чего находится точка минимально допустимого качества. Затем бизнес одобряет один из вариантов или корректирует требования к новой ИТ-услуге, исходя из значимости сервиса, бенчмаркинга, опыта и пр. Согласованные уровни услуг фиксируются в SLA или каталоге услуг.

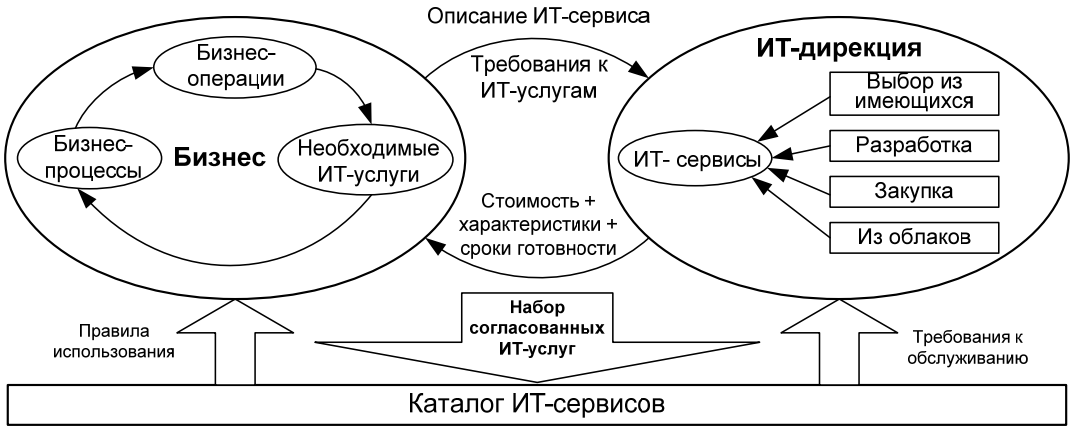


Рис. 3. Процесс формирования каталога ИТ-сервисов

План внедрения новых услуг ИТ-дирекция готовит с учетом финансового и ресурсного обеспечения их предоставления и управления. Эту задачу целесообразно решать с применением методов эффективного использования ресурсов ИТ-инфраструктуры, предложенных в [9—11].

Согласование уровня услуг производится в условиях избытка или дефицита ресурсов (см. ниже). При дефиците ресурсов запускается итеративная процедура согласования уровня услуг, когда на основании (12) процесс планирования определяет значения q_{ki}, $k=\overline{1,M_i}$, $i=\overline{1,K}$, затем полученные показатели в виде матрицы Q предъявляются бизнесу, и если значения элементов матрицы Q не удовлетворяют бизнес, то производится перераспределение ресурсов между услугами и запускается новый цикл определения значений q_{ki}, $k=\overline{1,M_i}$, когда для заданного плана закрепления ресурсов $R_1,...,R_m$ за услугами s_i, $i=\overline{1,K}$ определяется ожидаемое качество Q.

При решении задач согласования уровня услуг могут быть использованы методы математического программирования, исследования операций, теории принятия решений, методы теории игр, методы решения многокритериальных задач, принятия решений в условиях неопределенности и риска, методы искусственного интеллекта, балансовые модели и др.

Процесс планирования ресурсов. На уровне планирования ресурсов решаются задачи в интересах процесса согласования уровня услуг, производится определение необходимого количества ресурсов, распределение и закрепление ресурсов за услугами s_i, $i=\overline{1,K}$.

Услуги s_i, $i=\overline{1,K}$ поддерживаются приложениями из множества $A=\{A_l\}$, $l=\overline{1,I}$, где I — количество приложений, причем каждая услуга s_i, $i=\overline{1,K}$ поддерживается одним или несколькими приложениями, в то время, как каждое приложение A_l, $l=\overline{1,I}$ поддерживает одну или несколько услуг.

Введем понятие степени поддержки услуги. Под поддержкой услуги s_i будем понимать взаимодействие нескольких приложений из множества A, обеспеченных ресурсами, направленное на достижение единого результата — работоспособность услуги s_i. Пусть ресурсы R_j, $j=\overline{1,m}$ используются для поддержки услуг s_i, $i=\overline{1,K}$. Сначала введем в рассмотрение булевую переменную x_i, $i=\overline{1,K}$, определяющую степень поддержки i-й услуги и принимающую значения:

$$x_i=\begin{cases}1, \text{если } i\text{-я услуга поддерживается в полном объеме;}\\ 0, \text{если } i\text{-я услуга не поддерживается.}\end{cases} \tag{17}$$

Пусть пользователи услуги s_i формируют в среднем a_{li} запросов к приложению A_l за единицу времени. Тогда количество клиентских запросов приложения A_l определяется следующим образом:

$$a_l=\sum_{i=1}^{K} a_{li}\cdot x_i\ , \tag{18}$$

а количество запросов к приложениям A_l, $l=\overline{1,I}$ представим вектором $\hat{a}=\{a_l, l=\overline{1,I}\}$.

Потребности приложения A_l в ресурсах типа j определяются выражением:

$$r_{jl} = b_{jl}\, a_l + d_{jl}, \qquad (19)$$

где b_{jl} — среднее количество ресурсов типа j, используемых приложением A_l для обработки одного клиентского запроса; d_{jl} — количество ресурсов типа j, используемых приложением A_l независимо от количества клиентских запросов.

Тогда общее количество ресурсов r, необходимое для поддержания всех услуг из S, определяется с помощью выражения:

$$r = \sum_{j=1}^{m}\sum_{l=1}^{I} r_{jl} = \sum_{j=1}^{m}\sum_{l=1}^{I}(b_{jl}a_l + d_{jl}). \qquad (20)$$

Задачи планирования существенно зависят от ограничений на бюджет ИТ-подразделения. При отсутствии финансовых ограничений, когда критерий $\min C$ не учитывается, планируется и проектируется ИТ-инфраструктура, ресурсы r которой будут достаточными для поддержания приложений $\{A_l\}$ с требуемыми значениями показателей качества услуг q_{ki}, $k = \overline{1, M_i}$, $i = \overline{1, K}$ при максимальных допустимых значениях вектора $\hat{a}$. Кроме того, наиболее важные ресурсы дублируются или резервируются $(r_r > 0)$, а также учитывается возможное запредельное увеличение количества запросов a_{li}.

При ограниченном бюджете на создание и развитие ИТ-инфраструктуры дефицит ресурсов может закладываться уже на стадии проектирования или возникать в процессе эксплуатации, и сложившаяся ИТ-инфраструктура в принципе не может обеспечить выполнение условия (14). Несмотря на это, ИТ-инфраструктура, спроектированная с дефицитом ресурсов, способна эффективно предоставлять услуги, если априорно обеспечивается поддержка наиболее важных приложений. Для этого ИТ УИТИ перераспределяет ресурсы в процессе работы в соответствии с определенным регламентом использования ресурсов. Решению подобных задач посвящена работа [12].

Таким образом, задачи планирования и управления ресурсами необходимо решать в условиях как избытка, так и дефицита ресурсов. В каждом конкретном случае необходимо учесть некоторые специфические особенности распределения ресурсов.

После расчета потребности ресурса по выражению (20) и сравнения ее с имеющимися ресурсами получим задачу управления уровнем сервисов при дефиците ресурсов, если нештатные ситуации в ИТ-инфраструктуре, увеличение интенсивности клиентских запросов $\hat{a}$ и другие факторы приводят к невыполнению равенства (14).

В этом случае для принятия решений при распределении ресурсов необходимо учитывать дополнительную информацию.

Введем понятие важности w_i услуги s_i, $i = \overline{1, K}$, которое будем использовать при решении задачи согласования уровня услуг в условиях дефицита ресурсов.

Задача согласования уровня услуг при дефиците ресурсов сводится к определению значений матрицы Q при заданном количестве ресурсов:

$$Q = \mathcal{F}_1(S, r, W_p, Z_s), \qquad (21)$$

где $W_p = \{w_i \mid i = \overline{1, K}\}$; $Z_s = \{z_i \mid i = \overline{1, K}\}$; z_i — планируемая степень поддержки i-й услуги. Здесь нормативное значение количества клиентских запросов учитывается в величине r, а значение $z_i, i = \overline{1, K}$, в отличие от (17), является непрерывной переменной, принимающей значения из отрезка [0,1].

Если же после расчета потребности в ресурсах и ее сравнения с имеющимися ресурсами, последних оказывается больше, то получим задачу управления уровнем услуг при избытке ресурсов. В частности, может быть выделен резервный объем r_r ресурсов, и количество ресурсов $\hat{r}$, выделяемое для поддержания всех услуг из S, определяемое по выражению

$$\hat{r} = r + r_r, \qquad (22)$$

возрастает. Тогда объем ресурсов r определяется, исходя из значений элементов матрицы Q, а величина r_r — исходя из вероятности возникновения нештатных ситуаций и предельного значения величины a_{li}.

В этом случае выделяются две задачи согласования уровня услуг. Задача первого рода аналогична задаче (21) и состоит в определении значений показателей качества при известном объеме выделенных ресурсов:

$$Q = \mathcal{F}_2(S, \hat{r}, r_r, C). \qquad (23)$$

Задача второго рода состоит в определении необходимого объема ресурсов для обеспечения заданных значений показателей качества:

$$r = \mathcal{F}_3(S, Q, C). \qquad (24)$$

При решении задач (21), (23) и (24) могут быть использованы методы теории массового

обслуживания [13], теория надежности [14], теории фракталов, аналитическое и имитационное моделирование, в частности, с применением методов теории массового обслуживания и искусственного интеллекта.

Процесс управления уровнем услуг. После согласования уровня услуг и планирования ресурсов процесс управления осуществляется так, чтобы выполнялись критерии:

$$\min(q_{ki} - q_{ki}^*),\ k = \overline{1, M_i}, i = \overline{1, K}, \text{ при } q_{ki}^* < q_{ki} \quad (25)$$

или

$$\min \hat{C},\ \text{при } q_{ki}^* > q_{ki},\quad k = \overline{1, M_i}, i = \overline{1, K}. \quad (26)$$

В последнем случае для экономии затрат производится сокращение выделенных приложениям $\{A_l\}$ ресурсов, а незадействованные ресурсы отключаются. Такие задачи решаются в [9—11].

Критерии (25) и (26) применяются только тогда, когда $\forall k = \overline{1, M_i}$ и $\forall i = \overline{1, K}$, а при сравнении значений q_{ki}^* и q_{ki} выполняется условие «только не больше» или «только не меньше». В противном случае ресурсы между приложениями из $\{A_l\}$ могут быть перераспределены так, чтобы приложениям, для которых $q_{ki}^* < q_{ki}$, были выделены ресурсы за счет приложений, для которых $q_{ki}^* > q_{ki}$, $\forall k = \overline{1, M_i}$, $i = \overline{1, K}$.

Если средствами нижнего уровня не удается обеспечить равенство $q_{ki}^* = q_{ki}$ при $q_{ki}^* < q_{ki}$, то запускается итеративная процедура, в которой задействуются вышерасположенные уровни (см. рис. 1). В этом случае на вышерасположенных уровнях приложению $A_l^* \in A$, для которого $q_{ki}^* < q_{ki}$, дополнительно выделяется квант Δr ресурсов. После чего проверяется выполнение условия (14). Если по-прежнему $q_{ki}^* < q_{ki}$, то выделяется еще один квант ресурсов Δr. Процедура повторяется до тех пор, пока не начнет выполняться условие (14). При отсутствии в ИТ-инфраструктуре запаса ресурсов при управлении уровнем услуг возможны две ситуации:

1) пересматриваются значения матицы Q;

2) выделяются кванты ресурса Δr за счет приложений из множества $\{A_l\}$ с учетом важности W_p услуг.

Зависимость значений показателей качества q_{ki}, $\forall k = \overline{1, M_i}$, $i = \overline{1, K}$ от объема ресурсов r без потери общности можно представить в следующем виде:

$$q = f_{qr}(r), \quad (27)$$

где q — показатель качества услуги. Для увеличения значения q соответствующему приложению из множества $\{A_l\}$ необходимо выделить дополнительные ресурсы. Тогда

$$q' = f_{qr}(r + \Delta r). \quad (28)$$

Если $\Delta r > 0$, то $q' \geq q$, что позволяет сделать предположение о монотонном характере функции f_{qr}.

Аналогичным образом можно предположить, что функция

$$q = f_{qa}(\hat{a}) \quad (29)$$

также будет монотонной.

Тогда, если функции (27) и (29) монотонные, то функция

$$q = f_q(r, \hat{a}) \quad (30)$$

также будет монотонной [15].

Пусть управление $u^+ \in U$, где U — множество управляющих воздействий, заключается в выделении дополнительных ресурсов приложению $A_l^* \in A$, для которого фактическое качество $q_ф$ хуже целевого $q_ц$, $q_ф < q_ц$, а $u^- \in U$ — управление, изымающее ресурсы у приложения $A_l^* \in A$, если $q_ф > q_ц$.

Учитывая монотонный характер зависимости q_{ki}, $\forall k = \overline{1, M_i}$, $i = \overline{1, K}$ от r, докажем необходимые утверждения, предварительно сделав следующие выкладки.

Использование ресурсов приложениями зададим матрицей $\hat{\mathrm{P}} = \left\| \hat{\rho}_{lj} \right\|$, $l = \overline{1, I}$, $j = \overline{1, m}$, причем:

$$\hat{\rho}_{lj} = \begin{cases} n_{lj}\Delta r_j, \text{если } l\text{-е приложение использует } j\text{-й ресурс;} \\ 0, \text{если } l\text{-е приложение не использует } j\text{-й ресурс,} \end{cases} \quad (31)$$

где n_{lj} — количество квантов j-го ресурса, выделенного l-му приложению; Δr_j — размер кванта j-го ресурса. При этом должны выполняться ограничения:

$$\Delta r_j \sum_{l=1}^{I} n_{lj} \leq r_j, \quad \forall j = \overline{1, m}. \quad (32)$$

Тогда можно определить следующее отображение:

$$\hat{Q} = U \times \mathrm{P} \times \bar{\hat{a}} \to Q, \quad (33)$$

где $\bar{\hat{a}} = \{\hat{a}\}$ — множество векторов $\hat{a} \in \bar{\hat{a}}$.
В свою очередь :

$$\hat{U} = Q^* \times \mathrm{P} \times \bar{\hat{a}} \to U. \quad (34)$$

Утверждение 1. Для заданных значений q_{ki}, $\forall k = \overline{1, M_i}, i = \overline{1, K}$ в случае, когда $q_{ki}^* < q_{ki}$, существует управление $u^+ \in U$, позволяющее обеспечить $q_{ki}^* = q_{ki}$ при $\min r$ на поддержание уровня услуг.

Доказательство следует из монотонности функций (27)—(30), конечности множеств Q^*, P и $\bar{\hat{a}}$ и сопоставимости целей процессов на рис. 1.

Управление $u^+ \in U$ находится итеративно.

Для доказательства следующего утверждения введем отображение:

$$\hat{F} = Q \times \bar{\hat{a}} \to R. \quad (35)$$

Утверждение 2. При выполнении условия (22) и $q_{ki}^* < q_{ki}$, если известны значения $\hat{a}$, то управление $u^+ \in U$, позволяющее восстановить равенство $q_{ki}^* = q_{ki}$, $\forall k = \overline{1, M_i}, i = \overline{1, K}$, может быть найдено без использования итерационных процедур.

Доказательство. При фиксированных значениях вектора $\hat{a}$, исходя из (35), существуют зависимости $q_{\text{ц}} \to r$, $q_1 \to r_1$ и $q_2 \to r_2$. Тогда, по аналогии с $q_1 - q_2 \to r_1 - r_2 = \Delta r_{12}$, при $q_{ki}^* < q_{ki}$ для обеспечения выполнения равенства $q_{ki}^* = q_{ki}$ необходимо l-му приложению, поддерживающему i-ю услугу с учетом (31), выделить дополнительно $\Delta\hat{\rho}_{lj}, i = \overline{1, K}$, $l = \overline{1, I}$, $j = \overline{1, m}$, причем величина $\Delta\hat{\rho}_{lj}$ может быть определена на основе $\Delta q_{ki} = q_{ki} - q_{ki}^* \to \Delta\hat{\rho}_{lj}$. В общем случае выделение дополнительных ресурсов с учетом ограничения (32) без итераций может быть осуществлено только при выполнении условия (22), что и доказывает утверждение.

Следствие. Если $q_{ki}^* < q_{ki}$ и i-я услуга поддерживается приложением $A_l^* \in A$, а за счет приложений из множества A, для которых $q_{\text{ф}} > q_{\text{ц}}$, можно высвободить объем ресурсов $\Delta\hat{\rho}_{в}$, причем $\Delta\hat{\rho}_{в} \geq \Delta\hat{\rho}_{lj}$, где $\Delta\hat{\rho}_{lj}$ — дополнительный объем ресурсов, который нужно выделить l-му приложению для того, чтобы обеспечить выполнение $q_{ki}^* = q_{ki}$, то управление $u^+ \in U$ в условиях дефицита ресурсов позволяет восстановить уровень i-й услуги, поддерживаемой l-м приложением за один проход.

Реализацию нижнего уровня, на котором непосредственно осуществляется оперативное управление уровнем услуг, целесообразно осуществлять на основе координатора [15].

Исследование предложенного декомпозиционно-компенсационного подхода к управлению уровнем услуг не только подтвердило его работоспособность, но и выявило способность повышения эффективности использования ресурсов. Предлагаемый подход сравнивался с методом полного резервирования ресурсов и методом наращивания нод по эффективности использования ресурсов $q_{Э}$, определяемой, как:

$$q_{Э} = \frac{r_{и}}{r_{в}}, \quad (36)$$

где $r_{и}$ — количество фактически используемых ресурсов, а $r_{в}$ — количество зарезервированных и выделенных ресурсов.

Наиболее распространенный на практике метод полного резервирования предполагает определение количества ресурсов $r_{\max,l}$ по выражению (19) для максимального количества $a_{\max,l}$ пользователей l-й услуги, оговоренному в SLA. Объем ресурсов $r_{\max,l}$ закрепляется за приложением A_l и в процессе работы не изменяется. Если фактическое количество клиентских запросов $a_l < a_{\max,l}$, то ресурсы используются неэффективно. При $a_l > a_{\max,l}$ качество услуг снижается, управление для повышения уровня услуг не осуществляется, и пользователи довольствуются фактическим качеством услуг.

Метод поддержания качества услуг путем наращивания нод при горизонтальном масштабировании отслеживает не уровень услуг, а процент утилизации выделенных ресурсов. При превышении степени задействованности отдельных ресурсов предварительно заданного порогового значения осуществляется увеличение объема выделенных приложению ресурсов на размер ноды. При этом производится увеличение объема всех видов ресурсов независимо от реальной потребности услуг в увеличении

объемов только некоторых из них. При этом незадействованные ресурсы не могут быть использованы другими приложениями.

Зависимость эффективности использования ресурсов $q_Э$ от соотношения $a_l / a_{\max,l}$ приведена на графиках рис. 4 для предлагаемого подхода (кривая 1), метода полного резервирования ресурсов (кривая 2) и метода наращивания нод (кривая 3) при разном количестве d_l ресурсов, используемых приложением A_l, независимо от количества клиентских запросов. Для проведения исследований величина кванта ресурсов устанавливалась на уровне 10% от размера ноды.

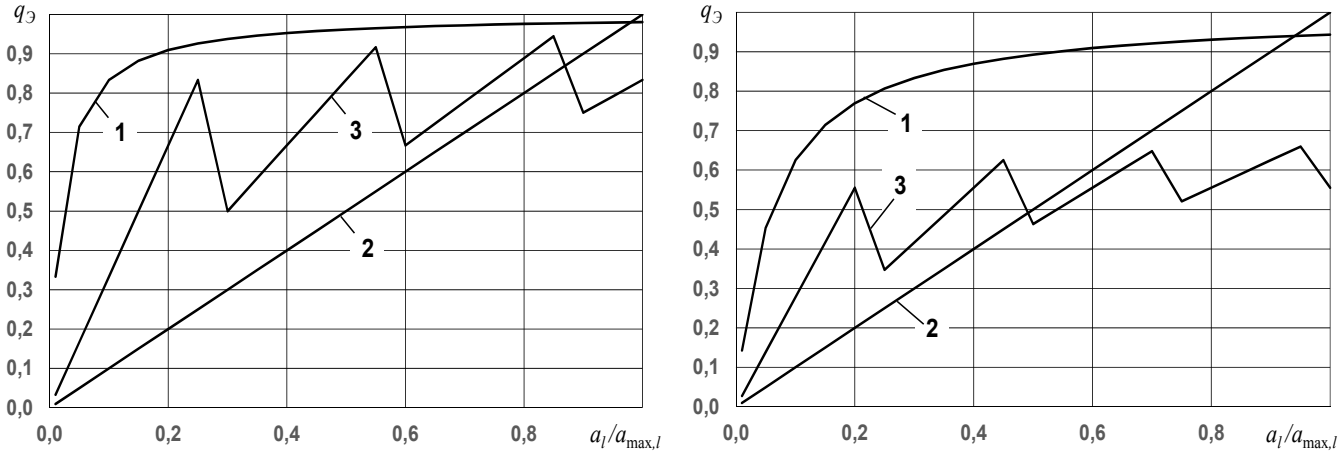


Рис. 4. Зависимости $q_Э$ от $a_l / a_{\max,l}$ для случаев: а) пропорциональных и б) непропорциональных потребностей в ресурсах

Из анализа графиков на рис. 4 видно, что предлагаемый подход гораздо эффективнее использует ресурсы корпоративной ИТ-инфраструктуры, причем эффективность использования ресурсов увеличивается по мере уменьшения значения соотношения $a_l / a_{\max,l}$.

Выводы

Эффективное управление уровнем услуг в корпоративных ИТ-инфраструктурах возможно при условии применения предложенного декомпозиционно-компенсационного подхода, предполагающего декомпозицию задач управления уровнем услуг и компенсацию негативного влияния различных факторов выделением дополнительных ресурсов критичным приложениям. Подход основан на интегрированном взаимодействии трех иерархических процессов — согласования уровня услуг, планирования ресурсов и управления уровнем услуг с учетом иерархии ИТ-инфраструктуры. Это дает возможность создать иерархию решений по управлению и поддержанию согласованного уровня услуг с учетом существующих ресурсных ограничений и полномочий уровней по выбору управления, задействуя механизмы и возможности верхних уровней иерархии для выбора управления при невозможности реализации управления на нижних уровнях.

Список литературы

1. Брукс П. Метрики для управления ИТ-услугами: пер. с англ. / П. Брукс. – М.: Альпина Бизнес Букс, 2008. – 283 с.
2. Ролик А.И. Система управления корпоративной информационно-телекоммуникационной инфраструктурой на основе агентского подхода / А.И. Ролик, А.В. Волошин, Д.А. Галушко, П.Ф. Можаровский, А.А. Покотило // Вісник НТУУ «КПІ»: Інформатика, управління та обчислювальна техніка. – К.: «ВЕК+», 2010. – № 52. – С. 39–52.
3. Теленик С.Ф. Управління ресурсами центрів оброблення даних / С.Ф. Теленик, О.І. Ролік, К.О. Крижова // Сучасні проблеми прикладної математики та інформатики: XVI Всеукр. наук. конф., 8–9 жовт. 2009: матеріали. – Львів: Видавничій центр ЛНУ, 2009. – С. 203–205.

4. Hubbert E. TechRadar™ For I&O Professionals: IT Service Management Processes, Q1 2012 / E. Hubbert, J.P. Garbani, G. O'Donnell, S.Mann, J. Rakowski. – Forrester Research, Inc. – 2012. – Feb. 7. – 44 p.
5. IT Service Management: An Introduction// J.V. Bon, G. Kemmerling, D. Pondman, Publisher: Van Haren Publishing. – 2002. – 217 p.
6. Information technology. Service management. Part 1: Specification: ISO/IEC 20000-1:2005. – ISO/IEC, 2005. – 16 p.
7. Information technology. Service management. Part 2: Code of practice: ISO/IEC 20000-1:2005. – ISO/IEC, 2005. – 34 p.
8. Ролик А.И. Концепция управления корпоративной ИТ-инфраструктурой / А.И. Ролик // Вісник НТУУ «КПІ»: Інформатика, управління та обчислювальна техніка. – К.: «ВЕК+», 2012. – № 56. – С. 31–55.
9. Теленик С.Ф. Моделі управління віртуальними машинами при серверній віртуалізації / С.Ф. Теленик, О.І. Ролік, М.М. Букасов, А.Ю. Лабунський // Вісник НТУУ «КПІ»: Інформатика, управління та обчислювальна техніка. – К.: «ВЕК+», 2009. – № 51. – С. 147–152.
10. Теленик С.Ф. Управляемый генетический алгоритм в задачах распределения виртуальных машин в ЦОД / С.Ф. Теленик, А.И. Ролик, П.С. Савченко, М.Е. Боданюк // Вісник ЧДТУ. – 2011. – № 2. – С. 104–113.
11. Теленик С.Ф. Адаптивный генетический алгоритм для решения класса задач распределения ресурсов ЦОД / С.Ф. Теленик, А.И. Ролик, П.С. Савченко // Вісник НТУУ «КПІ»: Інформатика, управління та обчислювальна техніка. – К.: «ВЕК+», 2011. – № 54. – С. 164–174.
12. Ролик А.И. Модель управления перераспределением ресурсов информационно-телекоммуникационной системы при изменении значимости бизнес-процессов// Автоматика. Автоматизация. Электротехнические комплексы и системы. – 2007. – №2 (20).– С. 73–82.
13. Клейнрок Л. Теория массового обслуживания / Л. Клейнрок. – М.: Машиностроение, 1979. – 432 с.
14. Острейковский В.А. Теория надежности / В.А. Острейковский. – М.: Высшая школа. – 2003. – 463 с.
15. Месарович М. Теория иерархических многоуровневых систем: пер. с англ. / М. Месарович, Д. Мако, И. Такахара; пер. с англ. под ред. И. Ф. Шахнова. – М.: «Мир», 1973. – 344 с.

УДК 519.85

ШОВГУН Н.В.

ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ РОБОТИ НЕЧІТКИХ НЕЙРОННИХ МЕРЕЖ В ЗАДАЧІ ОЦІНКИ КРЕДИТОСПРОМОЖНОСТІ ПОЗИЧАЛЬНИКА

В статті зроблено огляд методів оцінки кредитоспроможності позичальника на основі нечіткої логіки. Зроблені експериментальні дослідження методів навчання та адаптації нечітких нейронних мереж. Запропоновано модифікований алгоритм адаптації.

The problem of assessing the creditworthiness of the borrower is considered. The application of fuzzy neural networks for this problem solution was suggested. The experimental investigations of the adaptation and learning algorithms of fuzzy neural networks were carried out. The modification of adaptation and learning algorithms of fuzzy neural networks was suggested.

1. Вступ

У даній статті розглянуто практичне застосування нечітких нейронних мереж (ННМ) до задачі оцінки кредитоспроможності позичальника. Досліджувалися алгоритми навчання і адаптації вибраних ННМ, особливості їх застосування та ефективність запропонованих методів до поставленої задачі.

2. Нечіткі нейронні мережі та їх переваги

Для зниження кредитного ризику банку традиційно виконують оцінку кредитоспроможності позичальника. Інформація, на основі якої приймається рішення про кредитування, може бути неточною, неповною, а дані про позичальника можуть бути такими, що важко формалізуються. Для врахування цих факторів доцільно використовувати нечітку логіку, де процес прийняття рішень базується на зрозумілій базі правил. В статті [1], наприклад, представлена методика оцінки кредитоспроможності фізичних осіб з використанням методу парних порівнянь та нечітких систем з логічним висновком типу Мамдані. Але недоліком систем нечіткого логічного висновку є те, що вони не здатні автоматично навчатися, і параметри та вид функцій належності, які описують фактори кредитоспроможності задаються експертом та можуть бути неадекватними. Нечіткі нейронні мережі поєднують переваги систем нечіткого логічного висновку та нейронних мереж – здатність до адаптації та автоматичного навчання, а також можливість інтерпретації процесу отримання результату.

Для аналізу кредитоспроможності юридичних осіб застосовувались нечітка нейронна мережа з виводом Мамдані [2] та нечітка нейронна мережа з виводом Сугено для фізичних осіб [3].

Першим етапом навчання ННМ є початкове налаштування нечіткої бази правил: вибір не тільки параметрів функцій належності, а й структури мережі – кількості правил. Існує декілька алгоритмів адаптації ННМ:

- проста побудова повної бази правил;
- алгоритм скорочення бази правил;
- алгоритм поступового нарощування бази правил.

Після визначення структури ННМ відбувається навчання отриманої бази правил.

Одним з найпоширеніших алгоритмів навчання нейронних мереж є градієнтний метод. Для ННМ на кожній ітерації методу уточнюються параметри функцій належності.

Наступний найпоширеніший метод навчання – генетичний алгоритм, де параметри, що настроюються складають особину популяції, до якої застосовуються механізми схрещування, мутації, селекції. В даній статті для поставленої задачі використовувались ННМ з виводом Мамдані та з виводом Сугено, а також рекурентна ННМ з виводом Сугено [4] та каскадна неофаззі мережа [5], яка має високу швидкість навчання.

3. Алгоритми навчання та адаптації нечітких нейронних мереж з виводом Мамдані та з виводом Сугено

На основі анкети позичальника необхідно обґрунтовано прийняти рішення – чи варто видавати кредит. Кожну таку анкету можна представити як вектор $\{X_1, X_2, ..., X_i, ..., X_M\}$, де X_i – деяким чином формалізовані дані про позичальника та параметри кредиту. Такий вектор являє собою вхідні дані мережі. Виходом

мережі буде рішення, щодо кредитування позичальника.

Перед початком роботи необхідно провести етап адаптації мережі - визначити її структуру, тобто вибрати кількість правил в базі знань, а відповідно визначити кількість нейронів другого та третього шару нечіткої нейронної мережі. Кількості правил вибирається або експертом, або за допомогою алгоритму адаптації. Серед алгоритмів адаптації ННМ найбільш ефективним виглядає алгоритм поступового нарощування правил. Так, в статті [4] пропонується алгоритм адаптації на основі активуючої сили правила. В цьому алгоритмі нове правило додається, якщо виконується умова: $I = \arg\max_{1\le k\le r} F^k(x) \le F_{in}$, де $F^k(x(t)) = w_k$ - активуюча сила правила k для вхідного вектора t, а F_{in} - заздалегіть обраний поріг активуючої сили правила, який зменшується в процесі навчання. В згенерованому правилі параметри функцій належності встановлюються наступним чином: $c_j^{(k)} = x_j(t)$, $b_j^{(k)} = 1$, $\sigma_j^{(k)} = \beta \cdot \prod_{j=1}^{N} \frac{1}{1+(\frac{x_j - c_j^{(I)}}{\sigma_j^{(I)}})^{2b_j^{(I)}}}$. Тобто, ми додаємо правило, якщо жодне з вже існуючих правил не описує достатньо добре поточний вхідний вектор.

Алгоритм, запропонований в [7], базується на оцінці точності апроксимації. Він має вигляд:

1. Задаються початкові значення: ε - точність апроксимації, $m = m_0$ - число правил в базі знань;

2. Вибирається наступна точка $\vec{x}^i$, для якої значення $\left|d^i - y^i\right|$ максимальне. Якщо база знань пуста, то переходимо до кроку 4. В іншому випадку за допомогою алгоритму нечіткого логічного висновку Сугено та існуючої бази правил рахується значення d^i.

3. Далі перевіряється умова $\left|d^i - y^i\right| \le \varepsilon$, тобто оцінюється точність апроксимації поточної бази правил. Якщо вона не виконується, то переходимо на крок 4, інакше на крок 5.

4. До бази знань додається правило з таким значенням параметрів функції належності (ФН): $c_j^{(k)} = x_j^i$, $b_j^{(k)} = 1$, $\sigma_j^{(k)} = \beta \cdot \prod_{j=1}^{N} \frac{1}{1+(\frac{x_j - c_j^{(o)}}{\sigma_j^{(o)}})^{2b_j^{(o)}}}$. Кількість правил стає рівною $m = m + 1$.

5. Перевіряється умова зупинки алгоритму: чи переглянуті всі навчальні точки. Якщо ні, то вибирається наступна навчальна точка.

В цьому алгоритмі ми додаємо правило, якщо існуюча база знань дає завелику помилку для поточної точки.

Для підвищення ефективності пропонується в обох алгоритмах робити додаткову перевірку умови $f_o = \min(f_1, f_2, ..., f_m) > R$, де $f_k = \left\|\vec{c}^{(k)} - \vec{x}^i\right\|$, $k = 1, m$ - відстань між центрами ФН кожного правила та поточною точкою, $R = const$ - порогова відстань між правилами та поточною точкою. Таким чином, посилюється контроль за кількістю правил.

В ННМ результат отримується за допомогою апарату нечіткої логіки, але відповідні функції належності настроюються з використанням алгоритмів навчання ННМ, наприклад, алгоритму зворотного поширення помилки. Для функції помилок виконується крок градієнтним методом. Таким чином, мережа використовує апріорну інформацію, отримує нові знання та залишається логічно прозорою для користувача.

Так для ННМ з виводом Мамдані критерій оптимізації (функція помилок) має вигляд: $E(z) = \frac{1}{2}(z^* - z_0)^2 \to \min$, де z^* - бажаний вихід мережі; z_0 - фактичний вихід. Алгоритм навчання має вигляд:

$$\left.\begin{aligned} a_k(n+1) &= a_k(n) - \gamma_n^{'} \frac{\partial E(n)}{\partial a_k} \\ \sigma_k(n+1) &= \sigma_k(n) - \gamma_n^2 \frac{\partial E(n)}{\partial \sigma_k} \end{aligned}\right\},$$

$$\left.\begin{aligned} a_{ik}(n+1) &= a_{ik}(n) - \gamma_n^3 \frac{\partial E(n)}{\partial a_{ik}} \\ \sigma_{ik}(n+1) &= \sigma_{ik}(n) - \gamma_n^4 \frac{\partial E(n)}{\partial \sigma_{ik}} \end{aligned}\right\},$$

де - $\gamma_n^{'}$, γ_n^2, γ_n^3, γ_n^4 - розмір кроку.

В ННМ з виводом Сугено – ННМ TSK (Takagi, Sugeno, Kang'a) використовується така база правил:

Π_1 : *якщо* $x_1 \in A_1^{(1)}; x_2 \in A_2^{(1)}, ..., x_n \in A_n^{(1)}$, *то* $y_1 = p_{10} + \sum_{j=1}^{N} p_{1j} x_j$,

Π_M : *якщо* $x_1 \in A_1^{(M)}; x_2 \in A_2^{(M)}, ..., x_n \in A_n^{(M)}$, *то* $y_M = p_{M0} + \sum_{j=1}^{N} p_{Mj} x_j$,

де $A_i^{(k)}$ - значення лінгвістичної змінної x_i для правила Π_k з функцією належності

$$\mu_A^{(k)}\left(x_i\right) = \frac{1}{1 + \left(\frac{x_i - c_i^{(k)}}{\sigma_i^{(k)}}\right)^{2b_i^{(k)}}}.$$

В рекурентній ННМ TSK вихід кожного правила подається на вхід мережі, таким чином враховується минулий вплив правила. Вихід кожного правила має вигляд: $y_k(x) = p_{k0} + \sum_{j=1}^{N} p_{kj} x_j + p_{kN+1} h_k$, де h_k - вихід правила на минулому кроці.

Можна використовувати звичайний градієнтний алгоритм та настроювати параметри $c_j^{(k)}$, $\sigma_j^{(k)}$, $b_j^{(k)}$, p_{k0}, p_{kj}, та h_k. Критерій оптимізації має вигляд: $E = \frac{1}{L}\sum_{l=1}^{L}(y_l - d_l)^2$, де L – об'єм навчальної вибірки. Але найчастіше використовують гібридний алгоритм навчання, в якому параметри p_{k0}, p_{kj} знаходяться в результаті рішення системи з L лінійних рівнянь при фіксованих інших параметрах. Детальний опис методів можна знайти в [6].

Також для знаходження параметрів функції належності використовують генетичний алгоритм. В генетичному алгоритмі навчання всі змінні, що підлягають настроюванню включають до особини популяції:

$$W = [c_1^{(1)}, ..., c_N^{(1)}, b_1^{(1)}, ..., b_N^{(1)}, \sigma_1^{(1)}, ..., \sigma_N^{(1)}, p_{10}, p_{11}, ..., p_{1N}, c_1^{(M)}, ..., c_N^{(M)}, b_1^{(M)}, ..., b_N^{(M)}, \sigma_1^{(M)}, ..., \sigma_N^{(M)}, p_{M0}, p_{M1}, ..., p_{MN}].$$

Спершу задається початкова популяція з L особин. Один крок алгоритму містить такі етапи:

1. Знаходиться індекс придатності $FI(W_i) = C - \frac{1}{L}\sum_{l=1}^{L}(y_l - d_l)^2$, де C – константа.

2. Вибираються особини для схрещування на основі ймовірнісного механізму. Ймовірність вибору і-го батька $P_i = \frac{FI(Wi)}{\sum_{l=1}^{L} FI(W_i)}$.

3. Далі застосовується алгоритм схрещування.

4. Отримана популяція підлягає мутації.

5. Формується нова популяція.

Оскільки в генетичному алгоритмі для різних запусків популяція може сходитися до різних рішень, доцільно використовувати острівну модель алгоритму. В цій моделі основна популяція ділиться на декілька популяцій, де для кожної виконується генетичний алгоритм, це дає можливість поєднати «досягнення» кожної підпопуляції.

Кожний з розглянутих алгоритмів має свої недоліки. Так, градієнтний алгоритм сильно залежить від початкових умов, а генетичний часто сходиться до локальних оптимумів. Автором пропонується використовувати гібридний алгоритм, в якому початкове наближення (початкові значення параметрів функцій належності) близьке до оптимального знаходиться за допомогою генетичного алгоритму, а вже далі воно розглядається як початкова точка для градієнтного алгоритму.

Для порівняння з вищеозначеними ННМ використовувалася каскадна неофаззі мережа. В цій мережі в якості вузлів використовується неофаззі нейрон, який реалізує $\hat{y} = \sum_{i=1}^{n} f_i(x_i) = \sum_{i=1}^{n}\sum_{j=1}^{h} w_{ji}\mu_{ji}(x_i)$, де μ_{ji} - трикутна функція належності вхідної змінної, w_{ji} - синаптична вага. Детальний опис алгоритму навчання можна знайти в [5].

4. Результати досліджень

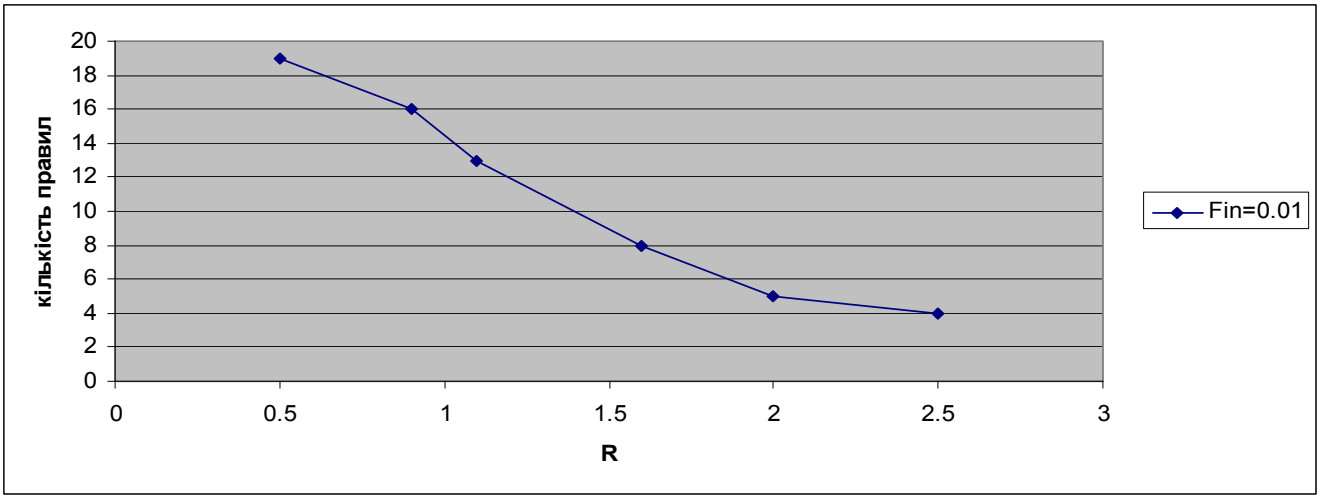
Для аналізу кредитоспроможності за допомогою запропонованих методів використовувалась вибірка даних одного з українських банків, яка складається з 998 заявок за 2010 рік. Для оцінки запропонованих методів використовувались такі показники: середньо квадратична похибка (СКП), відсоток вірних класифікації та час роботи алгоритму. Результати досліджень, щодо розглянутих алгоритмів адаптації та їх модифіковані варіанти представлені в таблиці 2. Для дослідження алгоритмів адаптації використовувалась вибірка з 300 заявок.

Табл. 1. Результати оцінки кредитоспроможності позичальника для різних алгоритмів адаптації

Метод	Кількість правил	СКП	Відсоток вірних класифікацій, %	Час роботи алгоритму, с
ННС TSK де кількість правил встановлюється експертом	8	0.1139	89	4021
ННС TSK з алгоритмом, заснованим на активуючій силі правила	6	0.1097	89.4	2281
ННС TSK з модифікованим алгоритмом, заснованим на активуючій силі правила	5	0.117	88.6	1649
ННС TSK з алгоритмом, заснованим на оцінці точності апроксимації	7	0.1103	88	3539
ННС TSK з модифікованим алгоритмом, заснованим на оцінці точності апроксимації	4	0.1139	87	1053

Як бачимо, кількість правил при модифікованих алгоритмів адаптації менше ніж при звичайних, при цьому відсоток вірних класифікацій суттєво не змінюється, а час роботи алгоритму зменшується. Експертом може бути задано надлишкове число правил в базі знань, тоді в процесі навчання, деякі правила будуть повторюватися. Але структура мережі буде відповідати запропонованим числу правил - буде надмірно складною і час навчання буде більшим. В модифікованих алгоритмах менша кількість правил генерується завдяки додатковій перевірці відстані між центрами ФН кожного правила та поточною точкою. Тобто нове правило додається якщо поточна точка погано описується існуючою базою правил та відстань між центрами ФН кожного правила та поточною точкою достатньо велика (задля того щоб не було однакових правил). Тож ефективніше застосовувати модифікований алгоритм.

Варто зауважити, що кількість згенерованих правил залежить від параметрів F_{in}, R для модифікованого алгоритму адаптації, заснованому на активуючій силі правил, та від ε, R для модифікованого алгоритму заснованому на оцінці точності апроксимації. Результати досліджень для алгоритму заснованому на активуючій силі правила представлені на графіках 1-2.



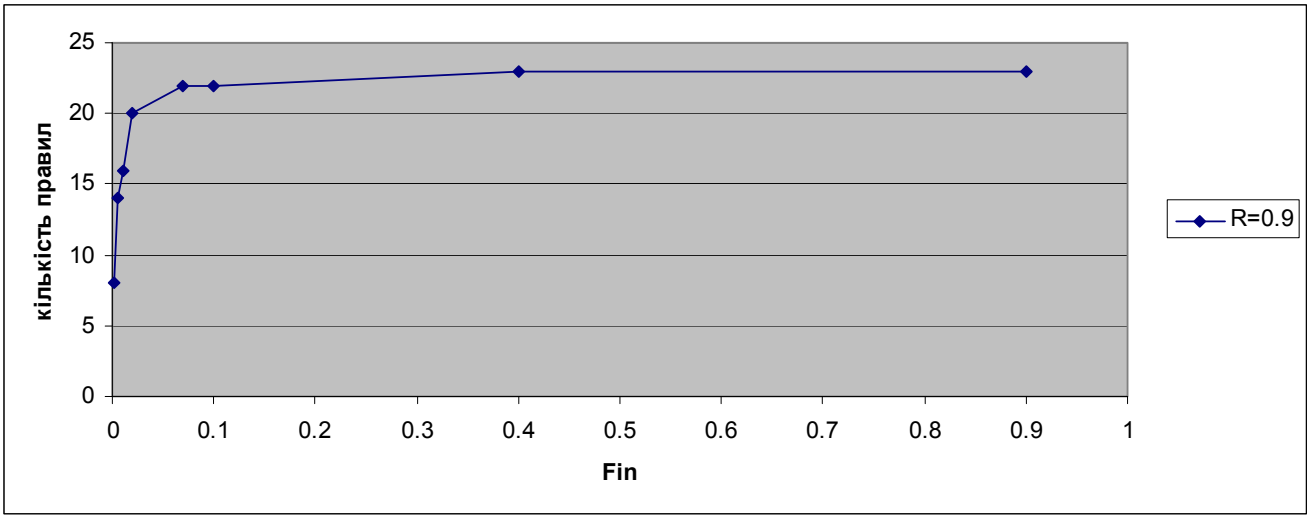
Графік 1. Залежність згенерованої кількості правил від параметру R

Як бачимо на графіку 1, при фіксованому значенні параметру F_{in}, більша кількість правил генерується для меншого значення R.

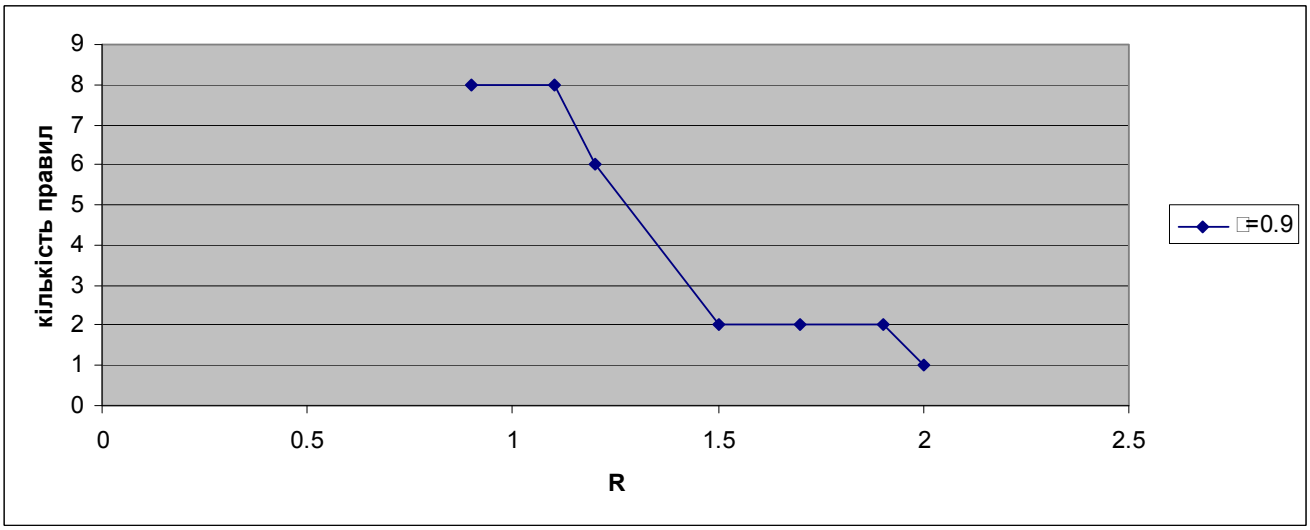
При фіксованому значенні R, кількість правил збільшується при збільшенні F_{in} - чим вище активуюча сила правила, тим краще воно описує вхідний вектор. Але збільшення правил відбувається лише до деякого граничного значення F_{in}, після якого вже не має сенсу додавати правила, так як побудована база вже описує всі можливі вхідні вектори з навчальної вибірки. Результати досліджень для алгоритму, заснованому на оцінці точності апроксимації, представлені на графіках 3-4.

На графіку 3 бачимо, що при фіксованому значенні ε - кількість правил збільшується

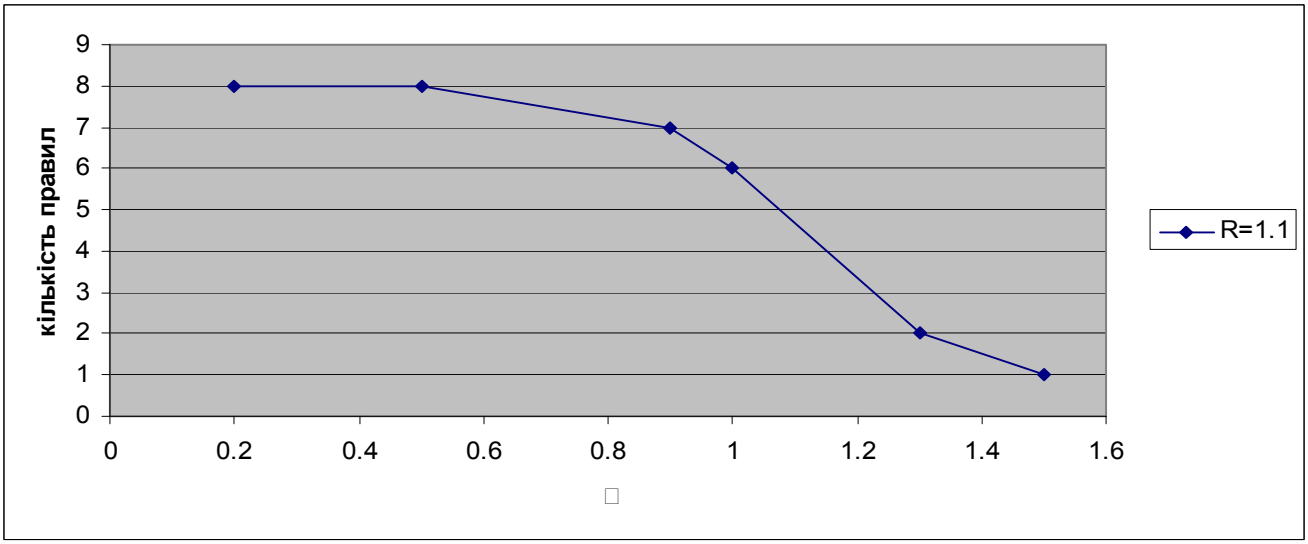
при зменшенні параметра R. В обох модифікованих алгоритмах адаптації параметр R відповідає за різноманітність правил – чим більше значення R, тим більша відстань між згенерованими правилами.



Графік 2. Залежність згенерованої кількості правил від параметру Fin



Графік 3. Залежність згенерованої кількості правил від параметру R



Графік 4. Залежність згенерованої кількості правил від параметру ε

З графіку 4 видно, що при фіксованому значенні R - кількість правил збільшується при зменшенні параметру ε. Тобто, чим більшу точність ми вимагаємо від бази правил, тим більше правил генерується.

Результати експериментів для різних нечітких нейронних мереж представлені в таблиці 1. Число правил вибрано 9.

Табл. 2. Результати оцінки кредитоспроможності позичальника для різних ННМ

Метод	СКП	Відсоток вірних класифікацій, %	Час роботи алгоритму, с
ННМ TSK з генетичним алгоритмом навчання	0.129	86.3	1161
ННМ TSK з градієнтним алгоритмом навчання	0.116	87.0	2024
ННМ TSK з гібридним алгоритмом навчання	0.115	87.7	3129
РННМ TSK з гібридним алгоритмом навчання	0.121	86.0	4214
Неофаззі каскадна мережа	0.102	85.3	187
ННМ з виводом Мамдані з генетичним алгоритмом навчання	0.187	85.0	1093
ННМ з виводом Мамдани с градієнтним алгоритмом навчання	0.112	85.3	2386

Як бачимо, найбільший відсоток вірних класифікації у ННМ TSK з гібридним алгоритмом навчання. Час роботи алгоритму найменший у каскадної неофаззі мережі, але вибір параметрів трикутної ФН, що використовується в алгоритмі, лежить на експертах.

5. Висновки

Однією з переваг ННМ являється їх здатність до навчання, тобто відпадає необхідність залучення експертів до налаштування бази правил для поточної ситуації.

Проведено аналіз ефективності застосування нечітких нейронних мереж до задачі оцінки кредитоспроможності позичальника та проведено аналіз ефективності методів їх навчання. Отримані результати показали ефективність використання при навчанні ННМ гібридного алгоритму (комбінації градієнтного та генетичного методів). Кращі результати за відсотком вірних класифікацій показала ННМ TSK з гібридним алгоритмом навчання. Запропоновано та проаналізовано новий алгоритм адаптації нечіткої нейронної мережі, в якому база правил настроюється автоматично. Як наслідок, відбувається побудова оптимальної структури мережі, для навчання якої необхідно менше часу. Реалізація запропонованих алгоритмів дає можливість побудови бази правил без допомоги експертів.

Перелік посилань

1. Кузнецов Л.А. Оценка кредитной истории физических лиц на основе нечетких моделей / Л.А. Кузнецов, А.В. Перевозчиков // Управление большими системами. - 2007. - № 21 – с. 84-106
2. Зайченко Ю.П. Оценка кредитних банковских рисков с использованием нечеткой логики // Intelligent Information and Engineering Systems. – 2008. - №13 – с. 190-20
3. Шовгун Н.В. Аналіз кредитоспроможності позичальника за допомогою методів з нечіткою логікою // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка: Зб. наук. пр. – 2012. – № 55. – с. 169-174
4. CF Juang and CT Lin A recurrent self-organizing neural fuzzy inference network // IEEE transaction of neural networks, Volume 10, Number 4, Pages 828–845, July 1999
5. Бодянский Е.В. Каскадная эволюционная нейронная сеть с неофаззи нейронами в качестве узлов// Восточно-Европейский журнал передовых технологий – 2011. – Вып. 4/3 (52). – с. 55–58.
6. Зайченко Ю.П. Нечеткие модели и методы в интеллектуальных системах. – К.: Издательский дом «Слово», 2008. – 334 с.
7. Круглов В.В., Борисов В. В. Искусственные нейронные сети. Теория и практика. -2-е изд., стереотип.- М.: Горячая линия-Телеком, 2002. -382

УДК 004.91

ПОЛТОРАК В.П.,
ГОЛКОВ В.Б.

ПОРІВНЯЛЬНИЙ АНАЛІЗ СИСТЕМ ЕЛЕКТРОННОГО ЦИФРОВОГО ПІДПИСУ ТА ОСОБЛИВОСТІ ЇХ РЕАЛІЗАЦІЇ НА ЕЛІПТИЧНИХ КРИВИХ

Робота присвячена огляду та аналізу алгоритмів формування електронного цифрового підпису. Порівнюються відомі алгоритми створення електронного цифрового підпису, розглядаються питання реалізації алгоритму електронного цифрового підпису на еліптичних кривих та пропонується схема його програмної реалізації.

The paper is devoted to the review and analysis of algorithms for creation electronic digital signature. Known electronic digital signature algorithms are compared, examined the questions of implementation of the elliptic curve digital signature algorithm, proposed scheme of its software implementation.

1. Вступ

Розвиток глобальних комунікацій в діловому і повсякденному житті привів до появи нової області взаємовідносин, предметом яких є електронний обмін даними. У такому обміні даними можуть брати участь органи державної влади, комерційні і некомерційні організації, а також громадяни в своїх офіційних і особистих стосунках.

Проблема збереження електронних документів від копіювання, модифікації і підробки вимагає для свого вирішення специфічних засобів і методів захисту. Одним з поширених в світі засобів такого захисту є Електронний цифровий підпис (ЕЦП), який забезпечує автентичність повідомлення та неспростовність застосування особистого ключа (автентифікація власника цифрового підпису)[1]. За допомогою спеціального програмного забезпечення ЕЦП підтверджує достовірність інформації документу, його реквізитів і факту підписання конкретною особою.

У даній роботі розглядаються методи створення ЕЦП та питання їх реалізації. ЕЦП являє собою реквізит електронного документу, що дозволяє встановити відсутність спотворення інформації в електронному документі з моменту формування ЕЦП і перевірити належність підпису власнику сертифіката ключа ЕЦП.

Використання ЕЦП дозволяє:

- замінити традиційні печатку та підпис при безпаперовому документообігу;
- удосконалити і здешевити процедуру підготовки, доставки, обліку і зберігання документів, гарантувати достовірність документації;
- значно скоротити час руху документів, прискорити і полегшити процес візування одного документу декількома особами;
- побудувати корпоративну систему обміну електронними документами;
- забезпечити цілісність – гарантію того, що інформація зараз існує в її початковому вигляді, тобто при її зберіганні або передачі не було проведено несанкціонованих змін.

Цифровий підпис провадить наступні функції [2]:

- автентифікація: для отримувача повідомлення повинна бути можливість переконатися в його достовірності;
- неможливість відмови від відправлення: відправник не повинен мати можливості заперечити відправлення та підпис повідомлення протягом деякого часу після відправлення;
- цілісність: отримувач повідомлення повинен мати можливість перевірити той факт, що повідомлення не було змінене при передачі.

Цифровий підпис повинен гарантувати наступні ознаки:

- немає можливості підробки підпису;
- підпис може бути перевіреним, засвідченим;
- після того, як повідомлення підписане, відправник не зможе відмовитися від факту його передачі.

2. Актуальність проблеми

На сьогоднішній день ЕЦП широко використовується юридичними та фізичними особами, активно впроваджується в державних установах і органах державної влади. У системах документообігу використовуються різні алгоритми

ЕЦП, і незважаючи на те, що загалом всі алгоритми виконують свої функції, між ними існує значна різниця, у кожного алгоритму є свої переваги та недоліки, які потребують дослідження. Таким чином, актуальність проблеми обумовлюється потребою вибору алгоритму ЕЦП, наявність власної його реалізації дозволить розширити можливості дослідження та аналізу якостей алгоритму, а також зіштовхне нас з проблемами практичного характеру.

3. Огляд існуючих рішень

Симетрична криптографія має, здавалося б, дуже просту і зрозумілу схему ключів – відправник і одержувач знають один загальний секрет, і третім особам він недоступний. Але ця на перший погляд зручна схема приховує в собі підводні камені при значному збільшенні числа абонентів, які бажають шифрувати взаємну переписку. Адже якщо користувач хоче конфіденційно вести справи з N партнерами, йому необхідно мати N секретних ключів, а загальна кількість ключів в системі зростає від кількості абонентів за формулою N*(N - 1)/2.

Крім того, з'являються питання довіреної доставки ключів абонентам і ситуації, коли одержувач, дешифрувавши повідомлення, може внести до нього зміни, а потім посилатися на документ, нібито вже отриманий у такому вигляді.

Асиметрична криптографія вирішує це питання кардинально інакше – по-перше кількість ключів скорочується до 2*N. Кожен абонент бере участь у спілкуванні в єдиному інформаційному полі, володіє двома ключами – відкритим (відомим всім іншим абонентам) і закритим (відомим тільки йому і зберігається ним у секреті). Відкритий ключ використовується для відправки повідомлень: з його допомогою на основі спеціального алгоритму будь-який бажаючий може зробити асиметричне шифрування – необоротне, без знання закритого ключа, перетворення документу. А ось дешифрувати повідомлення зможе лише власник закритого ключа, тобто законний одержувач.

Схеми ключів симетричною і асиметричною криптографії для порівняння наведені на рис. 1 та 2.

Ця робота зосереджена на дослідженні асиметричних схем цифрового підпису з доповненням (appendix) [2]. «Доповнення» означає, що використовується криптографічна хеш-функція для створення дайджесту повідомлення, і перетворення підпису повідомлення відбувається над дайджестом повідомлення, а не над власне повідомленням.

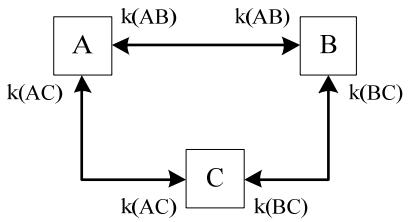


Рис. 1. Ключі в симетричній криптографії

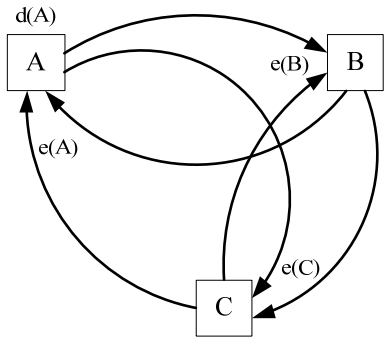


Рис. 2. Ключі в асиметричній криптографії

Сучасні схеми цифрового підпису можуть бути класифіковані таким чином відповідно до складної математичної проблеми, що лежить в їх основі, та забезпечує їх безпеку [3]:

1) Схеми факторизації цілих чисел (ФЦ), безпека яких залежить від складності розв’язку проблеми розкладу на множники (факторизації) цілих чисел. Прикладами цих схем є схеми підпису RSA та Рабіна.

2) Схеми дискретного логарифму (ДЛ), безпека яких залежить від складності розв’язку проблеми знаходження (звичайного) дискретного логарифму у скінченному полі. Прикладами є схеми підпису Ель-Гамаля, Шнорра, DSA.

3) Схеми на еліптичних кривих (ЕК), безпека яких залежить від складності розв’язку проблеми знаходження дискретного логарифму на ЕК. Прикладами є схеми підпису EC-KDSA, ECSS, ECDSA.

Схема цифрового підпису RSA. Для формування електронного підпису відправник виконує над контрольною сумою документу h ті ж самі дії, що і при шифруванні, але використовує не відкритий ключ одержувача, а свій власний закритий ключ, тобто $sign_i = (h_i^d \bmod n)$. Відкритий і закритий ключі просто міняються місцями. На приймальній стороні

одержувач зводить підпис у степінь відкритого ключа e відправника і отримує ($sign_i^e$ mod n) = (h_i^{de} mod n) = h_i (згідно з тими ж формулами, що і в асиметричному шифруванні RSA).

Якщо після зведення в степінь значення співпадає з обчисленою незалежно на приймальній стороні контрольною сумою документу, то перевірка вважається виконаною, а документ - справжнім. Ніхто, крім відправника, не знаючи d, не може обчислити такий підпис $sign_i$, щоб зведення її до рівня відкритого ключа e дало необхідну контрольну суму - це те ж саме важковирішуване завдання, що і в асиметричному шифруванні RSA. Отже, забезпечити документ таким підписом $sign_i$ міг тільки справжній власник закритого ключа. Схема ЕЦП наведена на рис. 3.

Схема цифрового підпису DSA. Згідно з FIPS Pub 186–2 (Federal Information Processing Standards (FIPS), 2000), DSA використовує наступні параметри:

1) p – простий модуль, де $2^{L-1} < p < 2^L$ для L=1024;

2) q – простий дільник p - 1, де $2^{159} < q < 2^{160}$;

3) $g = h^{(p-1)/q}$ mod p, де h будь-яке ціле число в межах $1 < h < p-1$ таке що $h^{(p-1)/q}$ mod $p > 1$ (g має порядок q mod p);

4) x = а випадково або псевдо випадково згенероване ціле число в межах $0 < x < q$;

5) $y = g^x$ mod p;

6) k = а випадково або псевдо випадково згенероване ціле число в межах $0 < k < q$;

Цілі числа p, q, та g можуть бути відкриті та загальновідомі групі користувачів. Закритий та відкритий ключі користувача - це x і y, відповідно. Параметри x та k використовуються лише для генерації підпису, та повинні триматися в таємниці. Параметр k повинен змінюватися для кожного підпису.

Підписом для повідомлення m буде пара чисел r та s обчислених згідно з рівняннями: $r=(g^k$ mod $p)$ mod q, та $s=(k^{-1}$ (SHA−1(m)+x r)) mod q.

Вище згадане k^{-1} є мультиплікативною інверсією k, mod q, тобто, $(k^{-1} * k)$ mod q=1 та $0 < {}^{k-1} < q$. Значенням SHA−1(m) є 160-бітовий рядок, що є виходом алгоритму Secure Hash Algorithm, визначеного у стандарті FIPS 180-1, та повинен бути перетворений у ціле число. Підпис разом з повідомленням передається отримувачу.

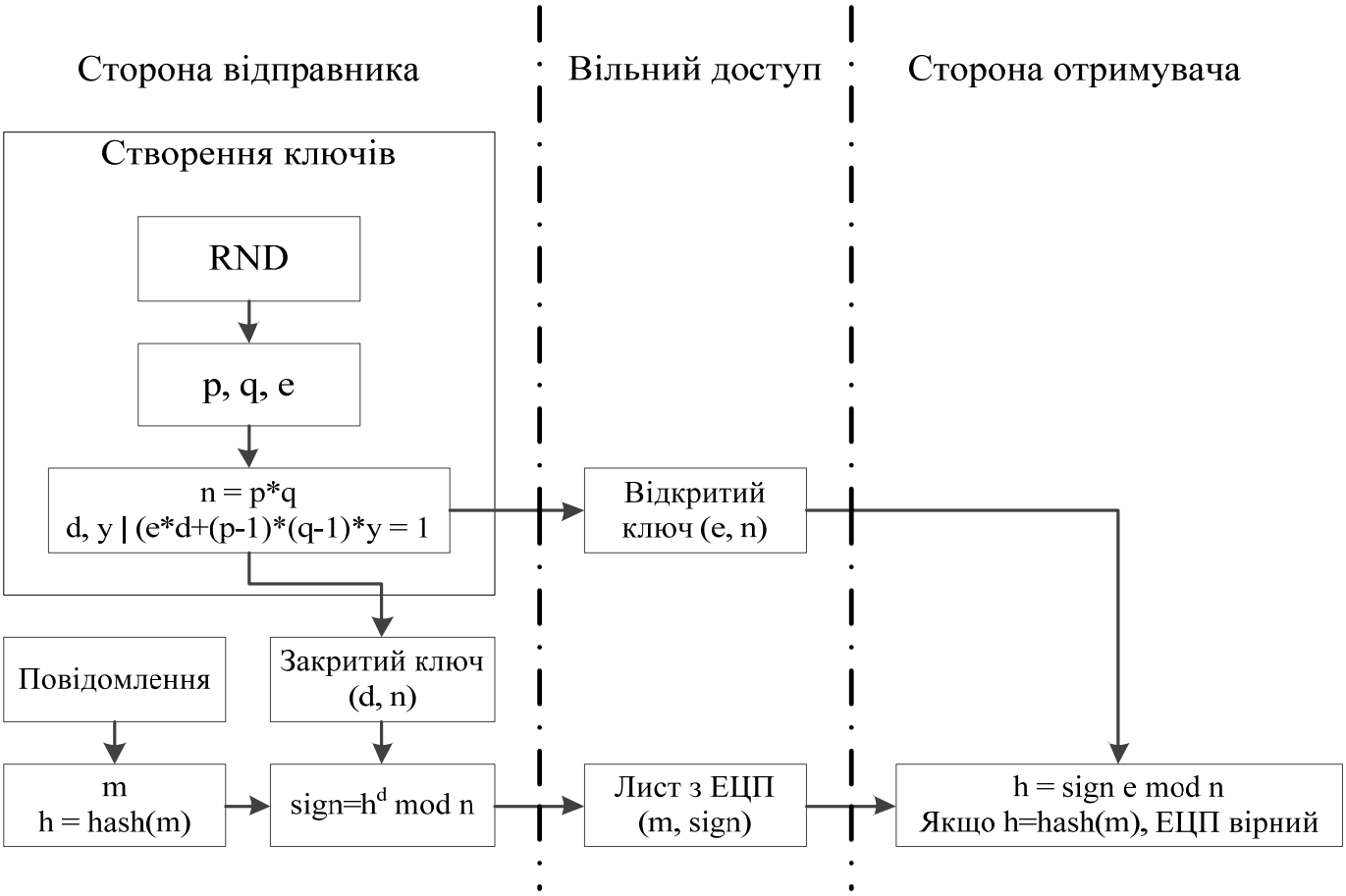


Рис. 3. Схема ЕЦП RSA

До процедури верифікації підпису необхідно отримати доступ до *p, q* та *g*. Як тільки відправник підписав повідомлення, він відправляє повідомлення, *m,* разом з цифровим підписом *r* та *s*.

Нехай *m′, r′,* та *s ’* – це отримані версії *m, r,* та *s,* відповідно, та нехай *y* буде відкритим ключем відправника. Щоб перевірити підпис відправника, отримувач спершу перевіряє чи лежать в заданих межах $0 < r' < q$ та $0 < s' < q$.

Якщо ця умова порушена, то підпис відкидається. Якщо ці дві умови виконуються, тоді перевірник обчислює:

$$w = (s')^{-1} \bmod q$$

$$u_1 = ((SHA-1(m')w) \bmod q$$

$$u_2 = ((r')w) \bmod q$$

$$v = ((g^{u_1} y^{u_2}) \bmod p) \bmod q$$

Якщо *v=r′,* тоді підпис є вірним та перевірник може бути певним, що це повідомлення дійсно було відправлене особою, що має закритий ключ, *x,* що відповідає відкритому ключу *y*.

Якщо *v* не дорівнює *r′,* тоді повідомлення могло бути змінене, або невірно підписане Алісою, або повідомлення могло бути підписане зловмисником, та, таким чином, повідомлення повинне бути визнане недійсним [2].

Схема цифрового підпису ECDSA. ECDSA є одним з варіантів схеми підпису Ель-Гамаля. ECDSA з’явився як модифікація алгоритму DSA, з метою використання еліптичних кривих в якості основи обчислень. Згодом алгоритм знайшов визнання і на цей час використовується у декількох стандартах [3].

Розглянемо базову схему алгоритму:

1) Загальновідомі параметри: в алгоритмі можливе використання скінченних полів двох видів: поля характеристики два або великого простого поля. Надалі розглянемо в якості базового велике просте поле. Відкриті параметри складаються з поля порядку *p*, еліптичної кривої визначеної двома елементами поля *a* та *b*, простого числа *n* більшого ніж 2^{160} , та елемента *G* порядку *n*, що належить еліптичній кривій. Рівняння еліптичної кривої має вигляд $y^2 = x^3 + ax + b$. Відкриті параметри є об’єктом для багатьох критеріїв безпеки.

2) Підготовка: обрати випадкове ціле число *d* в межах [1, *n* - 1], обчислити $Q = dG$. На виході отримаємо $(K_p, K_s) = (Q, d)$.

3) Генерація підпису:обрати випадкове ціле число *k* в межах [1, *n* - 1] та обчислити:

$$(x_1, y_1) = kG;$$

$$r = \bar{x}_1 \bmod n;$$

$$s = \frac{H(M) + ar}{k} \bmod n.$$

Тут $\bar{x}_1$ є певним чином перетворений в ціле число елемент скінченного поля x_1 . Якщо $r = 0$ або $s = 0$, знову переходимо на крок вибору *k*. Виходом процедури генерації підпису буде пара $\sigma = (r, s)$

4) Верифікація підпису: перевірити, що $Q \neq O$, $Q \in C$, та $nQ = O$. Перевірити, що *r* та *s* лежать в межах [1, *n* - 1] і $r = x_1 \bmod n$ для $(x_1, y_1) = u_1G + u_2Q$, $u_1 = \frac{H(M)}{s} \bmod n$, та $u_2 = \frac{r}{s} \bmod n$.

Складність алгоритму ECDSA полягає в необхідності роботи з об’єктами різних типів: точки еліптичної кривої, елементи поля, цілі числа, та рядки бітів. В стандартах, що використовують даний алгоритм, наведений докладний опис про способи представлення цих об’єктів та способи маніпуляції ними.

4. Порівняльний аналіз криптостійкості алгоритмів підпису

З трьох проблем, що використовуються в асиметричних схемах підпису, проблема факторизації цілих чисел та дискретного логарифму можуть бути вирішені алгоритмами, які виконуються за субекспоненційний час. Це означає, що проблема визнана досить важкою, але не настільки важкою як ті проблеми, що вирішуються алгоритмами лише за повністю експоненційний час. З іншого боку, найкращий алгоритм загального спрямування для проблеми дискретного логарифму в групі точок еліптичної кривої (ПДЛЕК) виконується за повністю експоненційний час. Це означає, що ПДЛЕК наразі визнана більш складною ніж проблема факторизації цілих чисел або проблема дискретного логарифму (ПДЛ).

Маючи на увазі цю властивість криптосистем на еліптичних кривих, висока безпека при відносно невеликому розмірі ключа, дослідимо властивості систем на еліптичних кривих порівняно з іншими криптосистемами ЕЦП.

Проблема факторизації цілих чисел (ПФЧ) полягає в наступному: маючи складене число *n*,

що є добутком двох великих простих цілих чисел p та q, знайти p та q. В той час як знаходження двох великих простих чисел є досить простим завданням, проблема розкладання добутку двох таких чисел на множники визнана обчислювально складною, якщо прості числа обрані належним чином. Схема ЕЦП RSA базується на цій проблемі.

Існує два основних типи алгоритмів факторизації [4]: загального спрямування та спеціалізованого спрямування. Алгоритми факторизації спеціалізованого спрямування намагаються використати спеціальні ознаки числа n, яке розкладається. На відміну від них, час виконання алгоритмів факторизації загального спрямування залежить лише від розміру числа n.

Одним з найбільш потужних алгоритмів факторизації спеціалізованого спрямування є метод факторизації з використанням еліптичних кривих (ECM), що був винайдений в 1985 році Хендріком Ленстрою. Час виконання цього алгоритму залежить від розміру простих множників числа n, тому в першу чергу алгоритм намагається знайти малі множники. Найбільшим простим множником знайденим на цей час за допомогою ECM є множник з 54 цифр (180 біт) числа з 127 цифр (422 біта) [4].

До розробки криптосистеми RSA, кращим алгоритмом факторизації загального спрямування був алгоритм неперервних дробів, який міг розкладати на множники числа до 40 десяткових цифр (133 біта). Цей алгоритм заснований на ідеї використання базового множника простого числа та генерації відповідної множини лінійних рівнянь, вирішення яких обов'язково приведе до розкладання на множники. Ця ідея лежить в основі і кращих алгоритмів, що використовуються на даний момент: квадратичне решето (QS) та загальне решето числового поля (NFS). Обидва алгоритми легко можуть виконувати факторизацію паралельно у розподілених мережах робочих станцій.

Квадратичне решето було використане, щоб розкласти на множники число з 129 десяткових цифр (429 біт), що було числом-викликом RSA поставленим в 1977 році.

Загальне решето числового поля, розроблене в 1989 році, найкраще працює на числах спеціальної форми (512+1). Згодом воно було розширене до алгоритму факторизації загального спрямування. Нещодавні експерименти довели, що NFS є дійсно потужним алгоритмом для розкладання на множники цілих чисел, що мають щонайменше 130 цифр (432 біта) [4].

Проблема дискретного логарифму в простому полі. Якщо p є простим числом, тоді Z_p визначає множину чисел $\{0, 1, 2, \ldots, p\text{-}1\}$, де додавання та множення виконуються по модулю p.

Проблема дискретного логарифмування в скінченному полі (ПДЛ) полягає в наступному: маючи просте число p, генератор α Z_p, та ненульовий елемент $\beta \in Z_p$, знайти унікальне ціле число $l, 0 \le l \le p-2$, таке що $\beta \equiv \alpha^l (\mathrm{mod}\, p)$. Ціле число l називається дискретним логарифмом β за базою α [4].

Базуючись на складності цієї проблеми, Діффі та Хеллман запропонували відому схему обміну ключами в 1976 році. З того часу багато інших криптографічних протоколів, чия стійкість залежить від ПДЛ, були запропоновані, включаючи DSA, схему Ель-Гамаля та схему Шнорра. Через інтерес до цих застосувань ПДЛ інтенсивно вивчалась математиками останні 20 років.

Як і з проблемою факторизації цілих чисел, існують два типи алгоритмів для вирішення проблеми дискретного логарифму. Алгоритми факторизації спеціалізованого спрямування намагаються використати спеціальні ознаки числа p, яке розкладається. На відміну від них, час виконання алгоритмів факторизації загального спрямування залежить лише від розміру числа p.

Найшвидші відомі алгоритми загального спрямування для вирішення ПДЛ засновані на методі, що називається індексне або диференційне обчислення. В цьому методі, створюється база даних з малих простих чисел та відповідних їм логарифмів, згідно з якою можуть бути отримані логарифми випадкових елементів поля. Це нагадує методи з базовим множником для проблеми факторизації. Тому якщо відбудеться прогрес в алгоритмі вирішення однієї з проблем, скоро після цього схожий покращений алгоритм буде знайдено і для іншої проблеми.

Найкращим відомим алгоритмом для вирішення ПДЛ на даний момент є загальне решето числового поля. Він має такий ж асимптотичний час виконання, що й відповідний алгоритм для факторизації цілих чисел. З цього можна

зробити висновок, що проблема знаходження логарифмів у випадку простого числа p розміром k біт має приблизно таку ж складність, що й розкладання складеного числа n розміром k біт. В 1998 році було оголошено розв'язок Виклику Діффі-Хеллмана, який включав вирішення ПДЛ по модулю простого числа з 129 цифр. Використане просте число мало спеціальну форму, тому був використаний алгоритм загального решета числового поля, який був дуже ефективним для цього виклику. Ці результати є доказами довгострокової безпеки, в криптосистемах, що використовують ПДЛ, слід використовувати модуль p розміром 1024 біта або більше [4].

Проблема дискретного логарифму у групі точок еліптичної кривої, визначеної над простим полем. Якщо p є простим ступенем, то F_p визначає скінченне поле, що містить p елементів. В застосуваннях p зазвичай є ступенем 2 (2^m) або непарним простим числом (p).

Проблема дискретного логарифму в групі точок еліптичної кривої (ПДЛЕК) полягає в наступному: маючи еліптичну криву E визначену над полем F_p, точку $G \in E(F_p)$ порядку n, і точку $Q \in E(F_p)$, необхідно визначити ціле число $l, 0 \le l \le n-1$, таке що $Q = lG$, за умови, що таке ціле число існує.

Базуючись на складності цієї проблеми, було запропоновано використання групи точок еліптичної кривої визначеної над скінченним полем для реалізації різних криптосистем. Один з таких протоколів є аналог DSA на еліптичних кривих, алгоритм ECDSA. Таким чином, ПДЛЕК може розглядатися як схожа на проблему ПДЛ, але в іншому алгебраїчному середовищі.

З 1985 року ПДЛЕК привернула значну увагу провідних математиків з усього світу. Алгоритм Похліга та Хеллмана [3,5] скоротив визначення l до визначення кожного з множників n по модулю l. Тобто, щоб досягнути максимального рівня безпеки, необхідно щоб n було простим числом. Кращим алгоритмом загального спрямування для вирішення ПДЛЕК на даний момент є метод Полларда, який з прискоренням запропонованим Галлантом, Ламбертом та Венстоном, має близько $\sqrt{\pi n}/2$ кроків, де під кроком маємо на увазі операцію додавання в групі точок еліптичної кривої.

Найважливішим є те, що для ПДЛЕК не відомо жодного алгоритму типу індексного обчислення на відміну від ПДЛ. Тому вирішення ПДЛЕК вважається складнішим ніж проблеми факторизації цілих чисел або ПДЛ, оскільки невідомо алгоритму загального спрямування з субекспоненційним часом виконання.

В 1991 році Менезес, Окамото та Венстон (MOV) [4] показали, як ПДЛЕК може бути скорочено до ПДЛ в розширенні полів F_p, де можна використати методи індексного обчислення. Тим не менш, цей алгоритм скорочення MOV ефективний тільки для дуже особливого класу кривих, відомих як суперсингулярні криві. Крім того, існує простий тест, за допомогою якого можна перевірити, що окрема еліптична крива не є вразливою до цієї атаки. Суперсингулярні криві є спеціально забороненими в усіх стандартах з використанням систем на ЕК, таких як IEEE P1363, ANSI X9.62, та ANSI X9.63.

Іншим так званим слабким класом еліптичних кривих є так звані аномальні криві – це криві E визначені над полем F_p, котре має кількість точок, що дорівнює точно p. Атака на ці криві була винайдена незалежно Семаєвим, Смартом та Сато і Аракі, та узагальнена Рюком. Як і у випадку з суперсингулярними кривими, існує простий тест, за допомогою якого можна перевірити, що окрема еліптична крива не є вразливою до цієї атаки.

Дослідження ПДЛЕК та пов'язаних з нею проблем та пошук нових рішень продовжується і на цей час. Необхідно відзначити, що в вище описаних програмних та апаратних атаках, обчислення єдиного дискретного логарифму в групі точок еліптичної кривої має ефект розкриття одного закритого ключа користувача. Ті самі зусилля необхідні для визначення другого закритого ключа користувача.

На графіку на рис. 4 порівнюється час, потрібний для злому ECDSA з часом необхідним для злому RSA або DSA для різних розмірів ключа з використанням найкращих відомих алгоритмів [5]. Значення обчислені в MIPS роках. MIPS рік представляє час обчислення, що дорівнює 1 року на машині спроможній виконувати 1 мільйон інструкцій в секунду. Для порівняльного тесту, загальноприйнятим є те, що 10^{12} MIPS років представляє значну безпеку на даний момент, оскільки таке значення потребує, щоб більшість обчислювальної потужності на планеті працювала значний проміжок часу. Щоб досягти такого рівня безпеки RSA та DSA необхідно використовувати ключ розмі-

ром 1024 біт, в той час як для алгоритму ECDSA достатньо ключа розміром 160 біт.

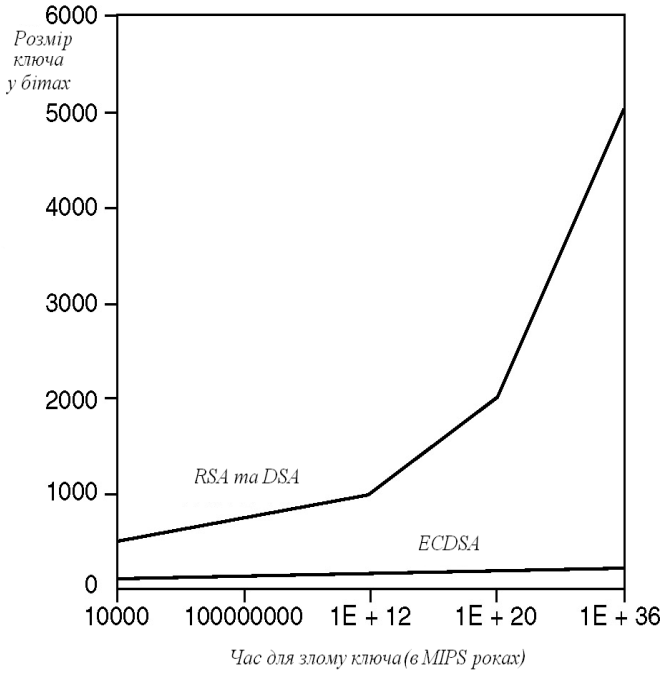


Рис. 4. Порівняння рівнів безпеки криптосистем ЕЦП

Зауважимо, що різниця між системами зростає по мірі збільшення розміру ключа. Наприклад, помітимо, як значення обсягу обчислень зростає для ECDSA з ключем розміром 300 біт порівняно з RSA та DSA з ключами розміром 2048 біт.

Порівняння трьох складних математичних проблем, на яких базуються відомі асиметричні криптосистеми підпису виявило той факт, що жодна з них не є доказово надійною. Роки інтенсивних досліджень призвели до загальноприйнятого переконання, що ПДЛЕК значно складніша ніж проблема факторизації цілих чисел та ПДЛ. Загальним висновком є те, що ПДЛЕК потребує повністю експоненційного часу для вирішення.

Розглянувши основні схеми постановки електронного цифрового підпису та аналізу їх переваг та недоліків, отримані результати можна звести до наступної табл. 1 для значення 10^{11} MIPS років необхідних для злому закритого ключа.

5. Принципи та особливості реалізації системи ЕЦП

Була розроблена система ЕЦП, що реалізує алгоритм ECDSA, так як даний алгоритм має ряд переваг, описаних вище, у порівнянні з іншими алгоритмами ЕЦП.

До розроблюваної системи ставились наступні вимоги:

- Можливість встановити порядок p еліптичної кривої у діапазоні [3; 2^{521}], що надає гнучкі засоби для вивчення та дослідження алгоритму;
- Можливість використання так званих аномальних та суперсингулярних кривих для дослідження їх уразливості;
- Побудова графіку еліптичної кривої невеликого порядку, наочне виконання операцій над точками побудованої еліптичної кривої;
- Поетапне відображення процедур постановки та верифікації ЕЦП;
- Можливість підстановки зміненого документу та його підпису на етапі верифікації підпису у цілях дослідження.

На рис. 5 представлена блок-схема реалізованого алгоритму постановки ЕЦП на основі алгоритму ECDSA.

Табл. 1. Порівняння сучасних алгоритмів цифрового підпису

Ознака порівняння	Алгоритм ЕЦП		
	RSA	DSA	ECDSA
Проблема що лежить в основі алгоритму	ПФЧ	ПДЛ	ПДЛЕК
Розмір системних параметрів в бітах	1024	2208	481
Розмір отриманого підпису в бітах	1024	320	320
Розмір відкритого ключа в бітах	1088	1024	161
Розмір закритого ключа в бітах	2048	160	160

У блоках 3 і 6 блок-схеми, представленої на рис. 5, виконується операція множення точки еліптичної кривої на число. Дана операція потребує значної кількості обчислень, так як в математичному апараті еліптичних кривих над скінченим полем не існує операції множення точки на число, більше 2, тому необхідне багаторазове виконання операції додавання точок.

При такому підході у разі виконання операції множення точки P на число k, необхідна одна операція знаходження точки 2P та k - 2 операцій додавання, які потребують великої кількості часу при значеннях числа k порядку 2^{160} і більше.

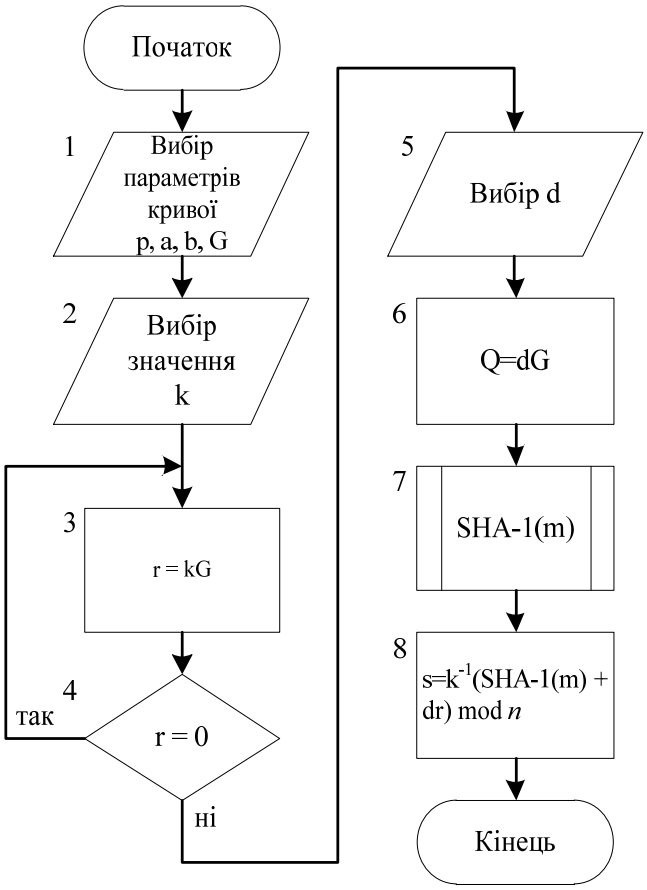


Рис. 5. Блок-схема реалізованого алгоритму постановки ЕЦП

На заміну даного алгоритму множення точки на число було використано наступний алгоритм. Для виконання операції множення точки P на число k послідовно знаходяться точки 2P, 2(2P) = 4P, 2(4P) = 8P, …, 2^iP,…, $2(2^{n-1}P) = 2^nP$, де n – кількість розрядів числа k у двійковому представленні. Точка kP знаходиться як сума тих точок 2^iP, для яких i-й розряд числа k у двійковому представленні рівний 1. Нехай двійкове представлення числа k містить r одиниць, тоді для виконання операції множення точки на число необхідно r - 1 операцій додавання та n - 1 операція множення точки на 2. У найгіршому для швидкодії випадку, коли r = n, необхідні n - 1 операція додавання та n - 1 операція множення точки на 2. Так як число n не перевищує $\log_2 k + 1$, досягається висока швидкість виконання операції множення точки на число. Блок-схема алгоритму множення точки на число представлена на рис. 6.

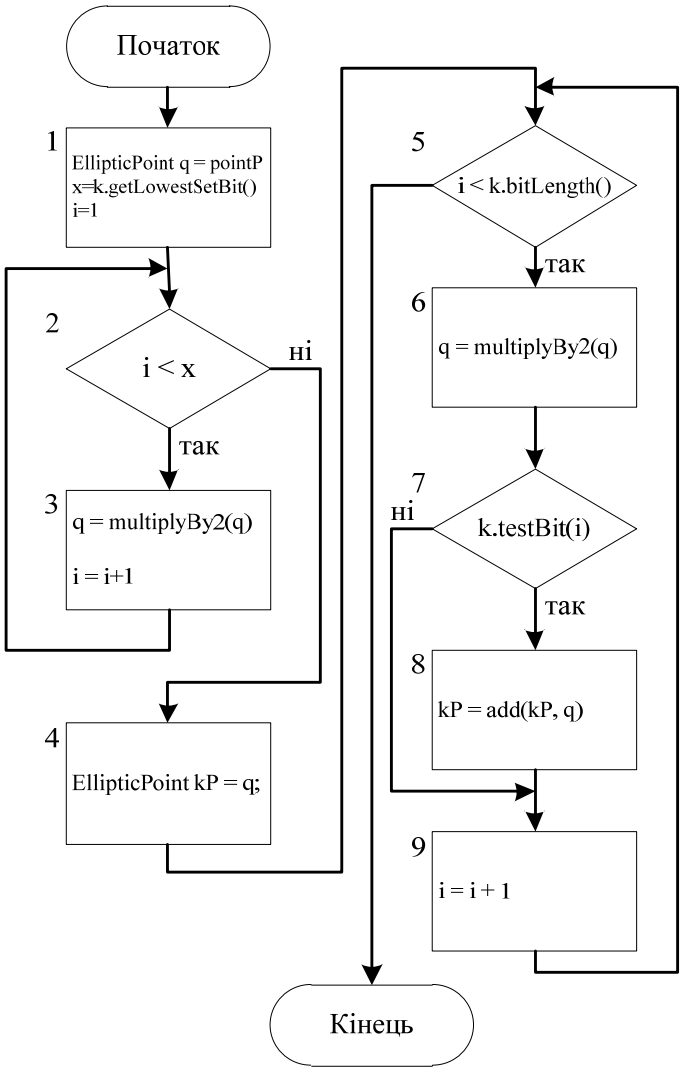


Рис. 6. Блок-схема алгоритму множення точки еліптичної кривої на число

6. Висновок

У даній роботі були досліджені сучасні схеми постановки електронного цифрового підпису та проведений порівняльний аналіз найбільш поширених схем.

В результаті дослідження схем постановки електронного цифрового підпису була отримана таблиця, у якій вказано розміри параметрів, що використовуються в обчисленнях, зберігаються на носіях а також передаються по каналам зв'язку. Згідно з отриманими результатами, алгоритм ECDSA при менших розмірах параметрів, що використовуються в обчисленнях та передаються по каналу зв'язку, здатен забезпечити більшу криптостійкість порівняно з іншими алгоритмами ЕЦП. Більша криптостійкість забезпечується завдяки використанню пробле-

ми знаходження дискретного логарифму в групі точок ЕК (яка аналогічна проблемі пошуку закритого ключа), для якої досі не знайдено алгоритму з реальним часом пошуку результату.

На основі алгоритму ECDSA реалізовано систему створення та верифікації ЕЦП на базі еліптичних кривих, що, на відміну від існуючих реалізацій, дозволяє варіювати параметри еліптичної кривої в широких межах, та надає засоби їх дослідження, серед яких побудова графіку точок еліптичної кривої з заданими параметрами, відображення результатів операцій над точками графіку, поетапне виконання постановки та верифікації ЕЦП, можливість використання суперсингулярних та аномальних еліптичних кривих для досліджень їх недоліків. Існуючі розробки були використані для створення учбового макету та програми для тестування знань в області еліптичних кривих та постановки ЕЦП. Також було досягнуте значне підвищення швидкості операції множення точки еліптичної кривої на число за рахунок використання покращеного алгоритму.

Порівняльний аналіз існуючих схем ЕЦП показав, що саме реалізований алгоритм підпису ECDSA є найбільш перспективним, так як довжина його ключа з підвищенням вимог до криптостійкості не так стрімко зростає, як у випадку інших існуючих алгоритмів, таких як RSA та DSA.

Список літератури

1. ДСТУ4145-2002. Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих. Формування та перевірка, Київ: Держстандарт України, 2003.
2. Serge Vaudenay. A Classical Introduction to Cryptography: Applications for Communications Security, Springer, 2006.
3. D.Johnson and A.Menezes. The Elliptic Curve Digital Signature Algorithm (ECDSA), Univ. of Waterloo, 1999.
4. D.Johnson and A.Menezes. The Elliptic Curve Cryptosystem. Remarks on the security of the elliptic curve cryptosystem, Univ. of Waterloo, 2000.
5. Harold F. Tipton and Micki Krause. Information Security Management Handbook, Sixth Edition, 2006.

УДК 681.3.06

ПУСТОВАРОВ В.И.,
БОЙКО В.В.,
ЛЕМЕШКО В.А.

ФУНКЦИОНАЛЬНЫЕ РАСШИРЕНИЯ ПРОМЫШЛЕННЫХ АВТОМАТИЗИРОВАННЫХ СИСТЕМ КОНТРОЛЯ И УПРАВЛЕНИЯ

Представлены особенности системы тестирования, контроля и управления наличными ресурсами элементов производства и требования, используемые в разных технических отраслях хозяйства. Рассмотрены основные свойства подсистем измерения и накопления результатов испытаний от отдельных технических элементов производства до подсистем сейсмического анализа состояния конструкций в подземных производственных комплексах. Отмечены проблемы подключения подсистемы к действующей системе и предложены пути расширения функциональных возможностей системы.

The peculiarities of automated industrial control systems and the requirements to the seismic subsystem are considered. The problems of insertion the seismic subsystem to the system operating are defined. The way of system functional extensions are offered.

Введение

Широкий круг систем автоматизированного контроля и управления для сбора, накопления данных и принятия по ним оперативных решений не требуют дорогостоящего оборудования, но должны обеспечивать достаточную работоспособность и надежность в сложных условиях эксплуатации. Для предотвращения аварий в разных отраслях промышленности и транспорта созданы автоматизированные системы, которые путем контроля и проведения испытаний оборудования в условиях реальной эксплуатации и усложненных условиях обеспечивают безопасные условия работы персонала. В то же время, наука и техника систем оперативного контроля и проведения испытаний постоянно развивается, предлагая новые решения для обеспечения безопасной эксплуатации транспорта, работы шахтеров под землей и во многих других отраслях промышленности и хозяйства [1-4].

Одним из наиболее распространенных методов реализации оперативного контроля и испытаний является применение открытых SCADA-систем (от Supervisory Control And Data Acquisition). Обмен данными в таких системах организован в соответствии со стандартом RS-485 в протоколе компьютерной сети ModBus. Протокол ModBus основан на принципе подчинения master-slaves, т.е. сервер SCADA выдает контроллеру адресный запрос на выдачу данных с его регистров. Контроллер с указанным адресом отвечает, при этом обмен осуществляется при помощи пакета PDU (Protocol Data Unit), который помещен в пакет ADU (Application Data Unit), который зависит от типа линии связи. Принципиальным недостатком протокола ModBus является невозможность инициализации связи для подчиненного устройства (slaves) с сервером (master). Необходимо ждать своей очереди опроса и частота опроса должна быть достаточной, чтобы обеспечить приемлемое время получения информации. В принципе любая передача информации между любыми двумя устройствами возможна лишь с участием master-устройства (сервера). Это существенно ограничивает применение ModBus в системах реального времени, но если необходимые частоты обмена посылками не превосходят десятка герц, то с помощью указанного интерфейса можно обеспечить своевременную передачу информации.

Типовые области применения

В общем случае любая автоматизированная система контроля и испытаний состояния базовых элементов промышленных объектов включает совокупность датчиков и приборов для оценки предаварийного состояния, алгоритма выработки решения в зависимости от набора угроз аварий и разрушений. Методы и средства сбора и накопления информации, реализуемые в таких системах, опираются на расходование части ресурсов ключевых элементов для предотвращения упомянутых критических угроз. Накопление данных и объективное до-

кументирование состава испытаний, состояния объекта и оценки системой наличия угроз и выявление их причин являются обязательными для накопления в ретроспективной базе данных (БД). По такому принципу работают автоматизированные системы на транспорте, производстве, строительстве и т.д.

Обязательной экономической основой такой системы является превышение затрат на испытания над стоимостью затрат ресурсов. Это должно обеспечить экономическую эффективность производства при достаточной защите от катастрофических последствий аварий и разрушений. Практически единственным средством предотвращения таких последствий является своевременное проведение испытаний и накопление достоверной информации об их результатах во встроенных БД автоматизированных систем.

Одним из наиболее важных применений такой автоматизации связано с добычей угля, которая в нашей стране относится к одной из самых опасных отраслей промышленности, Взрывы метана и горные удары, шахтные выбросы и обрушения – это далеко не полный перечень опасностей, подстерегающих шахтеров на глубинах, зачастую, свыше одного километра. Автоматизированные системы контролируют технологические участки шахт, работу оборудования и условия работы шахтеров. В большинстве шахт функционируют системы контроля за технологическим оборудованием (транспортерами, комбайнами, вентиляторами, подъемниками, их компонентами и т.д.). Стал актуальным, как никогда, модульный (блочный) принцип построения систем с возможностью оперативного расширения функциональности программного обеспечения. Если к такой системе добавить подсистему контроля за состоянием горного массива и воздуха в производственных помещениях, подсистему прогноза внезапных выбросов и горных ударов, подсистему определения местоположения персонала, то безопасность труда шахтеров только повысится. В шахтах Донбасса работает много подобных систем разных производителей. Часто они решают разные, иногда и одинаковые задачи, но по-разному: есть еще проблемы в прогнозировании внезапных выбросов и неслучайно предлагаются различные способы прогнозирования таких явлений и определения периодичности испытаний с использованием пассивных и активных сейсмических технологий [1-4].

В качестве примера реализации и использования результатов исследований рассмотрим анализ и определение направленности модернизации существующей унифицированной телекоммуникационной автоматизированной системы (УТАС) [11] приборами сейсмической подсистемы. Система УТАС – это многофункциональная унифицированная телекоммуникационная система диспетчерского контроля и автоматизированного управления машинами и технологическими комплексами, созданная в 2004 году на базе открытой SCADA-системы [11].

Сейсмическая подсистема предназначена для фиксации сейсмических событий для повышения безопасности шахтных работ. Для этого сейсмическая подсистема оперативно оценивает мощность сейсмического события, местоположении очага события (***x, y, z***), идентифицирует событие и вырабатывает, при превышении заданной величины мощности, сигнал «АВАРИЯ» при котором аварийно отключается электрическая энергия в подземном пространстве шахты. Для этого используются показания сейсмических датчиков, установленных в шахте. Кроме этого, подсистема экспортирует информацию о сейсмических событиях в систему УТАС для хранения, накопления и последующего анализа статистическими методами. Отдельной задачей является идентификация событий по форме сигналов, полученных в результате испытаний. Для этого в подсистеме должна накапливаться БД амплитудно-частотных характеристик технологических шумов от шахтных машин и механизмов.

Аппаратно подсистема состоит из сейсмических приемников (сейсмодатчиков), геофонов, контроллеров, аппаратуры приема-передачи, линии передачи (канала связи), сервера обработки и связи с системой УТАС. Требования к численным характеристикам и параметрам устройств, сформированные из задач подсистемы представлены в Табл. 1.

Остановимся на некоторых требованиях. Разрядность аналого-цифрового преобразователя датчика – это, прежде всего, разрешение датчика. Чтобы шумы (помехи, производственные колебания машин и механизмов) не смогли

подавить полезный сигнал, этот показатель (***R***), должен быть не менее 24 бит [5]. А, учитывая, что датчик работает в одном диапазоне, условие ***R***=24 должно быть выполнено. Следующие два параметра, F_v и ***dt***, относятся к разрешению датчика по времени. Для отыскания очага сейсмического события используется метод взаимной корреляции, в котором используется ***dt***, поэтому, чем меньше будет это время, тем точнее будут вычислены координаты очага. Параметры F_0 , ***V*** и ***Q*** характеризуют пропускную способность информационного тракта (канала) и определяется количеством датчиков ***N***, интервалом дискретизации датчика ***dt***, количеством проходящей информации в секунду и реакцией системы на угрожающее событие t_c .

Таблица 1

Основные требования к устройствам сейсмической подсистемы

№ п.п.	Уровень	Устройство	Параметр	Обозначение	Требование	Примечание
1	Нижний	Датчик	Разрядность, бит	*R*	24	
2	Нижний	Датчик	Полоса частоты регистр., Гц	F_v	0-1000	
3	Нижний	Датчик	Дискретизация, сек.	*Dt*	0.0001	$F_d = 1/dt$
4	Средний	Контроллер	Частота опроса датчиков, Гц	F_o	$F_o = F_d \cdot (N+2)$	*N* – колич. Датчиков
5	Средний	Контроллер	Скорость передачи, кадр/ с	*V*	10 000	
6	Верхний	Компьютер	Реакция системы на угрожающее событие, с.	t_c	0.1	
7	Верхний	Хранилище	Объем хранимой информации в сутки	*Inf*	$86400 \cdot F_o \cdot K$	
8	Верхний	Компьютер	Количество обработанной информации в с.	*Q*	Не менее 1.5МБ/с	

Время реакции системы t_c состоит из суммы времен преобразования АЦП датчика, передачи в контроллер, формирования слова и передачи по линии связи, потере времени в линии, в ретрансляторе, вводе в персональный компьютер, время, затраченное на преобразование и сравнение с критической величиной, время формирования и выдачи сигнала “авария”. Достаточно приблизительно $t_c = t_d + t_{перед.} + t_{анализа} + t_{сигнал.}$ (где t_d – время, затраченное датчиком, $t_{перед}$ – время, затраченное на передачу, $t_{анализа}$ – время, затраченное на анализ, $t_{сигнал}$ – время, затраченное на формирование сигнала “авария”. Возможны два варианта хранения: первый – с накоплением полной информации, второй – с пропусками записей при отсутствии сейсмических событий с энергией выше пороговой. Считаем, что на этом уровне нужно сохранять всю информацию в течение суток. Константа 86400 – количество секунд в сутках. С учетом времени дискретизации ***dt***, общее количество запомненной за сутки информации $N_{зс}$ составляет: $N_{зс}$(слов) = F_d 86400 ***N***. Общее количество датчиков зависит от характеристик шахты и составляет не менее 25 трехкоординатных сейсмодатчиков [6], т.е. минимальное число ***N*** = 75.

Взаимодействие сейсмической подсистемы с системой УТАС показано на Рис.1.

Рис.1. Взаимодействие сейсмической подсистемы с системой УТАС

Вся обработка сейсмической информации производится в сейсмической подсистеме (ввод данных от сейсмических датчиков и их анализ, проверка данных на критичность, вычисление местоположения сейсмического события, вычисление объема выделенной энергии, иденти-

фикация события, отображения события, формирование записи для экспортирования в систему для хранения и передача этой записи в хранилище системы). Подсистема имеет два выхода в систему:

- Для передачи сигнала АВАРИЯ в систему для сигнализации и отключения электроэнергии в подземном пространстве средствами системы.
- Для передачи информации о сейсмическом событии в специальный комплекс сбора данных для длительного хранения.

Для вычисления координат очага сейсмического события используются два метода: метод эпицентралей δt [7], при котором координаты очага возмущения находятся при решении системы уравнений.

$$(x_i - x)^2 + (y_i - y)^2 + (z_i - z)^2 = V^2 \cdot \delta t_i^2,$$

где x_i, y_i, z_i – координаты сейсмического датчика, x, y, z – координаты сейсмического очага, V – скорость продольной волны, δt_i – разность времени прихода волны на станцию и времени события в очаге, i – количество точек наблюдения (не менее 6).

Как видно, этот метод основан на измерении скорости распространения продольных волн в среде. Второй метод – это метод азимутов [7]. В нем по показаниям сейсмодатчиков вычисляются коэффициенты A и B уравнения прямой вида $y = Ax + B$ для каждого пункта установки датчиков. Место пересечения направляющих дает координаты очага возмущения. При удовлетворительном совпадении результатов считается, что координаты сейсмического очага найдены.

Энергия сейсмического события вычисляется, исходя из результатов обработки данных с сейсмодатчиков. В [8] представлены результаты исследования нахождения энергии сейсмических событий по данным, полученным с сейсмодатчиков. При этом, анализируя амплитуды сейсмоколебаний, предложено уравнение.

$$A_1 / A_2 = (r_2 / r_1)^n \cdot e^{-\acute{a}\, ras},$$

где A_1 , A_2 - амплитуды сейсмических импульсов, зафиксированные датчиками; r_1, r_2 – расстояния от источника до сейсмоприемника; n – показатель степени геометрического расхождения; $\acute{a}$ - коэффициент поглощения; $ras = r_1 - r_2$.

Коэффициенты n и $\acute{a}$ находятся методом наименьших квадратов. Подставив их значения в уравнение $A = A_0 \cdot r^{-n} \cdot e^{-\acute{a} \cdot r}$ и решив относительно A_0, можно найти амплитуду в эпицентре возмущения. В общем случае, форма импульса будет изменяться по закону $A = A_0 \cdot e^{-\beta \cdot t} \cdot \sin(\omega_0 \cdot t)$, где β – коэффициент затухания, ω_0 – круговая частота.

Энергия волны в точке регистрации вычисляется по формуле Рейли [9]:

$E = \sum(A_f^2 \cdot \Delta f)$, где A_f – значение спектральной плотности скорости смещения на соответствующей частоте. Закон ослабления можно применить и к энергии [8], т.е. тогда $E = E_0 \cdot r^{-n} \cdot e^{-\acute{a} \cdot r}$.

Типовые функции прогноза

В подсистему входит программа статистического анализа, которая выдает прогноз (в виде вероятности) выброса или горного удара в зависимости от характеристик сейсмического события, зафиксированного сейсмическими датчиками. Для этого обрабатывается материал БД, находятся подобные характеристики с заданными допусками и вычисляется вероятность появления вторичного динамического сейсмического события (выброса или горного удара) исходя из соотношений количества сейсмособытий, которые произошли в прошлом SP к общему количеству сейсмособытий с подобными характеристиками $(SP + SN)$ в этом забое, т.е. $P(S) = SP/(SP+SN)$, где $P(S)$-вероятность вторичного динамического события, SN – количество измерений, когда вторичных динамических событий не произошло [10]. Эта информация необходима, прежде всего, при проведении сотрясательных взрывов, когда необходимо разгрузить напряженные участки пласта или породы, а в первые минуты после сотрясательного взрыва ни выбросов, ни ударов не произошло.

По результатам работы сейсмической подсистемы, при превышении характеристик сейсмического события, в систему передаются экстренный сигнал “авария”, мощность сейсмического события, его координаты и результаты замеров всех сейсмических датчиков для долговременного хранения.

Выводы

Из вышеперечисленного можно сделать вывод, что при модернизации автоматизированной системы управления путем добавления к ней новых подсистем, необходимо, чтобы все новые (добавляемые) функции выполнялись в рамках подсистемы, т.е. подсистема должна быть максимально автономна. В то же время необходимо предусмотреть средства и принци-

пы для обеспечения экстренного и стандартного входа в базовую (основную) систему. Подключение дополнительных подсистем существенно повысит безопасность подземных работ в шахтах, позволит оперативно анализировать результаты сотрясательных и технологических взрывов, что потребует дополнительной функциональности и гибкости в специально разрабатываемых программных модулях. Предложенные методы модернизации позволяют подключать к существующим автоматизированным системам подсистемы: сейсмического анализа и прогноза внезапных выбросов и горных ударов, контроля нахождения шахтеров и контроля состава воздуха, контроля за динамикой выделения метана, аварийного отключения электроэнергии и т.д. Отметим дополнительные требования к разработке программного обеспечения:

1. Для повышения уровня автоматизации и достоверности накопленных данных испытаний во встроенной БД системы контроля необходимо хранить данные о контролируемых ресурсах и данные от датчиков физических величин и распознанные приборами образы ситуаций.

2. Программное обеспечение таких систем и их подсистем должно работать с управляющими и информационными данными стандартных внутренних форматов, предоставляющих возможность автоматической настройки на расширение входного набора данными подсистем и частичное сохранение работоспособности при авариях.

3. Основными частями ключевых элементов поиска образов критических событий, должны быть функционалы статистик, ограничений и экстремумов от выборок значений характеристик объекта на разных интервалах времени.

4. Использование статистических функционалов критериев доверия и ретроспективного анализа статистик позволяют уточнять прогнозы, задавая эффективную периодичность активных экспериментов, требующих значительных производственных ресурсов.

Список ссылок

1. Мещанинов С. К., Король В.И. Имитационная система промышленной безопасности угольных шахт//Науковий вісник НГУ, 2010. – С.19-22.
2. Александров П.А., Калечиц В.И., Маслаков О.Ю. и др. Система раннего обнаружения аварийных и предаварийных состояний на основе мониторинга микро- и наночастиц. (www.ntsr.info/science /library/3196.htm)
3. Зберовский В.В., Опрышко Ю.С., Поляков Ю.Е. Методические основы контроля аппаратурой ЗУА-98 эффективности гидроимпульсного воздействия: Материалы XXI Межд. науч. школы. – Алушта – 2011. – С.136-137.
4. Захаров В.Н., Харченко А.В. Методика автоматизированного прогноза изменения напряженного состояния углепородного массива при горных работах: Материалы XVIII Межд. науч. школы. – Алушта – 2008. – С.108-111.
5. Сато Ю. Обработка сигналов. Первое знакомство. М.: Издат.дом “Додэка-XXI”, 2002. – 176с.
6. Аверин А.П., Захаров В.Н., Харченко А.В. Аппаратно-программный комплекс нового технического уровня для проведения шахтных сейсморазведочных работ ASD// Деформирование и разрушение материалов с дефектами и динамические явления в горных породах и выработках: XXI международная научная школа им. ак. С.А. Христиановича, 15-25 сентября 2012г. Материалы конференции. С.:ТНУ, 2012. - с.10-13.
7. Андреев С.С. О методах интерпретации близких землетрясений.//Труды института физики Земли, №25(192), М.:ИФЗ, 1962. – с.226-311.
8. Анцыферов М.С. Исследование местоположений очагов, энергии и частотных спектров сейсмоакустических импульсов, возникающих в процессе разработки угольного пласта. Краткий научный отчет. М.: ИГД, 1967. – 127с.
9. Харкевич А.А. Спектры и анализ. М.: Наука, 1957. – 127с.
10. Гнеденко Б.В., Хинчин А.Я. Элементарное введение в теорию вероятности. М.: Наука, 1964. – 164с.
11. Унифицированная телекоммуникационная автоматизированная система. http:www.itras.com.ua

УДК 004.942 (519.217)

ПРОШИН С.Ю.

ОЦІНКА МАТРИЦЬ ПЕРЕХОДІВ ЕНЕРГОНАВАНТАЖЕНЬ НА БАЗІ ПЕРІОДИЧНОГО ЛАНЦЮГА МАРКОВА

В роботі досліджується енергонавантаження з врахуванням його марковості та стохастичної періодичності. В якості математичної моделі запропоновано використовувати періодичний ланцюг Маркова. Описано алгоритм знаходження оцінок матриць переходів. В його основі лежить виділення із періодичного ланцюга елементів однорідності, що дозволяє використати існуючі методи статистичної обробки. Проведено визначення станів системи енергоспоживання і періоду ланцюга з врахуванням особливості енергонавантажень протягом доби і в робочі та вихідні дні. Створено програмне забезпечення, яке реалізує розроблений алгоритм, проведено оцінювання матриць переходів енергонавантажень. Отримані оцінки матриць відображають ймовірнісні особливості поведінки графіків споживання електроенергії і можуть бути використані в задачах прогнозу.

In this paper we research the power supply taking into account its Markov's properties and its stochastic periodicity. As the mathematical model it is suggested to use the periodic Markov chain. Algorithm of finding of estimation of transition matrices is described. In its basis lays the separation of homogeneous elements out from the periodic chain, which allows to use existing methods of statistical processing. We conduct the definition of states of the power supply system and the period of chain with the account of peculiarity of power supply during the day at workdays and weekends. The software on the basis of developed algorithm is created and the estimation of matrices of transition is conducted. Received estimations of matrices reflect the probable peculiarities of behavior of electric power supply graphs and can be used in problems of prognosis.

Вступ

В прикладних дослідженнях часто приходиться мати справу із стохастично- періодичними сигналами та явищами. Як приклад такого сигналу, на рисунку 1 наведено погодинний графік енергонавантаження першого навчального корпусу Тернопільського національного технічного університету ім. І.Пулюя протягом п'яти робочих днів в жовтні 2008 року.

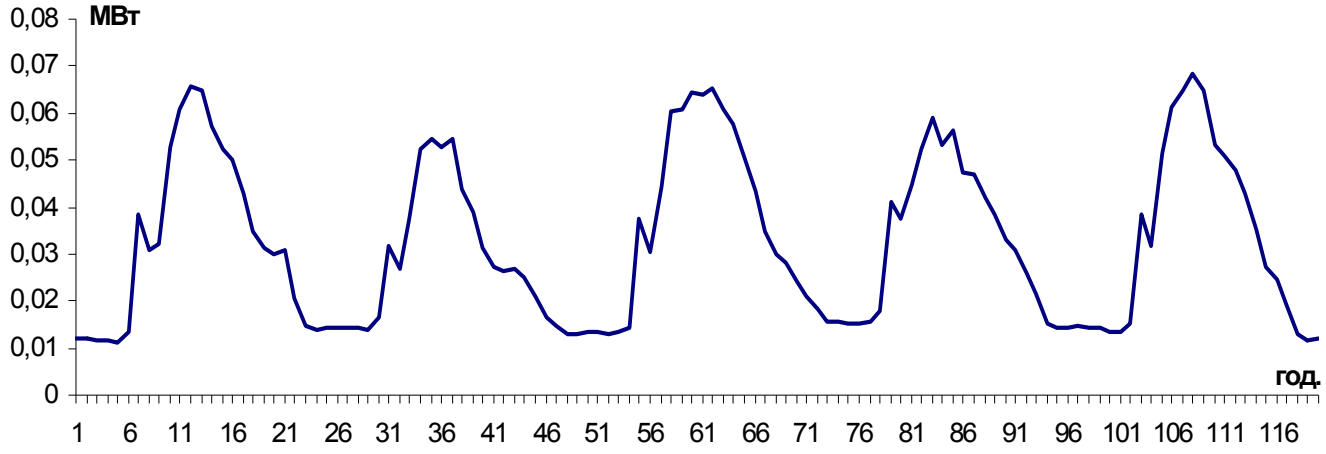


Рис. 1. Погодинний графік споживання електроенергії

Візуальний аналіз графіку показує, що спостерігається «приблизна повторюваність» споживання електроенергії через кожні 24 години. Під стохастичною періодичністю розуміється, що хоча для графіків детермінована періодичність відсутня, але при цьому припускається, що періодично змінюються певні ймовірнісні характеристики сигналу, в даному випадку енергонавантаження

Для дослідження сигналів використовується трьохкроковий підхід «модель-алгоритм-програма». На першому кроці обґрунтовується модель сигналу, в якій враховуються його основні його властивості. На другому кроці на

базі моделі розробляються методи і алгоритми обробки сигналів. На третьому – створюється програмне забезпечення для реалізації розроблених алгоритмів.

На даний час в якості математичних моделей стохастично-періодичних сигналів використовувалися періодично корельовано випадкові процеси, періодичні процеси та лінійні періодичні процеси [1].

Перші дві моделі – періодично корельовано та періодичні випадкові процеси є описовими, тобто лише враховують характер зміни ймовірнісних характеристик, лінійні періодичні процеси враховують фізику утворення енергонавантажень, і тому дозволяють більш всесторонньо вивчати їх особливості.

В [2] запропоновано використання моделі, що враховує як стохастичну періодичність так і марковість досліджуваного сигналу у вигляді періодичного марківського процесу. В [3] введено поняття періодичного ланцюга Маркова, який пропонується використати у якості математичної моделі енергонавантаження. Основна причина такого підходу полягає в тому, що енергосистему в цілому чи її окремий енергорегіон можна розглядати як систему масового обслуговування. А вхідні потоки такого роду систем переважно є частинним випадком ланцюгів Маркова. Виникненню періодичності ланцюгів сприяє ритмічний характер функціонування енергорегіонів.

В роботах [4]-[6] проведено теоретичні дослідження, щодо створення методів статистичної обробки сигналів на базі моделі у вигляді періодичного ланцюга Маркова, які полягають у знаходженні оцінок матриць переходів. Однак розроблені методи оцінювання на реальних сигналах не використовувались.

Мета роботи. Знайти оцінки матриць переходів енергонавантажень, використовуючи в якості їх математичної моделі періодичний ланцюг Маркова.

Періодичний ланцюг Маркова та його властивості

Нагадаємо поняття періодичного ланцюга Маркова [3] та його властивості [4].

Означення. Ланцюг Маркова

$$\{\xi_n\},\ n = 0,1,2,... \qquad (1)$$

називається періодичним, якщо періодичними є його ймовірності переходів, тобто існує ціле $L > 1$, що

$$p_{ij}^{(k)} = p_{ij}^{(k+L)},\ i, j \in X, \qquad (2)$$

де i, j – стани, $i, j \in X$.

Очевидно, що для періодичного ланцюга Маркова його матриці переходів $\Pi(k) = \left| p_{ij}^{(k)} \right|$ теж змінюються періодично з цим же періодом L:

$$\Pi^{(k)} = \Pi^{(k+L)},\ k = 0,1,... \qquad (3)$$

Із означення також випливає, що періодичний ланцюг визначається першими L матрицями переходів

$$\left\{\Pi_0^{(0)},...,\Pi_0^{(k)},...,\Pi_0^{(L-1)}\right\}. \qquad (4)$$

Множину чисел $0,...,k,...L-1$, де L – період ланцюга, позначимо через φ:

$$\varphi = \{0,...,k,...,L-1\}. \qquad (5)$$

Множина (5) називається φ-множиною фаз, окремі числа k цієї множини – фазами. Відповідно до (5) множину матриць (4) називається множиною фазових матриць, а окремі матриці $\Pi_s^{(k)} = \Pi_0^{(k)},\ s = 1,2,...$ – фазовими матрицями. Кожна із матриць $\Pi_s^{(k)}$ утворюється ймовірностями (2).

$$\Pi_s^{(k)} = \begin{vmatrix} p_{11}^{(k)} & ... & p_{1m}^{(k)} \\ ... & ... & ... \\ ... & p_{ij}^{(k)} & ... \\ ... & ... & ... \\ p_{m1}^{(k)} & ... & p_{mm}^{(k)} \end{vmatrix}, \qquad (6)$$

де m – кількість станів системи, $k \in \varphi$ - множині фаз.

Щоб перейти до питання оцінки матриць переходів з використанням реалізацій енергонавантажень необхідно розглянути ще ряд питань.

Визначення станів енергонавантажень і періоду ланцюга

Важливим кроком аналізу енергонавантажень є необхідність вирішення питання визначення станів, тобто фіксованих значень одного або декількох параметрів системи. В якості окремого стану доцільно використати інтервал в який попадає значення кількості спожитої електроенергії за певний проміжок часу.

Зазвичай розглядається періодичний ланцюг Маркова із скінченим числом станів. Щодо енергонавантажень, то їх значення неперервні в певному діапазоні. Для отримання скінченого числа станів поступимо наступним чином. Проаналізуємо значення енергонавантажень за ча-

совий період на якому проводимо оцінювання матриць переходів. Знайдемо максимальне ($E_{\max}$) та мінімальне ($E_{\min}$) значення, спожитої за визначений проміжок часу електроенергії. Діапазон зміни значень $[E_{\min};E_{\max}]$ розіб'ємо на m рівних відрізків, кожний довжиною:

$$\Delta E = \frac{E_{\max} - E_{\min}}{m}. \qquad (7)$$

В результаті отримаємо сукупність інтервалів:

$$\{[E_0;E_1),\ [E_1;E_2),\ \ldots\ [E_{i-1};E_i),\ \ldots\ [E_{m-1};E_m]\} \qquad (8)$$

,

де $[E_{i-1};E_i) = E_{\min} + i\Delta E$

Кожному з інтервалів (8) поставимо у відповідність число x_i, $i = \overline{1,m}$, яке буде визначати стан системи. Якщо в деякий фіксований момент часу t_k значення реалізації енергонавантажень $E(t)$ попадає в інтервал $[E_{i-1};E_i)$ із сукупності (8), то будемо вважати що система знаходиться в стані x_i. Числа x_i, $i = \overline{1,m}$ утворюватимуть фазовий простір системи $X = (x_1,\ldots\ x_i,\ldots\ x_m)$, а ймовірності переходів із стану x_i в стан x_j утворюють матрицю (6).

Проаналізувавши об'єм вибірки вибрано кількість станів системи $m = 3$.

Зауважимо, що при необхідності розбиття діапазону $[E_{\max};E_{\min}]$ може відбуватись і на нерівні інтервали. Також зазначимо, що при виборі більшої кількості станів у фазовому просторі системи модель більш точніше відображати сигнал.

Розглянемо питання вибору періоду ланцюга. Вище було наголошено, що «приблизна повторюваність» енергонавантажень відбувається через часовий проміжок $T = 24$. В нашому випадку значення спожитої електроенергії фіксується з кроком $\Delta t = 1$ год. Тому для періодичного ланцюга Маркова, як моделі енергонавантажень, період дорівнюватиме $L = T/\Delta t = 24/1 = 24$ год.

Відмінності між робочими та вихідними днями

Слід звернути увагу на те, що режими споживання електроенергії у робочі та вихідні дні суттєво відрізняється, що наглядно демонструють графіки оцінок математичного сподівання енергонавантажень з 24 години робочих та вихідних днів у жовтні 2008 року, наведені на рисунку 2.

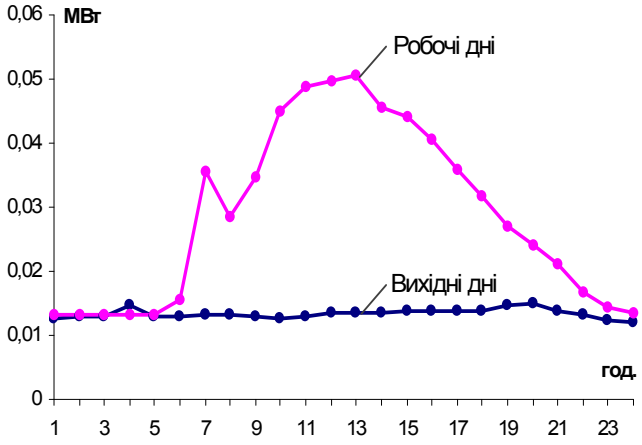


Рис. 2. Оцінки математичного сподівання енергонавантажень

Отже при оцінюванні матриць переходів енергонавантажень необхідно знаходити оцінки матриць окремо для робочих та вихідних днів.

Алгоритм оцінювання матриць переходів

Оцінювання матриць переходів проводимо згідно методу наведеного в роботі [5]

Розглянемо погодинні значення споживання електроенергії, що є реалізацією періодичного ланцюга (1).

$$\{\xi_n\},\ n = 0,1,\ldots,N. \qquad (9)$$

Оскільки період ланцюга Маркова $L = 24$, то на протязі одного циклу виділяємо 24 однорідних $\varphi^{(k)}$-послідовностей, властивості яких описаних в роботі [4]. Така $\varphi^{(k)}$-послідовність буде відповідати вибірці елементів на k-ту годину доби. Кількість відліків $\varphi^{(k)}$-послідовності дорівнюватиме кількості діб.

Із реалізації (9) згідно [4] виділимо k-послідовності пар

$$\{\hat{\xi}_s^{(k)}\},\ s = 0,1,\ldots,z,\ z = [N/L]. \qquad (10)$$

Оскільки кожна пара $\hat{\xi}_s^{(k)} = (\xi_s^{(k)}, \xi_s^{(k+1)})$, то послідовність (10) можна записати у вигляді

$$\{(\xi_0^{(k)}, \xi_0^{(k+1)}),\ldots,(\xi_s^{(k)}, \xi_s^{(k+1)}),\ldots,(\xi_{z-1}^{(k)}, \xi_{z-1}^{(k+1)})\}. \qquad (11)$$

В нашому випадку $z = 24$.

Ймовірності переходів між реалізаціями кожної із пар $\hat{\xi}_s^{(k)} = (\xi_s^{(k)}, \xi_s^{(k+1)})$ визначаються однією і тією ж фазову матрицю переходів $\Pi_0^{(k)}$.

Для того щоб оцінити елемент $p_{ij}^{(k)}, i, j = 1,2,3$, матриці $\Pi_0^{(k)}, k = 0,1,...,23$, в послідовності (11) серед перших елементів $\xi_s^{(k)}$ кожної із пар $\hat{\xi}_s^{(k)} = \left(\xi_s^{(k)}, \xi_s^{(k+1)}\right), s = 0,1,...,23$, виявляємо ті елементи, значення яких рівні i. Число таких елементів позначаємо через $N_i(k)$. Після цього серед тих пар $\hat{\xi}_s^{(k)} = \left(\xi_s^{(k)}, \xi_s^{(k+1)}\right)$, значення перших елементів яких рівні x_i, виявляємо ті пари, значення других елементів $\xi_s^{(k+1)}$ яких рівні x_j. Їх кількість позначаємо через $N_{ij}(k)$.

Оцінку перехідної ймовірності $p_{ij}(k)$ знаходимо за формулою

$$\widetilde{p}_{ij}(k) = \frac{N_{ij}(k)}{N_i(k)}, i, j = 1,2,3. \qquad (12)$$

Провівши наведені обчислення для всіх $k = 0,1,...,23$, знайдено всі оцінки $\widetilde{p}_{ij}(k)$, з яких, в свою чергу, сформовано оцінки фазових матриць переходів $\widetilde{\Pi}_0^{(0)},...,\widetilde{\Pi}_0^{(k)},...,\Pi_0^{(23)}$

Випадок появи нульових рядків в оцінках матриць переходів

Під час проведення прикладних досліджень при знаходжені оцінок матриць переходів виявилось, що деякі рядки містять тільки нульові значення ймовірностей, що не відповідає властивостям перехідних матриць ланцюгів Маркова. Причиною появи таких рядків є те, що в досліджуваній системі на певному кроці k не відбувається перехід в деякий стан r, відповідно матриця переходів $\Pi^{(k)}$ містить r-й стовбець, в якому всі значення ймовірностей переходів дорівнюють нулю, тобто $p_{ir}^{(k)} = 0$, $i = \overline{1,m}$. Як наслідок, на наступному кроці в матриці $\Pi^{(k+1)}$ всі ймовірності переходу із стану r в будь-який інший стан також дорівнюють 0, тобто для r-го рядка матриці переходів $p_{rj}^{(k+1)} = 0$, $j = \overline{1,m}$.

Рядок матриці, в якому всі значення ймовірностей переходів дорівнюють нулю називатимемо нульовим рядком, а рядок, в якому хоча б одне значення ймовірності переходу відмінне від нуля – ненульовим.

Для збереження властивостей матриці переходів необхідно довизначити ймовірності переходів нульового рядка r, тобто присвоїти значення

$$d_{r1}^{(k+1)}, d_{r2}^{(k+1)}, \ldots d_{rm}^{(k+1)}, \qquad (13)$$

причому, необхідно щоб виконувались властивості матриці переходів ланцюга Маркова

$$\sum_{j=1}^{m} p_{rj}^{(k+1)} = 1, \; p_{rj}^{(k+1)} \le 1, \; i = \overline{1,m}. \qquad (14)$$

Було перевірено вплив довизначення нульового рядка на властивості періодичного ланцюга Маркова. Оскільки основною особливістю ланцюгів Маркова є те, що майбутній стан системи не залежить від її попередніх станів, то визначимо яким чином вплине введення значень ймовірностей (13) в оцінці матриці $\widetilde{\Pi}^{(k+1)}$ на те, в якому стані перебуватиме система на наступному кроці $k+2$.

Як відомо [7], матриця переходів за 2 кроки обчислюється наступним чином

$$\Pi^{(k,2)} = \Pi^{(k)} \times \Pi^{(k+1)} \qquad (15)$$

Після проведеня обрахунків виявлено, що в елементах матриці $\Pi^{(k,2)}$ довизначенні значення нульових рядків (13) відсутні, а отже вони не впливають на те, в якому стані перебуватиме система на кроці $k+2$. Аналогічним чином можна показати відсутність впливу довизначення нульових рядків на стан системи, якщо їхня кількість більше одного.

Отже при виборі значень послідовності (13) достатньо враховувати лише умови (14), тому приймемо, що для j-го стовпця матриці значення ймовірностей переходів кожного нульового рядка дорівнюватимуть середньому значенню ймовірностей переходів ненульових рядків даного стовпця, тобто

$$d_{rj}^{k+1} = \frac{\sum_{i=1}^{m-a} p_{ij}^{k+1}}{m-a}, \; j = \overline{1,m}, \qquad (16)$$

де a – кілікість довизначених рядків.

Важливо, що умови (14) в даному випадку виконано і властивості матриць переходів ланцюга Маркова збережено.

Результати оцінювання

Враховуючи описані вище зауваження проведено оцінювання матриць переходів реалізації енергонавантаження в робочі дні Тернопільського національного технічного університету в 2008 році, в результаті чого отримано 24 оцінки, які наведено у таблиці 1.

Табл. 1. Оцінки матриць переходів енергонавантаження

$$\widetilde{\Pi}^{(0)} = \begin{vmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(1)} = \begin{vmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(2)} = \begin{vmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{vmatrix}$$

$$\widetilde{\Pi}^{(3)} = \begin{vmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(4)} = \begin{vmatrix} 0{,}9786 & 0{,}0214 & 0 \\ 0{,}9786 & 0{,}0214 & 0 \\ 0{,}9786 & 0{,}0214 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(5)} = \begin{vmatrix} 0{,}1092 & 0{,}8908 & 0 \\ 0 & 1 & 0 \\ 0{,}0546 & 0{,}9454 & 0 \end{vmatrix}$$

$$\widetilde{\Pi}^{(6)} = \begin{vmatrix} 0{,}4800 & 0{,}5200 & 0 \\ 0{,}0335 & 0{,}9665 & 0 \\ 0{,}2568 & 0{,}9454 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(7)} = \begin{vmatrix} 0{,}2632 & 0{,}7368 & 0 \\ 0 & 1 & 0 \\ 0{,}1316 & 0{,}9454 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(8)} = \begin{vmatrix} 0{,}6000 & 0{,}4000 & 0 \\ 0 & 0{,}8384 & 0{,}1616 \\ 0{,}3000 & 0{,}6192 & 0{,}0808 \end{vmatrix}$$

$$\widetilde{\Pi}^{(9)} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 0{,}8144 & 0{,}1856 \\ 0 & 0{,}0270 & 0{,}9730 \end{vmatrix} \quad \widetilde{\Pi}^{(10)} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 0{,}8742 & 0{,}1258 \\ 0 & 0{,}0695 & 0{,}9306 \end{vmatrix} \quad \widetilde{\Pi}^{(11)} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 0{,}9514 & 0{,}0486 \\ 0 & 0{,}2069 & 0{,}7931 \end{vmatrix}$$

$$\widetilde{\Pi}^{(12)} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 0{,}9936 & 0{,}0064 \\ 0 & 0{,}3553 & 0{,}6447 \end{vmatrix} \quad \widetilde{\Pi}^{(13)} = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 0{,}9890 & 0{,}0110 \\ 0 & 0{,}2600 & 0{,}7400 \end{vmatrix} \quad \widetilde{\Pi}^{(14)} = \begin{vmatrix} 1 & 0 & 0 \\ 0{,}0052 & 0{,}9896 & 0{,}0052 \\ 0 & 0{,}4103 & 0{,}5897 \end{vmatrix}$$

$$\widetilde{\Pi}^{(15)} = \begin{vmatrix} 1 & 0 & 0 \\ 0{,}0097 & 0{,}9563 & 0{,}0340 \\ 0 & 0{,}5833 & 0{,}4167 \end{vmatrix} \quad \widetilde{\Pi}^{(16)} = \begin{vmatrix} 0{,}6667 & 0{,}3333 & 0 \\ 0{,}0190 & 0{,}9810 & 0 \\ 0 & 0{,}9412 & 0{,}0588 \end{vmatrix} \quad \widetilde{\Pi}^{(17)} = \begin{vmatrix} 0{,}8750 & 0{,}1250 & 0 \\ 0{,}0267 & 0{,}9733 & 0 \\ 0 & 1 & 0 \end{vmatrix}$$

$$\widetilde{\Pi}^{(18)} = \begin{vmatrix} 1 & 0 & 0 \\ 0{,}0996 & 0{,}9004 & 0 \\ 0{,}5498 & 0{,}4502 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(19)} = \begin{vmatrix} 1 & 0 & 0 \\ 0{,}2462 & 0{,}7538 & 0 \\ 0{,}6231 & 0{,}3769 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(20)} = \begin{vmatrix} 1 & 0 & 0 \\ 0{,}5467 & 0{,}4533 & 0 \\ 0{,}7733 & 0{,}2267 & 0 \end{vmatrix}$$

$$\widetilde{\Pi}^{(21)} = \begin{vmatrix} 0{,}9699 & 0{,}0301 & 0 \\ 0{,}7794 & 0{,}2206 & 0 \\ 0{,}8746 & 0{,}1254 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(22)} = \begin{vmatrix} 1 & 0 & 0 \\ 0{,}9500 & 0{,}0500 & 0 \\ 0{,}9750 & 0{,}0250 & 0 \end{vmatrix} \quad \widetilde{\Pi}^{(23)} = \begin{vmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{vmatrix}$$

Оцінки матриць переходів характеризують особливості поведінки споживання електроенергії в кожну годину доби. Аналіз оцінок матриць $\widetilde{\Pi}_0^{(22)}, \widetilde{\Pi}_0^{(23)}, \widetilde{\Pi}_0^{(0)}, \ldots \widetilde{\Pi}_0^{(4)}$, які відповідають нічним годинам доби, показує, що споживання електроенергії є низьким і носить стаціонарний характер. Матриці $\widetilde{\Pi}_0^{(5)}, \ldots \widetilde{\Pi}_0^{(8)}$ свідчать про зростання енергонавантаження, яке, як видно з матриць $\widetilde{\Pi}_0^{(10)}, \ldots \widetilde{\Pi}_0^{(15)}$, набуває високих значень вдень, причому найбільші ймовірності переходу в стан з високим споживанням електроенергії спостерігаються в матриці $\widetilde{\Pi}_0^{(13)}$. Після цього енергонавантаження поступово зменшується, про що свідчать матриці $\widetilde{\Pi}_0^{(16)}, \ldots \widetilde{\Pi}_0^{(21)}$. Описані особливості підтверджуються графіком на рисунку 2. Однак зауважимо, що оцінка математичного сподівання не показує можливі сплески на графіку енергонавантаження, проте можливість такої поведінки енергосистеми проглядається у оцінках ймовірностей переходів.

Висновки

Отримані результати підтверджують теоретичні положення про можливість використання для опису енергонавантажень періодичних ланцюгів Маркова. Аналіз знайдених в роботі оцінок матриць переходів показує, що вони відображають ймовірнісні особливості поведінки графіків споживання електроенергії і можуть бути використані в задачах прогнозу. Методи оцінювання перехідних матриць періодичного ланцюга Маркова можуть бути успішно застосовані для дослідження стохастичних періодичних сигналів марківського типу.

Перелік посилань

1. Приймак М.В. Основи теорії моделювання, аналізу і прогнозу в автоматизованих системах управління ритмічними процесами: Автореф. дис докт. техн. наук: 05.13.06 /Київ: НАУ, 2001. – 34 с.

2. Приймак М.В. Марківські періодичні процеси / М.В. Приймак // Вісник Тернопільського державного технічного університету. – 2003. – Т.8, число 3. – С. 17–21.
3. Приймак М.В. Періодичні ланцюги Маркова в задачах статистичного аналізу і прогнозу енергонавантажень / М.В. Приймак // Технічна електродинаміка. – 2004. – №2. – С. 3–7.
4. Приймак М. Елементи однорідності для періодичних ланцюгів Маркова / М. Приймак, С. Прошин // Вісник ТДТУ. — 2009. — Том 14. — № 2. — С. 114–123
5. Приймак М.В. Оцінювання матриць переходів періодичних ланцюгів Маркова Маркова / М.В. Приймак, С.Ю. Прошин // ISSN 1990-558 Електроніка та системи управління. 2009. – №3(21). – С. 26–33.
6. Приймак М. Похибка оцінок матриць переходів періодичного ланцюга Маркова / Приймак М., Віцентій О., Прошин С. // Вісник ТНТУ. — 2010. — Том 15. — № 3. — С. 150-159.
7. Ширяев А.Н. Вероятность. Учебное пособие для вузов / Ширяев А.Н. – М.: Наука, 1980. – 576 с.

УДК 519.854.2, 681.5.13

АРАКЕЛЯН Г.А.,
СЕРГІЄНКО А.М.

ВИЗНАЧЕННЯ ЗОЛЬНОСТІ ВУГІЛЛЯ ЗА ДОПОМОГОЮ МОДЕЛЬОВАНОГО ВІДПАЛЮВАННЯ

В статье описан способ определения зольности угля на основе косвенных измерений. Благодаря тому, что поиск интерполяцинной функции от спектра измеренного сигнала выполняется методом моделируемого отжига, получена точность измерений до 7%.

A method for the coal quality estimation is described which is based on indirect measurements. The estimation consists in the Fourier spectrum analysis of the acoustic transducer signal using unlinear interpolation formula. The formula coefficients are searched by the simulated annealing method. The resulting formula provides the 7% precision of the coal ashes measurement.

Вступ

У складних технічних системах багато параметрів доводиться визначати непрямими методами. При цьому часто неможливо визначити аналітичні залежності, які пов'язують параметри системи з їх зовнішніми проявами чи навіть побудувати приблизну математичну модель системи. У цьому випадку можуть мати успіх такі методи, як методи нейронної мережі та модельованого відпалювання. Але для успішного використання цих методів необхідно використовувати доволі довгі навчальні послідовності [1,2].

У цій статті описано досвід використання методу модельованого відпалювання для визначення зольності паливного вугілля за непрямими вимірами. Таке визначення необхідне, наприклад, для відбракування вагонів з неякісним вугіллям, що приходять на теплові електростанції. При цьому вміст золи у вугільлі, що визначається прямим стандартним методом, стає відомим лише через добу, коли неякісне вугілля вже поступило на склад електростанції. Оскільки кількість вимірів при цьому не перевищує кількості вагонів, то навчальна послідовність для непрямих методів, які передбачаються для використання, є суттєво обмеженою.

1. Спосіб визначення вмісту золи у вугіллі

Розроблений на факультеті інформатики та обчислювальної техніки НТУУ "КПІ" спосіб визначення вмісту золи у вугіллі призначений для визначення якості вугілля, що перевозиться у вагонах залізничного транспорту. Згідно зі способом, у вугілля, яке знаходиться у вагоні, занурюють акустичний датчик, а у об'ємі вагона збуджуються акустичні коливання. Сигнал з датчика записується та аналізується за допомогою ПЕОМ.

Дослідним шляхом встановлено, що існує стійкий зв'язок між спектральним складом сигналу з датчика та якістю вугілля, а саме – з процентним складом золи у ньому. Цей зв'язок визначається за допомогою функції:

$$x_i = \xi(\Theta_1,\Theta_2,\ldots,\Theta_N, S_1,S_2,\ldots,S_N) + \Delta_i, \qquad (1)$$

де x_i – фактичний вміст золи у вугіллі; ξ – інтерполяційна функція; $S_1,S_2,\ldots,S_N$ – значення амплітуди спектру сигналу в N частотних смугах; $\Theta_1,\Theta_2,\ldots,\Theta_{N'}$ – коефіцієнти функції ξ , $N'\geq N$; Δ_i – похибка вимірювання.

Даний спосіб вимірювання відноситься до непрямих способів [3]. Пошук інтерполяційної функції ξ є одним із найважливіших етапів розробки і впровадження способу. Складність цього пошуку полягає з одного боку, в трудомістському накопиченні вимірювальних даних $S_1,S_2,\ldots,S_N$, з іншого боку, у визначенні якнайбільш підходящої функції ξ, яка б давала мінімізоване значення величини Δ_i. Причому, ця функція не є сталою, тому що весь час нові виміри складу вугілля можуть сприяти до більш кращої її інтерполяції.

Аналіз виміряних даних і визначення вмісту золи у вугіллі за допомогою ПЕОМ виконує розроблена програма Spectr. Ця програма виконує ввід даних, обчислення швидкого перетворення Фур'є, відображення спектру сигналу, виділення смуг S_1, $S_2,\ldots$, S_N, обчислення функції (1) та відображення результату. На рис. 1. показано вікно цієї програми з відображеним спектром.

На етапі навчання вимірюють параметри $S_1,S_2,\ldots,S_N$ для проб вугілля, процентний склад яких відомий завдяки стандартним вимірюванням. На основі серії вимірювань обчислюють функцію ξ, яку потім вбудовуть в програму Spectr.

На етапі перевірки вимірюють процентний склад золи у вугіллі в пробах з наперед невідомим складом і порівнюють результати з ре-

зультатами стандартних вимірювань. За результатами перевірки обчислюють середню похибку вимірювання Δ і приймають рішення про прийняття обчисленої функції ξ.

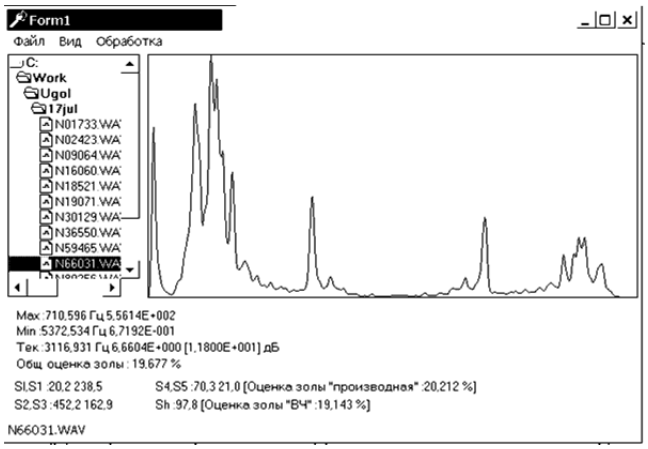


Рис.1. Вікно програми Spectr

2.Пошук інтерполяційної функції

З урахуванням похибок вимірювань зв'язок між результатами вимірювань і невідомими у (1) описується наступною фундаментальною системою рівнянь:

$$\begin{cases} \xi(\Theta_1,\Theta_2,\ldots,\Theta_{N'}, S_{1,1},S_{2,1},\ldots,S_{N,1}) + \Delta_1 = x_1, \\ \xi(\Theta_1,\Theta_2,\ldots,\Theta_{N'}, S_{1,2},S_{2,2},\ldots,S_{N,2}) + \Delta_2 = x_2, \\ \xi(\Theta_1,\Theta_2,\ldots,\Theta_{N'}, S_{1,n},S_{2,n},\ldots,S_{N,n}) + \Delta_n = x_n, \end{cases} \quad (2)$$

де $n > N$ – число вимірювань на етапі навчання системи.

При цьому за функцію ξ використовують наперед задану функцію, вид якої є підходящим як для опису фізичного явища, так і за складністю обчислень, наприклад, лінійну чи степеневу функцію. Неспівпадання істинної та прийнятої функціональної залежності ξ є джерелом похибок, які не залежать від якості вимірювальної апаратури і які вважають похибками моделі [3].

Під час пошуку функції (1) за мінімумом даних у системі рівнянь (2) залишають N' рівнянь, яких досить для однозначного визначення невідомих параметрів. Спочатку знаходять деяке початкове вирішення $\Theta^0_1,\Theta^0_2,\ldots,\Theta^0_{N'}$. Тоді припускають, що між малими змінами параметрів $\delta\Theta_i = \Theta_i - \Theta^0_i$ та невеликими змінами функції $\delta\xi = \xi - \xi(\Theta^0_1,\ldots,\Theta^0_{N'})$ є взаємозв'язок:

$$\delta\xi = \delta\Theta_1\partial\xi/\partial\Theta_1 + \delta\Theta_2\partial\xi/\partial\Theta_2 + \ldots + \delta\Theta_{N'}\partial\xi/\partial\Theta_{N'}.$$

Шукають поправки $\delta\Theta_1,\delta\Theta_2,\ldots,\delta\Theta_{N'}$ до початкового наближення, припускаючи, що існує лінійний зв'язок між приростами виміряних величин $\delta x = x - \xi(\Theta^0_1,\Theta^0_2,\ldots,\Theta^0_{N'})$ і поправками $\delta\Theta_i$.

Слід відзначити, що обробка вимірів за мінімумом даних пов'язана зі специфічними фізичними особливостями задачі, які в даному випадку є невідомими. З іншого боку, обробка надлишкових вимірів при наявності вдалого початкового приближення $\Theta^0_1,\Theta^0_2,\ldots,\Theta^0_{N'}$ є універсальною і мало залежить від фізичної суті задачі [3].

Знайдена апроксимація функції ξ за мінімумом даних для оцінки якості вугілля дає похибку Δ > 20%. Отже, обробка вимірів за цим методом є неприйнятною. Позитивним результатом цього методу є лише те, що знайдені початкові наближення $\Theta^0_1,\Theta^0_2,\ldots,\Theta^0_{N'}$, використовуються у подальшій оптимізації.

Оцінками, що одержані за методом найменших квадратів, називаються такі набори, $\Theta_1,\Theta_2,\ldots,\Theta_{N'}$, що призводять до мінімуму зважену суму:

$$\Phi(\Theta_1,\Theta_2,\ldots,\Theta_{N'}) = \Sigma_{i=1}^{n} p_i(x_i - \xi(\Theta_1,\Theta_2,\ldots,\Theta_{N'}, S_{1,i},S_{2,i},\ldots,S_{N,i}))^2; \quad (3)$$

де p_i – коефіцієнт важливості i-го виміру. Цей коефіцієнт приймає максимальне значення для найбільш достовірних вимірів.

Недоліком методу найменших квадратів є те, що він чутливий до непопадань, тобто вимірів $S_{1,i},S_{2,i},\ldots,S_{N,i}$ або x_i, що дають надмірно велике значення похибки Δ. А таке явище є доволі ймовірним при вимірюванні складу вугілля.

У випадку пошуку інтерполяційної функції вмісту золи в вугіллі вигляд функції ξ є наперед невідомим. Φ, ймовірно, є нелінійною функцією, яка має декілька екстремумів. Це пов'язане з тим, що проби вугілля з однаковим вмістом золи можуть мати різний спектр $S_{1,i},S_{2,i},\ldots,S_{N,i}$. Крім того, в процесі оптимізації функції вона може змінювати вигляд, що тягне за собою зміну системи вирішуваних рівнянь. Тому метод найменших квадратів для данної задачі є неприйнятним.

Існуючі методи випадкового пошуку розроблені для знаходження глобального екстремума функції Φ. Більшість з них умовно розділяють на два класи: клас "незалежних" методів та клас "блукаючих" методів пошуку [4]. До першого класу відносяться методи в яких спроби встановлення величини вектору $(\Theta_1,\Theta_2,\ldots,\Theta_{N'})$ виконуються у відповідності з деякою щільністю розподілення ймовірності його складових та

припущення, що вони не пов'язані між собою. Насправді, між виміряними частотними значеннями Θ_i є велика кореляція, що призводить до відмови від "незалежних" методів.

Другий клас складається з методів, що імітують блукання у просторі параметрів $(\Theta_1,\Theta_2,\ldots,\Theta_{N'})$ з метою пошуку екстремуму Ф. Більшість з цих методів використовують похідні $\partial\Phi/\partial\Theta_i$ функції якості для прискорення пошуку. При цьому імітується процес дифузії молекули, координати якої дорівнюють $(\Theta_1,\Theta_2,\ldots,\Theta_{N'})$. Оскільки пошук цих похідних представляє додаткову складну роботу, то такі методи для даного випадку є неприйнятними.

В роботі [1] було запропоновано комбінований метод, що вбирає в себе якості вищеназваних класів методів. Згідно з ним, моделюється процес броуновського руху молекули з координатами $(\Theta_1,\Theta_2,\ldots,\Theta_{N'})$ у фізичному тілі, що було нагріте до температури початку відпалювання і з часом остигає. Тому він був названий методом модельованого відпалювання (simulated annealing). Дія цього метода не залежить від фізичної природи об'єкта, що оптимізується, дає ефективні рішення і тому він використовується у даній роботі.

В роботі [1] показано, як моделювання остигання речовини перекласти на задачі оптимізації технічних об'єктів. При цьому прийняті наступні аналогії: стан речовини – вектор рішення $\boldsymbol{S} = (\Theta_1,\Theta_2,\ldots,\Theta_{N'})$; енергія – функція оптимальності Ф; зміна стану – перехід до наступного рішення ***Sn***; температура – окремий параметр, що змінюється в процесі оптимізації; кінцева температура t_{min} – точка останову оптимізації.

Узагальнений алгоритм модельованого відпалювання виглядає наступним чином:

Вибрати початкові рішення **Sn**, температуру t і коефіцієнт остигання a;
 repeat
 за випадковим законом вибрати рішення **S**, яке є сусіднім до **Sn**;
 знайти приріст енергії ΔФ=Ф(**S**)–Ф(**Sn**);
 if ΔФ<0 **then**
 Sn =**S**;
 else
 генерація випадкового числа p
 if p<exp(ΔФ/t) **then**
 Sn =**S**;
 end if;
 end if;
 t=t*a;
 until $t<t_{min}$;

Цей алгоритм був реалізований в програмі UGOL_KRIT, що знаходить оптимізоване значення $\boldsymbol{S} = (\Theta_1,\Theta_2,\ldots,\Theta_{N'})$.

Програма пошуку коефіцієнтів інтерполяційної функції UGOL_KRIT написана мовою VHDL, яка є зручною для експериментального моделювання методу відпалювання. В програмі функція ξ обчислюється за формулою:

$$\xi = 1 - \exp\left(- \frac{\Theta_{N'-1}}{\left| \sum_{i=1}^{N} \Theta_i S_i \div \sqrt{\sum_{i=1}^{N} S_i^2} + \Theta_{N'} \right|} \right),$$

$N' = N+2$.

3. Методика пошуку коефіцієнтів інтерполяційної функції

Була розроблена наступна методика пошуку коефіцієнтів інтерполяційної функції. За допомогою вимірювального стенда, що реалізує спосіб визначення складу золи у вугіллі, виконують серію вимірювань характеристик вугілля. При цьому беруть проби вугілля, процентний склад яких відомий завдяки еталонним вимірюванням. Можливе також визначення складу вугілля за документами поставника вугілля, або навіть за суб'єктивними оцінками експертів. Виміряні дані з датчика реєструють в ПЕОМ. Одночасно реєструють процентний склад вугілля і ступінь ймовірності (коефіцієнт важливості) виміру.

За допомогою програми SPECTR виконують спектральний аналіз виміряних даних. Таблицю параметрів $S_1,S_2,\ldots,S_N$ з процентним складом вугілля і коефіцієнтом важливості заносять у програму пошуку коефіцієнтів інтерполяційної функції UGOL_KRIT. При цьому коефіцієнт важливості виміру задають за Табл.1.

Таблиця 1 Коефіцієнт важливості

Джерело інформації про склад вугілля	*p*
стандартне вимірювання для одного вагона	8
стандартне вимірювання для партії вагонів	3
склад за сертифікатом вугілля, уточнений експертом	2
склад за сертифікатом вугілля	1,5
експертна оцінка	1
склад за сертифікатом, але дані неправдоподібні	0,1

Ця програма знаходить оптимізоване значення коефіцієнтів $\boldsymbol{S}=(\Theta_1,\Theta_2,\ldots,\Theta_{N'})$ для інтер-

поляційної функції (1). При цьому спочатку за допомогою програми знаходять початкове наближення коефіцієнтів $\Theta^0_1,\Theta^0_2,\ldots,\Theta^0_{N'}$ методом інтерполяції за мінімумом даних. Потім ці коефіцієнти заносять в ту саму програму і знаходять остаточне вирішення задачі інтерполяції.

На етапі перевірки вимірюють процентний склад золи у вугіллі в пробах з наперед невідомим складом і порівнюють результати з результатами стандартних вимірювань. За результатами перевірки обчислюють середню похибку вимірювання Δ і приймають рішення про прийняття обчислених функцій ξ_1 і ξ_2.

З появою нових серій достовірних вимірів складу вугілля цими вимірами доповнюють таблицю в програмі пошуку коефіцієнтів і розраховують коефіцієнти для більш точної інтерполяційної функції ξ і знову виконують етап перевірки.

4. Експериментальні результати

Експериментальна установка, що вимірює вміст золи у вугіллі, була випробувана у червні 2003 р. на Трипільській ДРЕС. Було зібрано більше 40 вимірів, що складаються з $S_1,S_2,\ldots,S_N$, x, і p . Виміри були вставлені в програму UGOL_KRIT. У результаті першого запуску програми було одержане початкове наближення

$$\boldsymbol{S} = (-19.0,2.0,6.0,3.0,8.0,-24.0,10.0).$$

Після інших запусків експериментальним шляхом були вибрані такі початкова температура та коефіцієнт застигання, щоб процес зменшення енергії дійшов до мінімального значення. Одержані залежності температури і енергії від номера ітерації алгоритму показані на Рис 2.

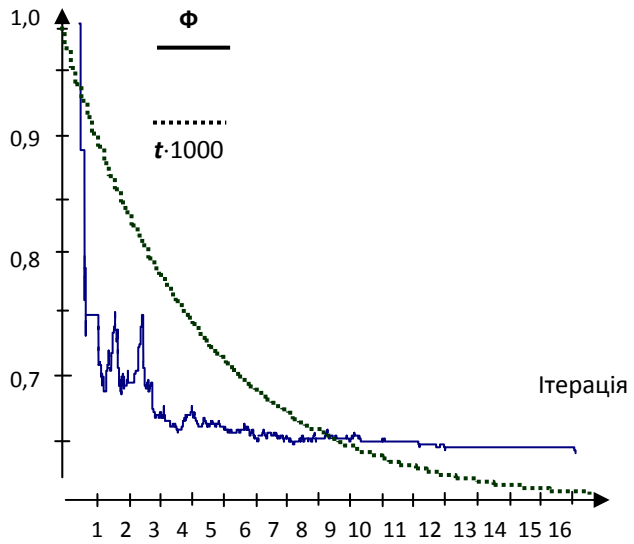


Рис. 2. Залежності температури і енергії від номера ітерації алгоритму

Після закінчення остигання було одержано наступний результат:

$$\boldsymbol{S} = (-20.47,11.22,7.13,16.59, 2.71, -27.30,45.08).$$

Набір коефіцієнтів був вбудований у програму SPECTR, після чого були виконані контрольні виміри. Ці виміри показали середню абсолютну похибку, яка дорівнює 7% і яка задовольняє вимоги користувача.

Висновки

Розроблений спосіб визначення вмісту золи у вугіллі відноситься до непрямих способів вимірювання, для якого пошук інтерполяційної функції ξ є одним із найважливіших етапів розробки і впровадження способу.

Встановлено, що при пошуку інтерполяційної функції вмісту золи у вугіллі слід використовувати методи випадкового пошуку, а саме – метод модельованого відпалювання.

Розроблена методика і програма пошуку коефіцієнтів інтерполяційної функції ξ на основі метода модельованого відпалювання, які необхідні для перевірки і впровадження способу визначення вмісту золи у вугіллі.

Експериментальною перевіркою було підтверджено дієвість і ефективність розроблених методики і програми пошуку коефіцієнтів інтерполяційної функції навіть для коротких навчальних виборок. Завдяки цьому, була одержана точність виміру складу золи у вугіллі 7%. Ця точність може бути підвищена при подальшому використанні методики після збору додаткових вимірів з достовірними значеннями.

Перелік посилань

1. Kirkpatrick S., Gellatt C.D., Vecchi Jr.M.P. Optimization by Simulated Annealing. –Science. –V.220. –1983. –N5. –p.671-680.
2. Хайкин С. Нейронные сети. Полный курс. – М.: Вильямс, 2006. – 1104с.
3. Мудров В.И., Кушко В.Л. Методы обработки измерений. –М.: Сов.Радио, –1976. – 192с.
4. Растригин Л.А. Статистические методы поиска. –М.: Наука. –1968. – 376с.

УДК 19.711+519.6+681.3.01

МЕЛЬНИК Т.П.

УДОСКОНАЛЕННЯ ТЕХНОЛОГІЇ ДИНАМІЧНОГО МОДЕЛЮВАННЯ СТОХАСТИЧНОЇ СИСТЕМИ ВИЗНАЧЕННЯ ЗОН ЗАТОПЛЕННЯ ІЗ ВРАХУВАННЯМ ДАМБ ОБВАЛУВАННЯ

Здійснено моделювання із застосуванням аналізу змін характеристик впливу на оптимальне розміщення дамб обвалування. Удосконалено технологію динамічного моделювання за рахунок використання випадкових складових формування паводкового стоку. Об'єктом розгляду є метод побудови моделей поведінки стохастичних систем.

Done using simulation analysis of changes in characteristics influence the optimal placement of dams diking. Enhanced dynamic simulation through the use of random elements forming flood runoff. Under review is a method for constructing behavior models of stochastic systems.

1. Вступ

Як правило, виділяють такі основні задачі побудови динаміко-стохастичних моделей стоку, як облік в динамічних не випадкових моделях процесів стохастичної просторової змінності гідрометеорологічної дії на водозбір і його характеристик, використання в якості входу детермінованих моделей статистичного опису гідрометеорологічної дії на водозборі та облік помилок вихідної інформації в детермінованих моделях. Однак, результатом таких моделей можуть бути усереднені найбільш ймовірні характеристики водного режиму або їх статистичний розподіл. Вигляд статистичних зв'язків і структура динаміко-стохастичної моделі повинні мінятися в залежності від еко-географічних умов, тому їх абстракція повинна проводитися в різних гідроекологічних системах.

В цілому, із-за труднощів відміни між обсягом теоретичних знань про формування стоку та емпіричними даними застосування методів динаміко-стохастичного моделювання не здійснюється. Однак, використання гносеологічних моделей, що будуються на пізнанні тільки фізичних закономірностей не враховують випадковість факторів впливу, що є джерелом стихійних явищ затоплення територій. Тому мною запропонована динамічна модель із застосуванням аналізу змін характеристик впливу на оптимальне розміщення дамб обвалування саме із врахуванням випадкових складових.

2. Постановка задачі

В задачі досліджень входило:

- здійснити аналіз факторів впливу формування ходу стоку під час дощових паводків на водозборах гідроекологічних систем і річок, у заплавах яких вони споруджені;
- опрацювати досвід застосування математичних моделей дощового та снігодощового стоку для прогнозування паводків, що надасть можливість наукового обґрунтування розташування дамб відносно русла річки;
- удосконалити технологію побудови динамічних моделей стохастичних систем;
- удосконалену технологію проілюструвати прикладом.

3. Виклад основного матеріалу дослідження

Оскільки теорія ймовірності та статистичні методи часто слугують єдиним шляхом кількісного оцінювання характеристик метеорологічних і гідрологічних явищ вони застосовуються в гідрометеорологічних дослідженнях при вирішенні багатьох завдань. Зокрема, гідрометеорологічні процеси за своєю сутністю багатофакторні, тому математичний опис їх результатів можливий лише статистичними методами. Однак, достовірність визначення характерних рівнів води в річках шляхом опрацювання даних їхніх багаторічних спостережень методами математичної статистики не завжди може бути забезпечена через нестабільність русла річки не лише у створі поста, але й на річковій ділянці побли

зу.

Класифікацією та алгоритмізацією задач даного характеру займався Гнатієнко Г.М. Дослідження Гермейєра і Байеса-Лапласа описують критерії експертного оцінювання, що використовуються у випадку ідеалізованих практичних

ситуацій з метою вивчення внутрішніх зв'язків оцінювання даних.

Теоретично дана задача для прогнозування досліджена досить обширно, як у неперервному так і у дискретному варіантах. Відповідними алгоритмами екстраполяції є форми фільтрів Вінера, Калмана-Б'юсі і різницевих рівнянь авторегресії. Серед екстраполяційних моделей важливе місце займають трендові моделі розвитку. Метою створення таких моделей є прогнозування показників, що формуються під впливом великої кількості чинників, інформація з яких частково може бути відсутньою. Емпіричні ряди динаміки несуть на собі вплив не тільки основних, але і другорядних факторів, що згладжують головну тенденцію у зміні досліджуваного показника. Хід зміни шуканого пов'язують не тільки з чинниками, а й з часом.

Дана методологія є недостатньо опрацьованою у вирішенні проблеми затоплення території Закарпаття з точки зору прогнозування випадкової складової моделювання процесів формування стоку в екстремальних ситуаціях паводку. Не завжди дослідження базуються на розрахунках засобами програмного забезпечення ЕОМ [1, 2], що є необхідним для більш точного і термінового вирішення ситуації. Варто, також, зупинитися на зручності та ефективності інтерполяції, що дає змогу визначити граничні умови, описати залежності з розривами функцій та їх похідні, правильно організувати введення та виведення даних при математичному моделюванні, виконати аналіз одержаних результатів, порівняти результати з експериментальними даними. Зокрема, вартою уваги, і ділянкою, що потребує розробки заходів для покращення пропуску критичних рівнів води під час паводків басейну р. Боржави, є місцевість території від вузькоколійного залізничного мосту біля с. Шаланки до автодорожнього моста на ділянці автодороги Заріччя-Вільхівка. В подальшому наші дослідження спиратимуться на дану територію.

3.1. Врахування взаємозв'язку та впливу метеорологічних факторів на формування стоку

Нехай, у результаті вишукувань отримали набір даних

$$x_1^{(n)}, x_2^{(n)}, ..., x_i^{(n)}, y_1^{(n)}, y_2^{(n)}, ..., y_j^{(n)}, n = 1,2,..., N.$$

За якими необхідно підрахувати значення $q_1, q_2, ..., q_k$, де x_i і y_j пов'язані із шуканими $q_1, q_2, ..., q_k$ залежністю

$$y_m^n = \varphi_m(q_1, q_2, ..., q_k; x_1^{(n)}, x_2^{(n)}, ..., x_i^{(n)}).$$

Результат кожного окремого процесу є випадковою величиною з деякою густиною ймовірності $f(y_1, y_2, ..., y_m)$.

Для виконання гідрологічних розрахунків при проектуванні та експлуатації гідромеліоративних систем необхідно виявити зокрема залежність між опадами і стоком на даній території, використовуючи інші фактори як параметри.

Згідно рекомендацій Хойта, цикл стоку ділиться на наступні фази:

- період перед початком стокоформуючих опадів;
- період перед закінченням зливи або зливової частини тривалого дощу;
- період після закінчення зливового дощу, але до вичерпування запасів поверхневого затримання і запасів, накопичених в руслах.

Обчислення зволоження водозбору до початку паводку, який виражається через дефіцит вологості, виконується безперервно, починаючи з кінця попереднього паводку, або ж з весняного періоду, коли вологість ґрунту може бути прийнята близькою до максимально можливої. Американські гідрологи широко використовують характеристику початкового зволоження індексу затримання води [3]:

$$I_d = [\, P - Q \,] * K^t ,$$

де P – опади, мм;

Q – витрата води, м^3/с;

K – параметр;

t – розрахунковий інтервал, год.

Ними було показано, що цей показник чисельно близький до кількості вологи у верхньому тридцяти сантиметровому шарі ґрунту.

У випадку безперервності випадання опадів:

$$I_a(t) = \int_0^t \left[\overline{P(\tau) - Q(\tau)}\right] * K^{t-\tau} \partial t$$

В.П. Петрук, аналізуючи формули, пише, що в даному випадку зробив неявне припущення, що за кожен день випаровується *(1-K)* – та частина вологи. Таким чином, запас води на водозборі можна буде розрахувати більш точніше, якщо підрахувати хід випаровування та інфільтрації.

Припустимо, що водозбір володіє деякою постійно максимальною водоутримуючою ємкістю, яка повністю заповнена, коли опади витрачаються лише на випаровування і стік, тобто коли можна знехтувати інфільтрацією. Такі періоди спостерігаються після сходу снігу, чи

після сильного дощу. Зміни запасів води в цій ємкості (R_b) за одиницю часу визначаються співвідношенням опадів, випаровування і стоку [3]:

$$R_b(t) = \int_{t_0}^{t} \left[\overline{P(\tau) - E(\tau) - Q(\tau)}\right] * K(t-\tau)\ \partial\tau$$

де $K(t - \tau)$ – функція впливу, яка відображає інерційність зміни запасів води в басейні. Якщо $R_b(t) > 0$ то йде процес заповнення водоутримуючої ємкості, при $R_b(t)<0$ водоутримуюча ємкість звільняється від води. Ту частину водоутримуючої ємкості, яка звільняється від вологи перед випаданням розрахункового дощу можна обчислити таким чином. Спочатку знаходиться позитивна хвиля $R_b(t) > 0$, а потім обчислюється інтеграл:

$$W_0 = -\int_{t_0}^{T} R_d(t)\partial t$$

де T – початок розрахункового дощу,

t_0 – момент часу, який при $R_b(t)$ стає рівним нулю після максимальної позитивної хвилі.

Тривалість паводку – це інтервал часу від початку хвилі зростання паводку до досягнення припущеного нульового значення складової паводкового стоку. Тривалість паводку можна виразити у вигляді:

$T_n = t_d + t_{cm}$,

де t_d – тривалість дощу,

t_{cm} – час стікання води до замикаючого створу.

Ця величина з часом, необхідним для переміщення об'єму води з найбільш віддаленої точки водозбору до замикаючого створу. Для русел величина t_{cm} – час стікання води замикаючого створу, обчислюється за формулами Шезі.

Під час другої фази стоку, в початковий період дощу, додатковим до стоку є лише та частина опадів, яка випадає на поверхню річки і каналів. Частина опадів перехоплюється рослинами (інтерполяція) і поступово повертається в атмосферу у вигляді випаровування. Більша частина дощу, яка досягає поверхні землі, частково проходить крізь ґрунт у вигляді фільтрації, заповнюючи ґрунтову водоутримуючу ємкість, частково затримується у вигляді запасів у депресіях. В цей час схиловий стік відбувається лише з невеликої частини басейну, а також із крутих схилів.

Значення перехоплення рослинами дощу в процесі формування високих паводків порівняно невелике. Проте на територіях заселених, або покритих іншими видами густої рослинності існує перехоплення дощу різними видами рослин, яке визначається за формулою:

$$V_t = S_i + C_p * E_a * t_d\ ,$$

де S_i – шар затримання на одиницю проекції площі,

C_p – відношення поверхні рослинного покриву до площі її горизонтальної проекції,

E_a – інтенсивність випаровування на одиницю площі.

Однак через нестачу даних про перехоплення, в модель призначену для обчислення динаміки стокоутворення і прогнозування паводків на водозборах меліоративних систем ми наклали емпіричні залежності для визначення величини перехоплення в різних умовах:

$$I_{пер}(t) = \begin{cases} a + b\overline{P}(t)^n, I_{поч} = 0, q_1 = 0; \\ 0, I_{поч} > 0, q_1 > 0, \end{cases}$$

де a, b, n – емпіричні параметри, значення яких наведені в таблиці,

h – висота рослин в метрах.

Середнє значення перехоплення по водозбору визначається як сума добутків обчислених величин перехоплення на відповідні коефіцієнти поритості водозбору кожним видом рослинності. Проте після повного змочування рослинного покриву подальша інтерцепція дорівнювала б нулю, якби не було значного випаровування з великої площі поверхні листя рослин. Звідси витікає, що після завершення змочування листя кількість води, яка досягає поверхні ґрунту, дорівнює кількості опадів мінус випаровування з поверхні рослинного покриву. Тому в моделі розрахунок величини інтерцепції ведеться лише на початкових стадіях дощу до моменту коли $I_{поч}$ (початкове значення інтенсивності інфільтрації) або q_1 (інтенсивність поверхневого стоку) стане >0.

Після тривалого випадання дощу, об'єм перехоплення рослинами заповнюється, решта води стікає на землю. Заповнюються всі поверхневі депресії і ємкості перехоплення, дефіцит вологості ґрунту погашається на значну глибину, а інтенсивність інфільтрації наближається до мінімуму. Настає третя фаза циклу стоку. Одночасно на непроникних і частково проникних поверхнях водозбору утворюється тонкий шар води – плівка поверхневого затримання. Накопичуючись, вода збирається в невеликі струмки, які дають початок силовому стоку в малі водотоки, зливаючись потім в більш крупних потоках річок. Русловий стік набуває зв'язку з інтенсивністю опадів, за виключен-

ням частини вологи, яка йде на випаровування. Ґрунтовий стік збільшує витрати води в річці, а запаси ґрунтових вод знову поповнюються.

Інтенсивність поверхневого стоку розраховується в моделі згідно рекомендацій Попова після внесення деяких змін:

Після закінчення опадів припиняється схиловий стік і витрата в річці формується за рахунок запасів води накопичених в товщі водозбору (ґрунтових вод) та русловій мережі, це наступає четверта фаза стоку. Збільшується інтенсивність випаровування, починається транспірація рослинами, вода депресій продовжує проникати в ґрунт, а гравітаційна вода в межах зони аерації продовжує поповнювати запаси ґрунтових вод. Величина ґрунтового стоку, головним чином, залежить від коефіцієнта фільтрації та дренованості території, а стік по всій товщі водоносного шару досягається лише після заповнення водоутримуючої ємкості за рахунок дії складової сили тяжіння, направленої вздовж водоупору.

Інтенсивність ґрунтового стоку в моделі розраховується залежністю [3]:

$$q_2(t)=\begin{cases}\dfrac{(W_m-0.5[d(t)-d(t-\nabla t)]}{k_3}, q_2(t)\le \overline{P}(t)-E(t)-h(t)-i_0;\\ \overline{P}(t)-E(t)-h(t)+i_0, q_2(t)<\overline{P}(t)-E(t)-h(t)-i_0.\end{cases}$$

Для використання методики розрахунку і підвищення точності розрахунків з метою прогнозування об'єму та ходу стоку паводків на конкретних водозборах, варто розділити територію на певну кількість стокоформуючих часткових площ. Модель надає можливість визначити числові значення параметрів та врахувати закономірності переміщання паводкових хвиль і їх інтерференцію по кожній конкретній частковій площі водозбору.

3.2. Моделювання ситуації паводку із урахуванням дамб обвалування

Іще в 1959 р. В.Д. Комаров в області гідравлічних розрахунків запропонував використовувати для дослідження весняного стоку криві розподілу висоти стоку. Облік ефектів усереднення по площі за допомогою теоретичних кривих розподілу параметрів водозбору і вхідних величин у концептуальній моделі дощового стоку здійснювався В.І. Корнем і Л.С. Кучментом.

Пропозиції по обліку просторових розподілів факторів, які визначають талий стік у моделях із зосередженими параметрами зробив В.А. Румянцев. Е.М. Гусев дослідив вплив на формування дощового стоку на схилі статистичної змінності і автокореляції коефіцієнта фільтрації. Для апроксимації статистичного розподілу в точці використовувався двопараметричний гамма розподіл. Сміт і Хебберт, вивчаючи вплив просторової зміни гідрографічних характеристик на стік, припустили, що коефіцієнт фільтрації розподілений по логарифмічно-нормальному законі.

Із-за труднощів відміни між обсягом теоретичних знань про формування стоку та емпіричними даними застосування таких методів було призупинено. Однак, використання гносеологічних моделей, що будуються на пізнанні тільки фізичних закономірностей не враховують випадковість факторів впливу, що є джерелом стихійних явищ затоплення територій. Тому я зупинилася на моделюванні процесу формування стоку саме із врахуванням стохастичних складових.

Нехай, стохастичними складовими характеристик водозбору A_{ij} , де i і j – номери вузлів просторової мережі, розподілені по нормальному закону із середнім μ_A і середньоквадратичним відхиленням ν_A і утворюють однорідні і ізотропні поля з автокореляційною функцією.

Нехай, у результаті експерименту отримали набір даних

$$x_1^{(n)}, x_2^{(n)},..., x_i^{(n)}, y_1^{(n)}, y_2^{(n)},..., y_j^{(n)}, n=1,2,..., N.$$

За якими необхідно підрахувати значення $q_1, q_2,..., q_k$, x_i і y_j пов'язані із шуканими $q_1, q_2,..., q_k$ залежністю

$$y_m^n=\varphi_m(q_1, q_2,..., q_k; x_1^{(n)}, x_2^{(n)},..., x_i^{(n)}).$$

Результат кожного окремого процесу є випадковою величиною з деякою густиною ймовірності $f(y_1, y_2,..., y_m)$.

Проблема зводиться до розв'язку задачі оптимізації з використанням методів екстремального оцінювання, яка передбачає відшукування

$$y_i=f_i(x)\to \max$$

Розв'язок такої задачі слідує із завдання деякого правила вибору на множині ефективних об'єктів єдиного розв'язку, який вирішується формально і базується на застосуванні евристик і умов раціональності.

Для опису використовуватимемо символічне представлення у вигляді кортежу (*A, S, R, E, C, P*), де

A – множина об'єктів, *S* – множина обмежень, *R* – множина принципів оптимальності, *E* – множина формальних характеристик, *C* – множина цілей, які стоять перед дослідником, *P* – система переваг.

Для кожного параметра можуть бути відомі вагові коефіцієнти їх відносної важливості $\beta_1, \beta_2, ..., \beta_m$, важливо задати напрями оптимізації параметрів. Для кількісних (абсолютних, відносних чи нормованих) використовується верхня, середня (найімовірніша, найприйнятніша чи бажана) та нижня оцінки. Для підрахунку точкових значень параметрів здійснюється обчислення одним із способів.

Множину S – обмежень задамо

$s_{*l} \le s_l(a^1, a^2, ..., a^m) \le s_l^*,\ l \in \{l, ..., l_0\}$,

де s_l – довільні дійснозначні функції дискретного аргументу, $\grave{a}^{з}, {}^{з} \in J$ – параметри об'єктів, s_{*l}, s_l^* – дійсні числа, l_0 – кількість обмежень.

Для розв'язання проблеми матимемо стохастичні складові характеристик водозбору a_{ij}, де *i* і *j* – номери вузлів просторової мережі, розподілені по нормальному закону із середнім μ_A і середньоквадратичним відхиленням ν_A і утворюють однорідні і ізотропні поля з автокореляційною функцією [3]:

$$f_A(l) = \exp(-\alpha_A |l|),$$

де l – відстань між двома точками поля; α_A – радіус кореляції.

Величина задаватиметься $\grave{a}_{ij} = \mu_A + e_{ij}\nu_A$,

де e_{ij} - випадковий процес із середнім, рівним нулю, середньоквадратичним відхиленням, рівним одиниці, і радіусом кореляції α_A.

Величина e_{ij} моделюється методом Монте-Карло по формулах

$$e_{ij} = (2/n)^{1/2} \sum_{m=1}^{N} \cos(w_m(x_i \sin\gamma_m + y_j \cos\gamma_m + \varphi_m),$$

$$w_m = \alpha_A((\frac{1}{1 - G(w_m)})^2 - 1)^{\frac{1}{2}},$$

де *N* = 50; $\gamma_ь, \varphi_ь$ – випадкові величини, рівномірно розподілені на відрізку [0, 2π];

G(w$_m$) – спеціальна випадкова функція з діапазоном зміни 0-1.

Відмітки поверхні водозбору z_{ij} задаватимуться за допомогою двовимірної просторової мережі у вигляді [106]

$$z_{ij} = I * y_n(1 - (\frac{y_j - y_n}{y_n})^2)^{1/2} + e_{ij}\nu_z,$$

де *i* і *j* – номери вузлів мережі вздовж осей *x* і *y*; *I* – середній нахил між *y* = 0 і $y = y_n$; y_n – ширина водозбору продовж вісі *y* (величина y_n приймається сталою для усіх *x*); ν_z - середньоквадратична зміна стохастичної складової величини z_{ij}.

Рух води по поверхні водозбору визначався шляхом зміщення поступання на час добігання. Величину часу добігання віднайдемо по формулі:

$$t_{ij} = g_T y_j + h_T x_i + e_{ij}\nu_T,$$

де g_T, h_T – параметри, які визначаються по заданих швидкостях нахилу і руслового стікання;

ν_T – середньоквадратична зміна стохастичної частини.

Розрахунок інфільтрації води у ґрунт здійснимо по формулі

$$c_{ij} = 0{,}3(2k_{ij}b_{ij}(n_{ij} - \theta_{ij}^0))^{1/2},$$

отриманої із рівняння дифузії вологи при деяких додаткових припущеннях. Тут k_{ij} - коефіцієнт фільтрації; n_{ij} - пористість;

$$b_{ij} = \frac{1}{k_{ij}} \int_{\theta^0}^{n_{ij}} d_{ij}(\theta) d\theta,$$

де θ^0 – початкова волога; $d_{ij}(\theta)$ – коефіцієнт дифузії.

Поля інтенсивності опадів моделюємо [Брас і Родрігес-Ітурбе, 1979] враховуючи просторову кореляцію опадів і швидкість ливню. Для суми опадів за дощ H_r. Нехай, величини T_r, H_r, а також продовжуваність не дощових періодів τ_p – розподілені по експоненціальному закону.

Швидкість ливню моделюємо по нормальному розподілі $T(\mu_u, \nu_u)$, де μ_u – середнє значення швидкості u_r, а σ_u – її середньоквадратичне відхилення.

По величинах τ_p, c_{ij} підраховуємо зміни середнього рівня ґрунтових вод $z_{ij}^{'}$. Потужність зони аерації визначалася як

$$z_{ij}^{''} = z_{ij} - z_{ij}^{'}.$$

Початковий розподіл вологи задамо залежністю

$$\theta_{ij}^0 = n_{ij}(\exp(-d_1 Z_{ij}^{''})\exp(-d_2(\tau_p - \mu_\tau)) + e_{ij}\sigma_\theta,$$

де d_1, d_2 – коефіцієнти;

σ_θ – середньоквадратичне відхилення стохастичної складової вологи ґрунту.

Початковий дефіцит вологи визначався як

$$s_{ij}^0 = \Delta x \Delta y z_{ij}^{''}(n_{ij} - \theta_{ij}^0).$$

Створення нормального випадкового процесу здійснила генерацією звичайним способом вектора незалежних випадкових чисел і побудувала інтерполяційну залежність у проміжках між ними.

Витрати визначимо [3]:

$$Q^m = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} r_{ij}^{m-m'},$$

де $m' = t_{ij} / \Delta t$ (Δt - крок часу).

На меліоративних системах у період підйому паводку зменшуються сумарні експлуатаційні витрати, однак зменшується акумулююча ємкість і збільшується ймовірність втрат урожаїв від перезволоження та затоплення сільськогосподарських угідь. Оптимальне рішення наведеної задачі може бути знайдено при співставленні різних варіантів віддалі між дамбами з урахуванням їх взаємозв'язку з динамікою паводку, продуктивністю культур і витратами на експлуатацію.

4. Висновки

Виконано аналіз значних за обсягом натурних даних стосовно процесів, умов та особливостей формування паводків і весняного стоку води на водозборах досліджуваного регіону із урахуванням розташування дамб відносно русла річки. Вперше запропоновані для реалізації методичні рішення удосконалення методу динамічного моделювання щодо процесу поведінки стохастичних систем. Результати досліджень частково впроваджені при розробці лівобережної дамби захисту території с. В. Ком'яти басейну р. Боржави.

Наукові засади та методичні розрахунки можуть бути використані проектними інститутами. Узагальнення отриманих результатів дозволить удосконалити розрахунок визначення зон можливих затоплень із використанням існуючих математичних моделей, розробити елементи комплексного програмного забезпечення автоматизації інформаційно-вимірювальної системи із використанням можливості оптимізації режиму роботи гідроекологічної системи на основі наукового обґрунтування розташування дамб відносно русла річки.

Список літератури

1. Сапсай Г.І., Чіпак В.П., Мельник Т.П. Елементи автоматизації управління водними ресурсами в басейні ріки Тиса. // Вісник НУВГП. Збірник наукових праць. – Вип.. 4(36). – Рівне. :НУВГП, 2006. – С. 102-109.
2. Сапсай Г.І., Чіпак В.П., Мельник Т.П. Автоматизація організації даних управління водними ресурсами в басейні ріки Тиса. // Вісник НУВГП. Збірник наукових праць. – Вип.. 4(36). – Рівне. :НУВГП, 2006. – С. 95-102.
3. Чіпак В.П., Мельник Т.П. Система протипаводкових заходів у басейні р.Боржава. – Рівне: Волинські обереги, 2008. – 202 с.

УДК 004.272

ПРИЙМАК А.В.

ОПТИМИЗАЦИЯ ВЫЧИСЛЕНИЙ НА CUDA ПРИ МОДЕЛИРОВАНИИ НЕУСТОЙЧИВОСТИ ЛЕНГМЮРОВСКИХ ВОЛН В ПЛАЗМЕ

Рассмотрена оптимизация вычислений при использовании технологии параллельного программирования на графических процессорах CUDA для моделирования параметрической неустойчивости ленгмюровских волн в плазме. Определены затраты времени на различные этапы численного моделирования. Проанализирована эффективность применения различных способов оптимизации вычислений. Показано, что для данной задачи наибольший прирост скорости вычислений можно получить при оптимизации использования тригонометрических функций.

Optimization of parallel computing on graphic processors with CUDA was considered for simulation of parametric instability of Langmuir waves in plasma. The time required for different stages of the numerical simulation have been defined. The success of different optimization methods was analyzed. It is shown that the greatest increase in computational speed can be obtained by optimizing the use of trigonometric functions.

1. Введение

В настоящее время растет популярность использования графических процессоров (graphics processing unit, GPU) для ускорения численных экспериментов, т.к. GPU имеют производительность параллельной архитектуры [1]. Такой подход использования GPU для вычислений общего назначения, которые обычно проводит центральный процессор (central processing unit, CPU), известен как general-purpose computing on graphics processing units (GPGPU) [2].

Изучение гибридных моделей параметрической неустойчивости ленгмюровских волн в плазме требует проведения большого количества численных экспериментов и, соответственно, значительных затрат времени. Затраты времени зависят от количества ионов в модели и от других параметров, и составляют обычно несколько часов на один численный эксперимент. Интерес к параметрической неустойчивости ленгмюровских волн обусловлен возможностями нагрева плазмы в установках термоядерного синтеза. Аппарат описания параметрической неустойчивости был создан в работах В.П. Силина [3,4] и В.Е. Захарова [5]. В настоящей работе модели параметрической неустойчивости ленгмюровских волн в плазме относятся к малоизученным гибридным, когда электроны описаны гидродинамически, а ионы описаны кинетически, крупными частицами.

Для проведения численных экспериментов выбрана compute unified device architecture (CUDA) - архитектура параллельных вычислений, разработанная компанией NVIDIA, и одна из основных GPGPU технологий [6].

При использовании технологии CUDA важную роль играет оптимизация, благодаря которой скорость вычислений можно увеличить в несколько раз [6-8].

2. Постановка задачи

При разработке алгоритма и программного обеспечения для проведения численных экспериментов необходимо учесть основные рекомендации по оптимизации вычислений, детально описанные в [6-8]. Необходимо оценить возможность переноса вычислений на GPU и возможность их распараллеливания, минимизировать обмен данными между памятью GPU и оперативной памятью (ОЗУ), выбрать правильные параметры распараллеливания и использовать для вычислений быстрые виды памяти GPU.

Необходимо оценить затраты времени на различные этапы моделирования и при оптимизации уделить внимание наиболее продолжительным этапам. При этом проведение оптимизации не должно сильно влиять на точность вычислений (не более чем на 1-2%) и не должно усложнять код программы, т.к. изучение гибридных моделей параметрической неустойчивости ленгмюровских волн требует периодического изменения некоторых параметров и уравнений.

Гибридные модели Силина и Захарова, результаты моделирования которых приведены в [9], состоят из уравнений амплитуды e_n, фазы ψ_n, реальной и мнимой частей плотности ионов M_{nr} и M_{ni}, напряженности электрического по-

ля $\bar{E}_{nr}$ и $\bar{E}_{ni}$ (число уравнений каждого типа равно числу мод, т.е. 400) и уравнений движения ионов с координатами и скоростями (ξ_s, v_S), число которых равно 20000. Также имеются два уравнения для описания волны накачки, амплитуда и фаза которой a_0, ϕ. Математические модели представлены в [10]. Динамика процессов параметрической неустойчивости в гибридных моделях Силина и Захарова подобна вследствие подобия систем уравнений [10].

Численный эксперимент представляет собой решение задачи Коши методом Эйлера. Начальные условия и параметры численных расчетов взяты следующие. Количество крупных частиц, моделирующих ионы $0 < s \le S = 20000$, число мод спектра $-N < n < N$, $N = S/100$, $a_0(0) = 0.06$, $d\xi_s / d\tau|_{\tau=0} = v_s|_{\tau=0} = 0$. Крупные частицы, моделирующие ионы, равномерно распределены на интервале $-1/2 < \xi < 1/2$. Начальная амплитуда ВЧ моды задается формулой $e_n|_{\tau=0} = e_{n0} = (2+g)\cdot 10^{-3}$ в модели Силина и формулой $e_n|_{\tau=0} = e_{n0} = (0.5+g)\cdot 10^{-4}$ в модели Захарова, где $g \in [0;1]$ - случайное число, $\psi_n|_{\tau=0}$ случайным образом распределялись в интервале $0 \div 2\pi$. Параметры легких ионов в модели Силина: $\frac{m_e}{M}\cdot\frac{\omega_p^2}{\delta^2} = 0.43$, $\frac{\delta}{\omega_0} = 0.034$, $\frac{\omega_0}{\delta} = 29.4$. Параметры легких ионов в модели Захарова: $\frac{m_e}{M}\cdot\frac{\omega_p^2}{\delta^2} = 20$, $\frac{\delta}{\omega_0} = 3.5\cdot 10^{-3}$, $\frac{\omega_0}{\delta} = 282.6$. Выбранные начальные условия отвечают равновесному состоянию электронов и ионов плазмы.

3. Реализация уравнений моделей с помощью технологии JCUDA

Программа, реализующая математическую модель задачи, создана с использованием технологии java compute unified device architecture (JCUDA). JCUDA обеспечивает взаимодействие с CUDA из Java-программы [11]. Вычисления производятся параллельно на GPU GeForce GT630 (GV-N630D3-2GL). Управление выполнением кода на GPU задается в Java-программе как в [12,13]. Код для выполнения на GPU пишется в виде CUDA модулей (.cu) без применения специализированных библиотек на основе CUDA API. При выполнении программы расчет всех уравнений моделей происходит на GPU. Алгоритм вычислений на GPU детально описан в [13].

Все вычисления выполняются в двойной точности (double), т.к. проведенные вычисления в одинарной точности (float) меняют динамику процесса.

На CUDA явно написан код, рассчитывающий значения функций Бесселя, т.к. встроенные в CUDA функции Бесселя дают недостаточную точность [1], близкую к одинарной. Другие используемые математические функции имеют двойную точность вычислений с малой погрешностью.

Наряду с параллельными вычислениями на GPU проведены непараллельные вычисления на CPU, которые подтвердили высокую скорость и точность вычислений на GPU [13]. Установлено, что скорость вычислений на GPU в 37 раз (модель Силина) и в 35 раз (модель Захарова) выше, чем на одном ядре CPU AMD Athlon 64 X2 с частотой ядра 2.2ГГц. Результаты вычислений на GPU и CPU отличаются на 1.5% в модели Силина и на 0% в модели Захарова. Сравнение результатов проводилось по распределению ионов $S(\xi)$ в момент времени, когда энергия ионов, пропорциональная величине $\sum_s (d\xi_s/dt)^2$ уже не возрастает. Отличия между случаями зависят от момента времени, с течением времени отличия возрастают.

4. Применение основных способов оптимизации вычислений на CUDA

4.1. Минимизация обмена данными между CPU и GPU

Для лучшей производительности приложений важно свести к минимуму передачу данных между памятью GPU и ОЗУ, даже если это означает запуск участков кода на GPU, которые не демонстрируют ускорение по сравнению с запуском на CPU [8]. Так, вычисление параметров $\partial a_0, \partial\phi$ происходит на GPU непараллельно на одном ядре, т.к. они описываются только двумя уравнениями. Вычисление $\partial a_0, \partial\phi$ на CPU происходило бы быстрее, однако требовало бы при вычислении для каждого момента времени передачи большинства массивов в ОЗУ, что привело бы к затратам времени на обмен данными. Как будет показано дальше, расчет $\partial a_0, \partial\phi$ занимает незначительное время,

поэтому не является узким местом в производительности.

Исходные данные для расчета загружаются в память GPU единожды перед началом моделирования. Далее обмен данными между CPU и GPU отсутствует кроме тех моментов времени, результаты которых отображаются графически. Дополнительным преимуществом такого подхода является то, что часть данных существует только в памяти GPU и не дублируется в ОЗУ для экономии памяти. Дублирование данных можно будет избежать, используя унифицированную виртуальную память, поддержка которой будет добавлена в CUDA Toolkit 6.0.

4.2. Максимальное распараллеливание вычислений

Вычисления для всех типов уравнений, кроме расчета параметров $\partial a_0, \partial\phi$, происходят параллельно. Это означает, что для вычисления одного уравнения создается один поток (thread). Выполнение потока представляет собой вызов функции, на вход которой подаются необходимые для вычисления данные, а на выходе возвращается результат вычисления. Функция имеет общий вид для всех уравнений одного типа.

4.3. Определение оптимальных размеров сетки потоков

Модель программирования в CUDA предполагает группирование потоков. Потоки объединяются в блоки потоков (thread block). Программа (ядро, kernel) исполняется над сеткой (grid) блоков потоков.

Для уравнений амплитуды ∂e_n, фазы $\partial\psi_n$, плотности ионов M_{nr} и M_{ni}, напряженности электрического поля $\bar{E}_{nr}$ и $\bar{E}_{ni}$ и функций Бесселя J_0, J_1, J_n расчеты можно выполнять с одинаковыми параметрами распараллеливания. Для расчета этих уравнений создается сетка, состоящая из одного блока потоков: размерность сетки $\dim3(1,1,1)$ и размерность блока $\dim3(N,1,1)$. Такой алгоритм поддерживает число мод до 1024, т.к. CUDA имеет ограничение до 1024 потоков в одном блоке. При проведении вычислений отдельно для отрицательных и отдельно для положительных мод, т.е. создании двух сеток размером $N/2$, скорость расчета уравнений падает на 38%.

Для расчета параметров $\partial a_0, \partial\phi$ задаются следующие параметры распараллеливания: размерность сетки $\dim3(1,1,1)$ и размерность блока $\dim3(1,1,1)$, что означает создание всего одного потока, т.е. непараллельное вычисление.

Для расчета уравнений движения ионов dv_s и $d\xi_s$, а также поля ВЧ давления на ионы $\bar{E}\xi_s$ выбрана размерность сетки $\dim3(S/500,1,1)$ и размерность блока $\dim3(500,1,1)$. При числе потоков в блоке меньше 500 скорость вычисления уравнений падает на несколько процентов.

4.4. Предварительный расчет функций Бесселя в гибридной модели Силина

Как было сказано выше, расчет функций Бесселя происходит явно для поддержания высокой точности вычислений. При этом для каждого момента времени для каждой моды необходимо вычислить по одной функции Бесселя J_0, J_1, J_n. Поэтому перед началом расчета уравнений для каждого момента времени происходит предварительный расчет функций Бесселя, и далее полученные значения используются в уравнениях на протяжении счета для данного момента времени.

4.5. Оптимизация использования памяти

Все загружаемые в память GPU данные хранятся в глобальной памяти, размер которой в используемом GPU составляет 2Гбайт. Однако глобальная память обладает низкой скоростью доступа [7]. GPU включает и другие типы памяти, которые имеют небольшой объем и высокую скорость доступа, в первую очередь разделяемую (shared) память и константную (constant) памятью. Для использования разделяемой памяти необходимо проводить дополнительное копирование данных, но даже при этом скорость вычислений должна повыситься. Однако попытки использовать разделяемую и константную память дали эффект около 1%. В разделяемую память были положены несколько массивов, например массивы со значениями функций Бесселя. В константную память были положены постоянные значения, например число π и другие константы.

4.6. Уменьшение количества вычислений

В некоторых местах программного кода многократно выполняются одинаковые вычис-

ления, поэтому применен способ для сокращения числа вычислений. Например, при расчете dv_S присутствует повторяющееся в цикле вычисление $2\cdot\pi\cdot\xi_S$, поэтому была создана переменная $p=2\cdot\pi\cdot\xi_S$, которая затем использовалась в цикле. Подобным образом были сделаны еще несколько оптимизаций, однако их суммарный эффект не превышает 1% и при этом усложняет программный код.

5. Определение затрат времени на различные этапы алгоритма

После оптимизации проводится определение затрат времени на выполнение разных этапов алгоритма (табл. 1). Измерение времени происходит в Java-программе с помощью метода System.currentTimeMillis().

Табл. 1 – Затраты времени на выполнение этапов алгоритма после оптимизации

	Время выполнения, минут	
	Модель Силина	Модель Захарова
Этап 1. Расчет дополнительных параметров для построения графиков и их рисование (CPU)	2.7	2.5
Этап 2. Расчет J_0, J_1, J_n, ∂e_n, $\partial\psi_n$, M_{nr}, M_{ni}, $\bar{E}_{nr}$, $\bar{E}_{ni}$ (GPU)	12.3	1.3
Этап 3. Расчет dv_S, $d\xi_S$ (GPU)	5	0.6
Этап 4. Расчет накачки $\partial a_0, \partial\phi$ непараллельно (GPU)	0.7	0.1
Этап 5. Обновление значений параметров с учетом посчитанных на данном шаге (GPU)	0.4	0.1
Этап 6. Получение результатов из GPU (GPU→CPU)	0.01	0.01
Этап 7. Расчет $\bar{E}\xi_S$ (GPU)	0.1	0.1
Общее время выполнения на GPU	18.5 (87.3% общего времени)	2.2 (47% общего времени)
Общее время выполнения на CPU	2.7 (12.7% общего времени)	2.5 (53% общего времени)
Общее время выполнения моделирования	21.2	4.7

Определение затрат времени показывает успешность минимизации обмена данными между GPU и CPU, а также малые затраты времени на расчет накачки, который проводится непараллельно.

Недостатком является высокая доля (53%) времени выполнения на CPU в модели Захарова при том, что расчет всех уравнений был перенесен на GPU. Однако на CPU происходит подготовка и рисование графиков и интерфейса (200 раз в течении численного эксперимента). Меньшая доля времени выполнения на CPU в модели Силина объясняется большим числом вычислений на GPU, а именно тем, что перед получением результатов из GPU происходит расчет для 40 моментов времени, в то время как у Захарова для 5 моментов времени. Скорость вычислений для одного момента времени в моделях Силина и Захарова практически одинакова.

Значительная доля времени моделирования приходится на 2-й и 3-й этапы, т.е. непосредственно на вычисление уравнений. При этом определено, что 85% этого времени занимает вычисление функций синусов и косинусов.

6. Оптимизация использования тригонометрических функций

Т.к. определено, что затраты на вычисление синусов и косинусов составляют 85% от общего времени вычисления уравнений, были опробованы различные способы оптимизации использования тригонометрических функций.

6.1. Использование альтернативных тригонометрических функций

Существуют функции $\sin pi(\mathrm{x}) = \sin(\pi\cdot\mathrm{x})$, $\cos pi(\mathrm{x}) = \cos(\pi\cdot\mathrm{x})$, однако в моделях Силина и Захарова только девятая часть функций имеет такой вид, поэтому эффект от их использования составляет лишь до 2%.

Также существует функция $\sin\cos(\mathrm{x})$, одновременно рассчитывающая синус и косинус. Эффект от ее использования составляет около

50%. Однако существенным недостатком функции является необходимость для каждого вычисления выделять память под хранение значений синуса и косинуса, что усложняет ее использование.

6.2. Создание таблицы значений тригонометрических функций

Сделана попытка делать предварительный расчет синусов и косинусов в каждый момент времени как это делается с функциями Бесселя, а затем использовать посчитанные значения в уравнениях. Четверть функций имеют подобную себе функцию с одинаковым аргументом, однако применение этого способа оптимизации дает эффект до 5% и усложняет код программы.

6.3. Понижение точности вычислений синуса и косинуса

В CUDA каждая математическая функция имеет несколько реализаций, отличающихся по точности результата вычисления – double-функции (например $\sin(x)$, $\cos(x)$), float-функции (например $\mathrm{sinf}(x)$, $\mathrm{cosf}(x)$), функции пониженной точности (например $__\mathrm{sinf}(x)$, $__\mathrm{cosf}(x)$). Таблицы функций различной точности приведены в [1].

Т.к. основные затраты времени идут на вычисление тригонометрических double-функций, сделана попытка использовать аналоги этих функций меньшей точности. При этом остальные вычисления продолжают проводиться в двойной точности.

В отличие от CPU, GPU не оптимизированы для вычислений двойной точности. В связи с этим, скорость вычислений на GPU в двойной точности меньше, чем в одинарной.

Так, применение float-функций позволяет ускорить вычисление уравнений на 60%. После окончания моделирования была проверена точность вычислений и отклонения результатов составили 1.2% от случая с использованием double-функций синуса и косинуса.

Применение функций синуса и косинуса пониженной точности позволяет ускорить вычисление уравнений на 70%. Отклонения результатов составили 1.7% от случая с использованием double-функции синуса и косинуса.

Кроме того, возможно комбинирование разных способов оптимизации тригонометрических функций. Однако этого не было выполнено в настоящей работе, т.к. основной эффект достигается только благодаря понижению точности вычислений синуса и косинуса.

7. Выводы

Важность оптимизации вычислений на CUDA подтверждена в настоящей работе. В то же время основные способы оптимизации показали разные результаты. Это говорит о том, что эффективность различных способов оптимизации зависит от специфики задачи.

В работе описаны результаты применения основных способов оптимизации CUDA-вычислений. Наибольший прирост скорости вычислений дает использование функций синуса и косинуса пониженной точности.

Дальнейшая оптимизация параллельных вычислений на CUDA при моделировании параметрической неустойчивости ленгмюровских волн в плазме возможна при использовании новых технологий, связанных с CUDA, динамического параллелизма и унифицированной виртуальной памяти.

Автор выражает благодарность Куклину В.М. за внимание к работе.

Список литературы

1. CUDA C Programming Guide. - NVIDIA Corporation [Электронный ресурс]. - Режим доступа: http://docs.nvidia.com/cuda/cuda-c-programming-guide. - Электрон. версия, 2013. – HTML формат.
2. CUDA. - Wikipedia [Электронный ресурс]. - Режим доступа: http://en.wikipedia.org/wiki/CUDA. - Электрон. версия, 2013. – HTML формат.
3. Silin V.P. Parametric resonance in plasma./ JETP. -1965. – Vol. 48. N6. – P. 1679-1691.
4. Silin V.P. Parametric Influences of high-energy Radiation on Plasma. – Moscow: Nauka, 1973.
5. Zakharov V.E. Weak-turbulence spectrum in a plasma without a magnetic field //Sov. Phys. JETP. –1967. – Vol. 24(2), P.455-459. The Instability of Waves in Nonlinear Dispersive Media // Sov. Phys. JETP. -1967. – Vol. 24. – P.740. Collapse of Langmuir Waves // Sov. Phys. JETP. -1972. –Vol. 35(5). –P.908-914.
6. Д. Сандерс, Э. Кэндрот. Технология CUDA в примерах. Введение в програмирование графических процессов: Пер. с англ. Слинкина А.А., научный редактор Боресков А.В. М.: ДМК Пресс, 2011. - 232 с.

7. Боресков А.В., Харламов А.А. Основы работы с технологией CUDA. М.: ДМК Пресс, 2010. - 232с.
8. CUDA C Best Practices Guide. - NVIDIA Corporation [Электронный ресурс]. - Режим доступа: http://docs.nvidia.com/cuda/cuda-c-best-practices-guide. - Электрон. версия, 2013. – HTML формат.
9. Belkin E.V., Kirichok A.V., Kuklin V.M., Pryjmak A.V., Zagorodny A.G. Dynamics of ions during development of parametric instability of Langmuir waves // Вопросы атомной науки и техники. Серия «Плазменная электроника и новые методы ускорения». 2013, №8, с.260-266.
10. Kuklin V.M. Similarity of 1D Parametric Instability description of Langmuir waves. / The Journal of Kharkiv National University, physical series: Nuclei, Particles, Fields. – 2013. - №1041. - Iss. 2 (58). - P.20-32.
11. Marco Hutter. JCUDA. - jcuda.org, 2008 [Электронный ресурс]. - Режим доступа: http://jcuda.org/. - Электрон. версия, 2008. - HTML формат.
12. Мишин А.В., Приймак А.В. Моделирование неустойчивости движущегося в плазме сгустка заряженных частиц. // Вестник Харьковского национального университета, – 2012. – № 1037. Сер. "Математическое моделирование. Информационные технологии. Автоматизированные системы управления", вып. 20. – С. 133-145.
13. Приймак А.В. Использование технологии JCUDA для моделирования динамики ионов при развитии параметрической неустойчивости ленгмюровских волн. – Информатика и компьютерные технологии / Сборник трудов IX международной научно-технической конференции студентов, аспирантов и молодых ученых. – 5-6 ноября 2013, Донецк, ДонНТУ. – 2013. – С. 200-204.

УДК 681.3.042

ФЕДОТОВА-ПИВЕНЬ И.Н.,
ПИВЕНЬ О.Б.

ПРОГРАММНОЕ МОДЕЛИРОВАНИЕ СОВМЕЩЕННОГО ВО ВРЕМЕНИ СЛОЖЕНИЯ ДВАДЦАТИ ЦЕЛЫХ ПОЛОЖИТЕЛЬНЫХ ЧИСЕЛ В ИЗБЫТОЧНОЙ РЕКУРРЕНТНОЙ СИСТЕМЕ СЧИСЛЕНИЯ ТРЕТЬЕГО ПОРЯДКА

В статье в рамках программных моделей рассмотрены сравнения быстродействий совмещенного во времени сложения 20-ти целых положительных 32-/16-разрядных чисел в избыточной рекуррентной системе счисления третьего порядка с алфавитом {0, 1}, образованной линейным рекуррентным соотношением $B_n=B_{n-1}+3B_{n-3}+2B_{n-4}$ с начальными значениями 1 1 1 1 2 4 8 и поочередного сложения 20-ти целых положительных 32-/16-разрядных двоичных чисел по стандартному алгоритму Уоллеса.

In an article in the framework of program models examined compare the performance of coincident in time addition of twenty positive integers 32-/16-bit numbers in the redundant recurrent numeration system of the third order created by the linear recurrence relation $B_n=B_{n-1}+3B_{n-3}+2B_{n-4}$, with starting values 1 1 1 1 2 4 8 and addition by turns of twenty integer positive 32-/16-bit binary numbers on a standard algorithm of Wallace.

1. Введение

Постановка проблемы. Повышение производительности и достоверности вычислений в компьютерных системах, включающее повышение быстродействия и обеспечение высокого уровня их надежности, является одной из важнейших задач современной цифровой электроники.

Для решения этой задачи предусматривается применение аппаратной и информационной избыточности, а следовательно и использование избыточных систем счисления (ССЧ) [1]. Благодаря множественности кодовых представлений одного и того же числа в избыточной системе счисления появляется возможность распараллеливания вычислений [2] и увеличение быстродействия арифметических устройств. Также, одним из возможных способов повышения скорости вычислений, выполняемых над большими объемами числовых данных, является использование совмещенного во времени выполнения операций [3]. Поэтому совмещенное выполнение операций можно применить для решения проблемы многооперандного сложения. Многооперандное сложение является фундаментальной проблемой [4,5], так как с увеличением числа операндов растет логическая сложность сумматора [6]. Таким образом, создание более быстрых алгоритмов совмещенного во времени многооперандного сложения в избыточных ССЧ является актуальной задачей.

Анализ последних исследований и публикаций. Разновидностью избыточных ССЧ являются линейные избыточные рекуррентные системы счисления (ЛИРСЧ). В работе [7] был разработан программный комплекс генерации систем счисления с правилами совмещенного сложения (ССЛ) в них до 5-ти включительно слагаемых с использованием структурно-блочных кодов (СБК). Оказалось, что наиболее простые правила ССЛ до 5-ти слагаемых включительно ($k \subset (2; 3; 4; 5)$) имеет ССЧ с алфавитом {0, 1}, образованная линейным рекуррентным соотношением (ЛРС)

$$B_n=B_{n-1}+3B_{n-3}+2B_{n-4} \qquad (1)$$

и начальными значениями 1 1 1 1 2 4 8 (НЗ) (т.е. $B_0 = 1$, $B_1 = 1$, $B_2 = 1$, $B_3 = 1$, $B_4 = 2$, $B_5 = 4$, $B_6 = 8$), для которой выполняются следующие правила ССЛ:

$$\begin{cases} 0+0=0; \\ B_n+0=B_n; \\ 2B_n=B_{n+1}; \\ 3B_n=B_n+B_{n+1}; \\ 4B_n=B_{n+2}; \\ 5B_n=B_n+B_{n+2}; \end{cases} \qquad (2)$$

Но вопрос совмещенного сложения более 5-ти двоичных слагаемых в этой ССЧ не рассматривался.

2. Постановка задачи.

С использованием СБК создать программную модель совмещенного сложения 20-ти целых положительных 32-/16-разрядных чисел в ЛИРСЧ 3-го порядка, образованной ЛРС $B_n= B_{n-1} + 3B_{n-3}+2B_{n-4}$ с НЗ, с алфавитом {0, 1} и сравнить ее быстродействие с быстродействием программной модели поочередного сложения 20-ти целых положительных 32-/16-разрядных двоичных чисел по алгоритму Уоллеса.

3. Доказательство правил совмещенного во времени сложения в избыточной рекуррентной системе счисления третьего порядка.

В [8] рассмотрен метод поочередного сложения 12-ти слагаемых с использованием дерева Уоллеса (рис.2 из [8]). При этом использовано 11 цифровых компрессоров типа 3:2 и один быстрый сумматор с распространением переноса (CPA - carry propagate adder) [8]. Используются методы оптимизации (теория графов) [9,10] для минимизации числа цифровых компрессоров и связей между ними для поочередного сложения более 3-х слагаемых. По аналогии с рис. 2 из [8] создадим дерево Уоллеса поочередного сложения 20-ти слагаемых и используем его для сравнения быстродействия совмещенного и поочередного сложения 20-ти слагаемых :

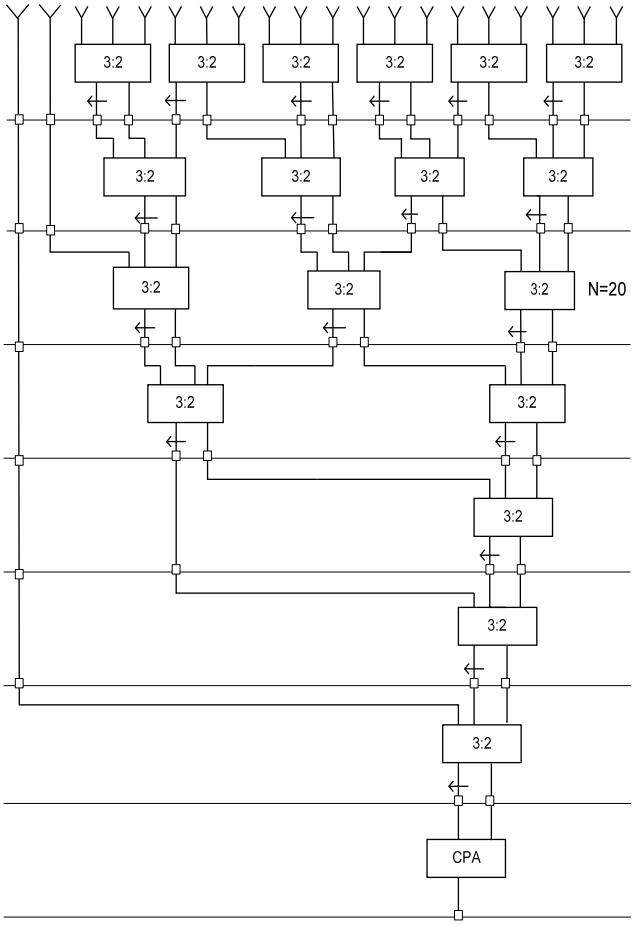


Рис.1. Поочередное сложение 20-ти слагаемых с использованием дерева Уоллеса по аналогии с рис. 2 из [8].

Из рис.1 видно, что для поочередного сложения 20-ти слагаемых используется 18 цифровых компрессоров типа 3:2 и один быстрый сумматор с распространением переноса.

Используя возможность совмещенного сложения, программный комплекс [7] был усовершенствован для получения ССЧ, в которых возможно совмещенное во времени сложение до 20-ти слагаемых включительно ($k \subset (2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12; 13; 14; 15; 16; 17; 18; 19; 20)$). Оказалось, что наиболее простые правила совмещенного сложения до 20-ти слагаемых включительно (3) имеет ССЧ, созданная ЛРС (1) с начальными значениями НЗ при $n \geq 7$.

$$
\begin{cases}
0 = 0 + 0; \\
B_n = B_n + 0; \\
2B_n = B_{n+1}; \\
3B_n = B_n + B_{n+1}; \\
4B_n = B_{n+2}; \\
5B_n = B_n + B_{n+2}; \\
6B_n = B_{n+1} + B_{n+2}; \\
7B_n = B_n + B_{n+1} + B_{n+2}; \\
8B_n = B_{n+3}; \\
9B_n = B_n + B_{n+3}; \\
10B_n = B_{n+1} + B_{n+3}; \\
11B_n = B_n + B_{n+1} + B_{n+3}; \\
12B_n = B_{n+2} + B_{n+3}; \\
13B_n = B_n + B_{n+2} + B_{n+3}; \\
14B_n = B_{n+1} + B_{n+2} + B_{n+3}; \\
15B_n = B_n + B_{n+1} + B_{n+2} + B_{n+3}; \\
16B_n = B_{n+4}; \\
17B_n = B_n + B_{n+4}; \\
18B_n = B_{n+1} + B_{n+4}; \\
19B_n = B_n + B_{n+1} + B_{n+4}; \\
20B_n = B_{n+2} + B_{n+4};
\end{cases}
\qquad (3)
$$

Докажем правила сложения (3) для ССЧ, заданной ЛРС (1) и НЗ. В работе [11] для ЛИРСЧ 3-го порядка (1) доказано правила совмещенного сложения до 5-ти слагаемых включительно (т.е. первые 6 равенств системы (3)). Докажем правила совмещенного сложения до 20-ти слагаемых включительно в ЛИРСЧ, заданной ЛРС (1) и НЗ, т.е. последние 15 равенств системы (3).

Доказательство. Седьмое правило в системе (3), $6B_n=B_{n+1}+B_{n+2}$ получается из правила 4 так: $6B_n=2 \cdot 3B_n = 2B_n + 2B_{n+1}$. По правилу 3 имеем: $2B_n = B_{n+1}$ и $2_{Bn+1} = B_{n+2}$. Подстановка этих двух равенств вместо $2B_n$ и $2B_{n+1}$ в 6Bn дает правило 7. Правило 8 получается из 7-го: $7B_n = 6B_n + B_n = B_n + B_{n+1} + B_{n+2}$. Правило 9 получается из 8-го так: $8B_n = B_n + B_n + B_{n+1} + B_{n+2}$. По правилу 3 имеем: $2B_n = B_{n+1}$, $2B_{n+1} = B_{n+2}$ і $2B_{n+2} = B_{n+3}$. Последовательное применение этих равенств дает: $8B_n = B_{n+1} + B_{n+1} + B_{n+2} = B_{n+2}+ B_{n+2} = B_{n+3}$. Правило 10 получается из 9-го: $9B_n = 8B_n + B_n = B_n + B_{n+3}$. Правило 11 получается из 10-го так:

$10B_n = B_n + B_n + B_{n+3}$. По правилу 3 имеем: $2B_n = B_{n+1}$. Поэтому $10B_n = B_{n+1} + B_{n+3}$. Правило 12 получается из 11-го так: $10B_n + B_n = B_n + B_{n+1} + B_{n+3}$. Поэтому $11B_n = B_n + B_{n+1} + B_{n+3}$. Правило 13 получается из 12-го так: $12B_n = 2B_n + B_{n+1} + B_{n+3}$. По правилу 3 имеем: $2B_n = B_{n+1}$ и $2B_{n+1} = B_{n+2}$. Подстановка этих двух равенств вместо $2B_n$ и $2B_{n+1}$ в $12B_n$ дает правило 13: $12B_n = B_{n+2} + B_{n+3}$. Правило 14 получается из 13-го так: $12B_n + B_n = B_n + B_{n+2} + B_{n+3}$. Поэтому $13B_n = B_n + B_{n+2} + B_{n+3}$. Правило 15 получается из 14-го так: $13B_n + B_n = 2B_n + B_{n+2} + B_{n+3}$. По правилу 3 имеем: $2B_n = B_{n+1}$. Поэтому
$14B_n = B_{n+1} + B_{n+2} + B_{n+3}$. Правило 16 получается из 15-го так: $14B_n + B_n = B_n + B_{n+1} + B_{n+2} + B_{n+3}$. Поэтому
$15B_n = B_n + B_{n+1} + B_{n+2} + B_{n+3}$. Правило 17 получается из 16-го так: $15B_n + B_n = B_n + B_n + B_{n+1} + B_{n+2}$. По правилу 3 имеем: $2B_n = B_{n+1}$, $2B_{n+1} = B_{n+2}$, $2B_{n+2} = B_{n+3}$ и $2B_{n+3} = B_{n+4}$. Последовательное применение этих равенств дает: $16B_n = 2B_n + B_{n+1} + B_{n+2} + B_{n+3} = B_{n+1} + B_{n+1} + B_{n+2} + B_{n+3} = 2B_{n+1} + B_{n+2} + B_{n+3} = B_{n+2} + B_{n+2} + B_{n+3} = 2B_{n+2} + B_{n+3} = B_{n+3} + B_{n+3} = B_{n+4}$. Правило 18 получается из 17-го так: $16B_n + B_n = B_n + B_{n+4}$. Поэтому $17B_n = B_n + B_{n+4}$. Правило 19 получается из 18-го так: $17B_n + B_n = 2B_n + B_{n+4}$. По правилу 3 имеем: $2B_n = B_{n+1}$. Поэтому $18B_n = B_{n+1} + B_{n+4}$. Правило 20 получается из 19-го так: $18B_n + B_n = B_n + B_{n+1} + B_{n+4}$. Поэтому $19B_n = B_n + B_{n+1} + B_{n+4}$. Правило 21 получается из 20-го так: $20B_n = 19B_n + B_n = 2B_n + B_{n+1} + B_{n+4}$. По правилу 3 имеем: $2B_n = B_{n+1}$ и $2B_{n+1} = B_{n+2}$. Подстановка этих двух равенств вместо $2B_n$ и $2B_{n+1}$ в $20B_n$ дает правило 21: $20B_n = B_{n+1} + B_{n+1} + B_{n+4} = 2B_{n+1} + B_{n+4} = B_{n+2} + B_{n+4}$. Этим систему правил совмещенного сложения (3) для ЛИРСЧ (1) с НЗ доказано.

4. Программная модель сравнения быстродействия совмещенного во времени сложения в избыточной рекуррентной системе счисления третьего порядка.

Для сравнения быстродействия совмещенного сложения 20-ти целых положительных 32-/16-разрядных чисел в ЛИРСЧ 3-го порядка, образованной ЛРС (1) с НЗ 1 1 1 1 2 4 8, алфавитом {0, 1} с быстродействием поочередного сложения 20-ти целых положительных 32-/16-разрядных двоичных чисел по часто используемому алгоритму Уоллеса создадим программную модель с помощью электронных таблиц Excel 2003 и встроенного языка программирования VBA.

Совмещенное во времени сложение до 20-ти целых положительных n - разрядных слагаемых включительно в ЛИРСЧ 3-го порядка с алфавитом {0, 1}, образованной ЛРС (1) с НЗ и по правилам (3) увеличивает разрядность суммы до n + 5 разрядов.

На рис. 2 приведен алгоритм программной модели совмещенного сложения 20-ти слагаемых в ЛИРСЧ 3-го порядка с алфавитом {0, 1} и НЗ, сгенерированной ЛРС (1) и алгоритм программной модели поочередного сложения 20-ти слагаемых в обычной двоичной ССЧ по алгоритму Уоллеса. Прямое применение правил совмещенного сложения (3) в ЛНРСЧ показано на рис. 2 в виде первого блока операторов, обведенного пунктиром. Моделирование работы сумматора с распространением переноса показано на рис. 2 в виде второго блока операторов, обведенного пунктиром. При моделировании работы сумматора с распространением переноса использована формула [12]:

$$B_n = \sum_{m=k}^{n-1} B_m + B_k \quad , \qquad (4)$$

где $k \geq 0$; $k \leq m \leq n - 1$; $n \geq 7$; $B_k \neq 0$.

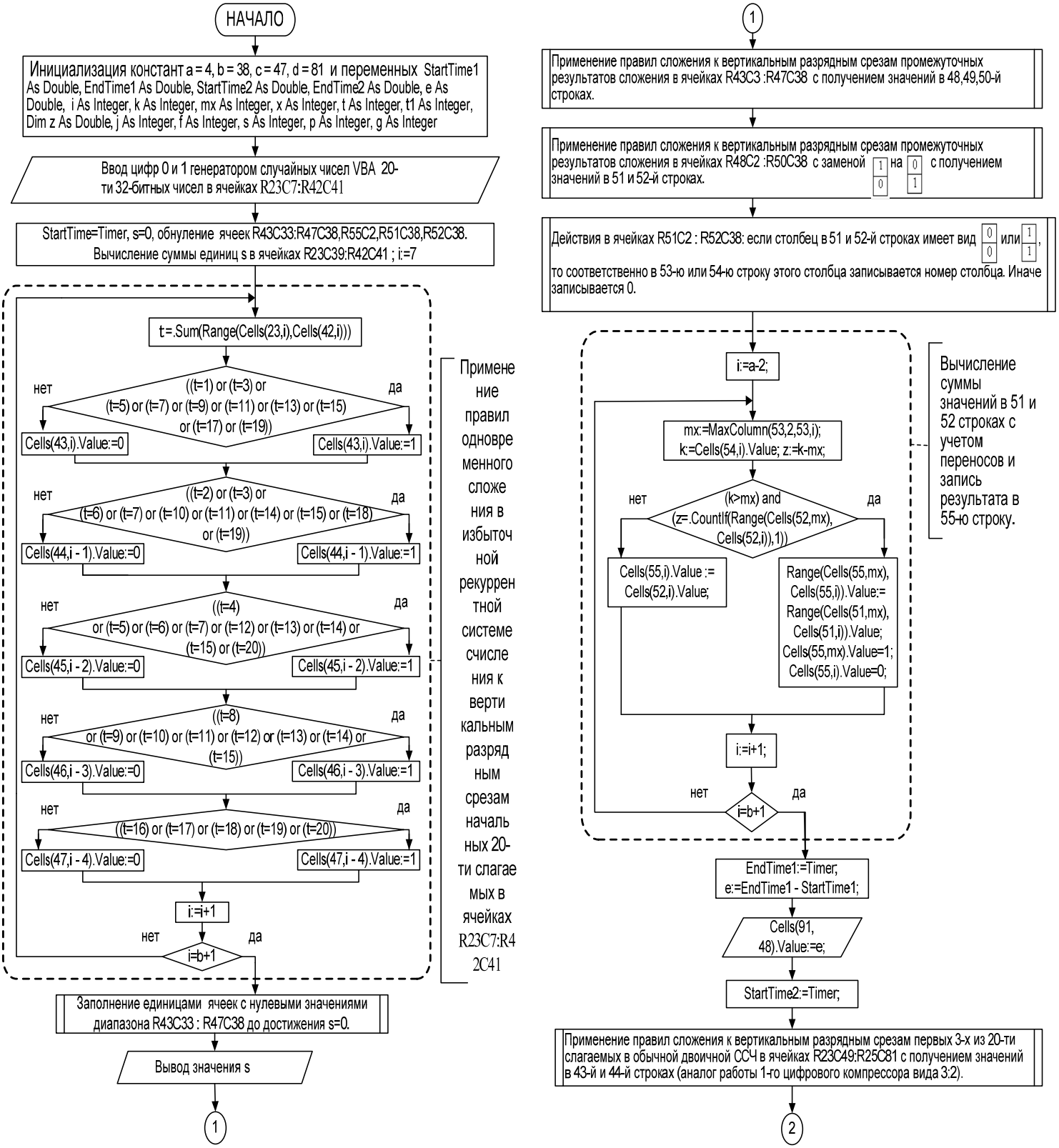
Сравнение скоростей вычисления суммы 20-ти слагаемых в обоих представленных в Excel моделях многооперандного сложения показало, что модель сложения в ЛИРСЧ для 32-разрядных целых положительных чисел требует меньше вычислений и на 73% быстрее, чем модель сложения в обычной двоичной ССЧ по алгоритму Уоллеса (определено как среднее значение времени вычисления суммы на 400 наборах по 20 случайных 32-разрядных слагаемых в каждом).

Сравнение скоростей вычисления суммы 20-ти слагаемых в этих же программных моделях, скорректированных для 16-разрядных целых положительных чисел показывает, что модель сложения в ЛНРСЧ на 73% быстрее, чем модель сложения в обычной двоичной ССЧ по алгоритму Уоллеса (определено как среднее значение времени вычисления суммы на 400 наборах по 20 случайных 16-разрядных слагаемых в каждом).

Во время вычисления суммы слагаемых в ЛИРСЧ происходит перевод результата в обычную двоичную ССЧ и полученный результат может быть использован без дополнительных преобразований в следующих вычислениях.

В программных моделях совмещенного сложения в ЛИРСЧ 3-го порядка 20-ти целых положительных слагаемых как для случая только 32-разрядных слагаемых, так и для случая только 16-разрядных слагаемых необходимо в 3,1 раза меньше ячеек памяти для хранения промежуточных результатов вычислений, чем в соответствующих программных моделях поочередного сложения в обычной двоичной ССЧ по алгоритму Уоллеса.

Итак, простые правила сложения (3) определяют метод совмещенного сложения до 20-ти слагаемых включительно в ЛИРСЧ 3-го порядка (1) при $n \geq 7$ с НЗ 1 1 1 1 2 4 8.



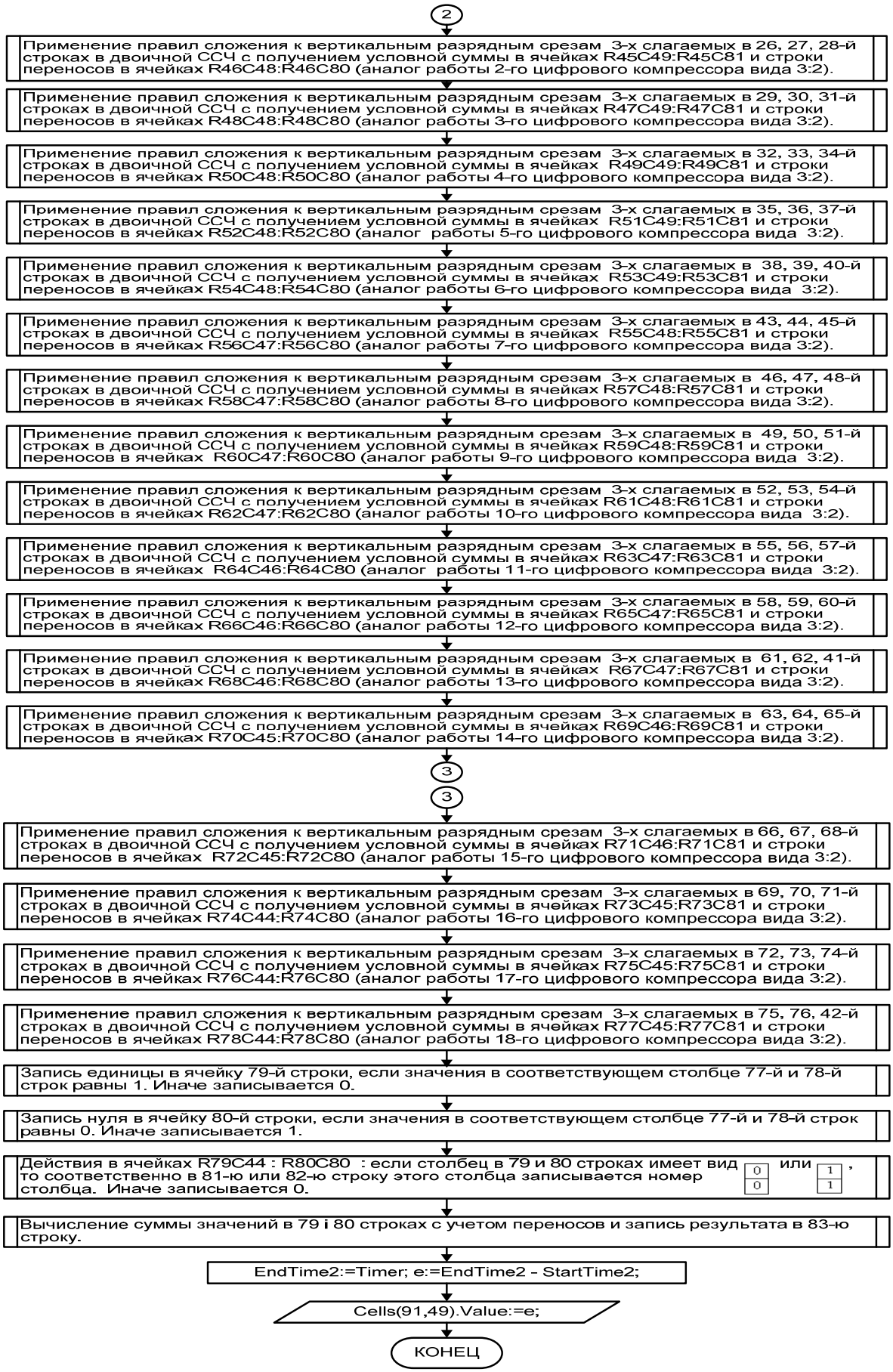


Рис. 2. Алгоритм программной модели совмещенного сложения 20-ти целых положительных 32-разрядных чисел в ЛИРСЧ 3-го порядка с алфавитом {0, 1} и НЗ 1 1 1 1 2 4 8, сгенерированной ЛРС (1) и алгоритм программной модели поочередного сложения 20-ти 32-разрядных слагаемых в обычной двоичной ССЧ.

Выводы

В результате проведенных исследований с использованием СБК определено, что модель совмещенного во времени сложения 20-ти слагаемых в ЛИРСЧ 3-го порядка с алфавитом {0, 1}, сгенерированной рекуррентным соотношением $B_n = B_{n-1} + 3B_{n-3} + 2B_{n-4}$ с НЗ 1 1 1 1 2 4 8 для 32-разрядных целых положительных чисел, требует меньше вычислений и на 73% быстрее, чем модель сложения в обычной двоичной ССЧ по алгоритму Уоллеса. Скорректированная для 16-разрядных целых положительных чисел модель совмещенного во времени сложения 20-ти слагаемых в этой же ЛИРСЧ на 79% быстрее, чем модель сложения в обычной двоичной ССЧ по алгоритму Уоллеса.

В программных моделях совмещенного сложения в ЛИРСЧ 3-го порядка 20-ти целых положительных слагаемых как для случая только 32-разрядных слагаемых, так и для случая только 16-разрядных слагаемых необходимо в 3,1 раза меньше ячеек памяти для хранения промежуточных результатов вычислений, чем в соответствующих программных моделях поочередного сложения в обычной двоичной ССЧ по алгоритму Уоллеса.

Практическое внедрение этой ЛИРСЧ необходимо проводить в зависимости от аппаратных возможностей и требований проектирования при условии повышения контролирующих свойств кодов. Эта ЛИРСЧ может быть использована на практике для более быстрой работы цифровых фильтров.

Список литературы

1. Брюхович Е. И. Экономическая стратегия разработки вычислительных систем: место и роль счислений. // Управляющие системы и машины. Научно-производственный журнал. №2 (106), 1990, февраль. – Институт кибернетики им. В. М. Глушкова АН УССР, с. 3-18.
2. A.Mignotte, J.M. Muller, O.Peyran. Synthesis for mixed arithmetic. //Ecole Normale Superieure de Lyon, Laboratoire de l'Informatique du Parallelisme, Unite de recherche associee au CNRS n° 1398, November, 1997, Research Report N° 97-41, pp. 1-24. или http://lara.inist.fr/bitstream/2332/689/ 1/LIP-RR1997-41.pdf
3. Лебедев С. А. Электронно-вычислительные машины / С. А. Лебедев // Сессия АН СССР по научным проблемам автоматизации производства. Пленарные заседания. – М.: АН СССР. – 1957. – Т. 1. – С. 162 – 180.
4. Chi-Hsiang Yeh, Benrooz Parhami. Efficient pipelined multi-operand adders with high throughput and low latency: designs and applications. Proc. 30th Asilomar Conf. Signals, Systems, and Computers, Pacific Grove, CA, 3-6 November 1996, pp. 894-898.
5. Мартинюк Т. Б. Рекурсивні алгоритми багатооперандної обробки інформації: [Монографія] / Т. Б. Мартинюк - Вінниця: "Універсум-Вінниця", 2000. -216 с. - ISBN 966 – 7199 – 98 - 3.
6. Wallace C.S. A suggestion for a fast multiplier. IEEE Transactions on Electronic Computers, C-13(2), February 1964, pp.14-17.
7. В.М Рудницький, І.М.Федотова-Півень. Метод підвищення швидкодії арифметичних пристроїв за рахунок суміщеного виконання операцій в структурно-блокових кодах./Системи обробки інформації. Збірник наукових праць. 2009, вип.4(78), с.117-119.
8. Martinez M. On the design of FPGA-based multioperand pipeline adders / M. Martinez, J. Valls, E. Boemo // Proceedings of the XII Design of Circuits and Integrated Systems Conference (DCIS'97). – Universidad de Sevilla, Seville, Spain, November 18-21, 1997. – P. 701-706.
9. Ienne P., Verma A.K. Arithmetic transformations to maximise the use of compressor trees. Second IEEE International Workshop on Electronic Design, Test and Applications, DELTA 2004, Perth, Australia, 28-30 January, pp.219-224.
10. Um J., Kim T., Liu C.L. Optimal allocation of carry-save adders in arithmetic optimization. 1999 InternationaL Conference on Computer Aided Design (ICCAD'99), San Jose, CA, 7-11 November, 1999, pp.410-413.
11. В.М. Рудницький, І.М. Федотова-Півень. Моделювання суміщеного додавання до п'яти доданків в надлишковій рекурентній системі числення 3-го порядку. //Системи управління, навігації та зв'язку. 2011, вип. 2(18), С.164-166.
12. В.М. Рудницький, І.М. Федотова-Півень. Програмна модель одночасного додавання п'яти додатніх цілих чисел в надлишковій рекурентній системі числення 3-го порядку. //Наука і техніка Повітряних Сил Збройних Сил України. 2011, вип. 2(6), С.158-161.

УДК 681.324

МИНУХИН С.В.,
ЗНАХУР С.В.,
КОРОВИН А.В.

КООРДИНИРОВАННЫЕ СТРАТЕГИИ ПЛАНИРОВАНИЯ ЗАДАНИЙ НА ОСНОВЕ ОБОБЩЕННОГО СТАТИСТИЧЕСКОГО АНАЛИЗ ИХ ХАРАКТЕРИСТИК В РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

В статье рассмотрены вопросы разработки и оценки стратегий планирования заданий в распределенных вычислительных системах на основе обобщенного статистического анализа характеристик заданий. Показано, что в условиях динамически изменяющейся нагрузки на вычислительную систему представляется эффективным для оценки производительности системы использование синтетических (сгенерированных для проведенных расчетов по экспоненциальному закону распределения) интенсивности и длительности заданий, с одинаковым средним значением, но различным соотношением между короткими и длинными заданиями. Приведены результаты расчетов метрик работы вычислительного кластера на реальных и синтетических данных, которые показали, что эффективность стратегий планирования можно повысить на основе дополнительной информации о распределении длительности входных заданий.

1. Введение

Одной из важнейших задач повышения производительности и эффективности работы распределенных вычислительных систем является оценка их текущей загрузки и выбор таких стратегий планирования входных потоков заданий, которые позволили бы в соответствии с выбранными критериями эффективности оптимизировать их работу [1–5]. В литературе решению этих вопросов уделяется значительное внимание. В связи с этим отметим и выделим следующие направления исследований. В работах [6–10], посвященных статистическому анализу находящихся в вычислительной системе заданий (workload), анализируются их следующие основные характеристики – закон распределения поступающих заданий (их интенсивности), закон распределения длительности заданий и времени между прибытиями заданий (interarival time). Основной задачей является определение условий, при которых вычислительная система обрабатывает потоки заданий таким образом, что при этом оптимизируются такие показатели ее функционирования, как время ответа системы, коэффициент использования (загрузки) ресурсов, время завершения последнего задания (makespan), а также обеспечивается определенный уровень качества обслуживания пользователей QoS (Quality of Service). В работе [6] для повышения эффективности работы вычислительной системы предложено использовать координированную по отношению к характеристикам поступающих заданий стратегию: для этого предложено использовать вариант агрессивного backfilling (Job Rank-backfilling policy), смысл которого заключается в том, что в первую очередь продвигаются те задания, которые имеют малую длительность (short или small jobs). Другой подход заключается в том, что с целью избегания «зависания» длинных заданий наряду с методом backfilling используется стратегия Shortest Job First (SJF) (Shortest Job Backfilled First, SJBF) [7, 8]. В работе [9] для исследования влияния стратегии работы с очередью был предложен «пороговый» подход: задание считается коротким, если ее длительность (рассчитанная или ожидаемая) меньше принятого порогового значения; работа считается длинной, если ее длительность (рассчитанная или ожидаемая) больше принятого порогового значения; при этом определяется процент тех заданий, длительность которых принимается по расчету как длинные, а они являются фактически короткими, и наоборот. Таким образом, данный подход позволяет определить целесообразность использования стратегий SJF или LJF вместе с алгоритмом backfilling, используемом для выполнения заданий на ресурсе [10].

Анализ приведенных методов координированных стратегий показывает, что координация осуществляется на основе 2-х основных подходов: анализа плотности распределения оценки (предварительно рассчитываемой) длительности поступающих на ресурс заданий и времени прибытия (поступления) заданий (стохастический вариант) или с использованием некоторых пороговых значений (детерминированный ва-

риант). В обоих случаях имеется неточность: в первом случае – в силу стохастичной природы оценивания имеются короткие и длинные задания, определяемые математическим ожиданием; во втором случае – неточность определяется самим подходом: пороговое значение, а, следовательно, и оценки значений для коротких и длинных заданий имеют погрешность. Это может привести к изменению (или повлиять на выбор) стратегий планирования заданий.

Необходимо отметить, что приведенные показатели функционирования отображают различные подходы для оценки эффективности работы распределенной системы: коэффициент использования является системным (системно ориентированным) показателем, характеризует уровень загрузки ресурсов и является ключевым для их владельцев; временные критерии отражают стратегии, ориентированные на пользователей (как одиночных, так и в составе виртуальных организаций). Это приводит к тому, что одновременное выполнение критериев работы указанных двух участников виртуальных организаций представляется затруднительным и, вместе с тем, определяет направленность исследований для нахождения компромиссного решения на основе реальных данных функционирования распределенных систем и определения границ, в которых такой компромисс может быть достигнут.

В работе [11] приведены результаты исследований для оценки степени влияния стратегий работы с входным потоком заданий на производительность вычислительных систем, для определения которого (и основных факторов) использовались такие методы планирования заданий как Shortest Job First (SJF), Longest Job First (LJF) и наиболее распространенный в промышленных системах метод FCFS. Данные о выполняемых заданиях разбивались на периоды (например, сутки, недели) и моделировались ситуации, в которых выбираемые задания сортировались по убыванию и по возрастанию их длительностей. После этого запускался механизм работы с очередью FCFS и задания обрабатывались в соответствии с методами пакета GridSim и интегрированным в программное обеспечение симулятора планировщика, основанного на методе наименьшего покрытия [5]. Необходимо отметить, что при проведении эксперимента не учитывались значения параметров плотности распределения такого ***показателя*** как математическое ожидание длительности заданий, которое, как уже отмечалось, может повлиять на выбор стратегии работы с входной очередью. Аналогичные вопросы рассматривались при исследовании эффективности пакетных режимов, в которых преобладание коротких или длинных заданий непосредственно влияет на выбор метода планирования заданий [11].

Целью настоящего исследования является определение влияния характеристик входных заданий на выбор стратегий планирования на основе статистического анализа длительностей и интенсивностей реальных заданий и синтетических заданий с заданными статистическими характеристиками для распределенной вычислительной системы.

2. Статистический анализ входных потоков заданий

Для статистического анализа входного потока заданий использованы данные о выполненных заданиях на кластере ИСМА НАН Украины за период 2012–2013 гг.

Обобщенный статистический анализ предусматривает:

статистический анализ интенсивности заданий по генеральной выборке – в целом, и по этой же выборке – по дням недели;

статистический анализ длительности заданий;

построение временного ряда для определения динамики интенсивности входных заданий за анализируемый период.

Статистический анализ характеристик входных заданий проводился на основе следующих этапов.

Этап 1. Статистический анализ количества заданий (плотность распределения количества) за весь период показал, что они распределены по экспоненциальному закону (рис. 1).

Этап 2. Статистический анализ количества поступающих заданий по дням недели – определения наиболее загруженных дней недели или дней, имеющих определенные закономерности поступления заданий на выполнение. Результаты определения плотности распределения приведены на рис. 2.

Полученные результаты (рис. 2) показывают сохранение экспоненциального закона распределения количества заданий и по дням недели, при этом наибольшее количество заданий выполняется в среду (в целом – в первой половине недели).

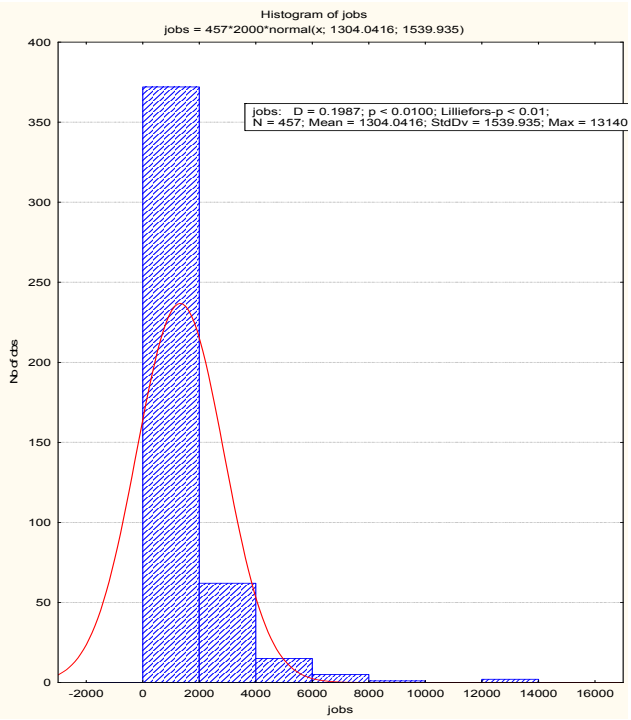


Рис. 1.Плотность распределения количества заданий за анализируемый период

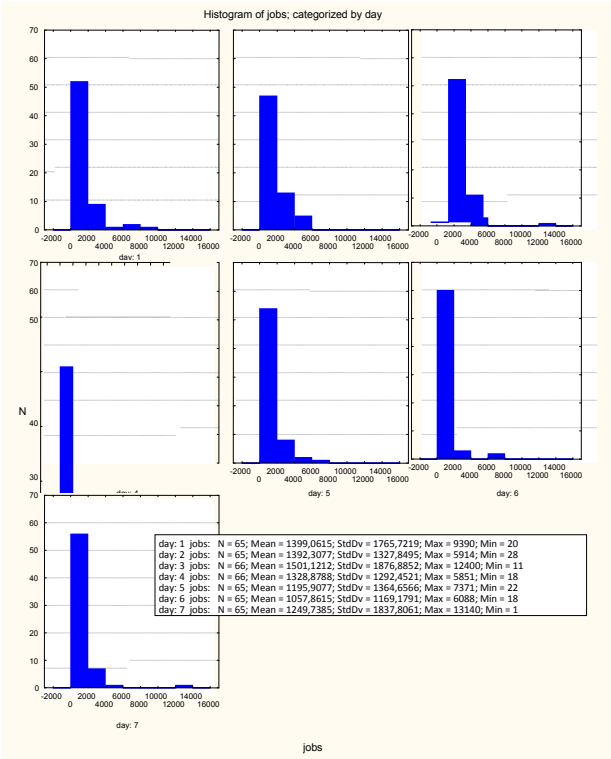


Рис. 2. Экспоненциальная плотность распределения количества поступающих заданий по дням недели

Таким образом, без потери общности анализа полученных результатов, в дальнейшем исследование проводилось для выборки заданий, поступивших за 1 неделю.

Этап 3. Анализ стационарности входных потоков заданий по дням.

Данный анализ проводился с целью определения стационарности потоков заданий (загрузки) кластера за весь анализируемый период (см. рис. 3). Из графика на рис. 3 следует, что имеется явный тренд увеличения количества заданий, поступающих на кластер на обработку, причем имеются изменения в политиках (стратегиях) использования ресурсов (240–250 дни исходной выборки заданий), при которых средняя загрузка на кластер увеличилась на 10%.

Последний результат свидетельствует о том, что интенсивность входного потока заданий возросла, что, в общем, может привести к изменению параметров функционирования кластера: в случае, если количество ресурсов не изменилось, коэффициент использования может уменьшиться; анализ динамики среднего времени ответа требует дополнительного исследования данных о выполненных заданиях.

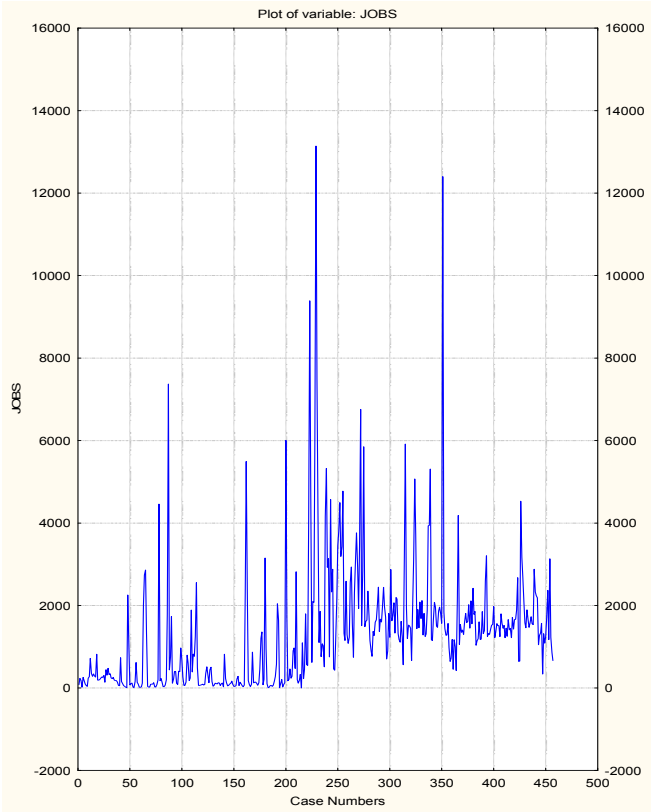


Рис. 3. Временной ряд количества поступивших заданий в течение 460 дней

Предварительный этап статистического анализа потоков заданий завершается определением закона распределения длительностей выполнения заданий (рис. 4).

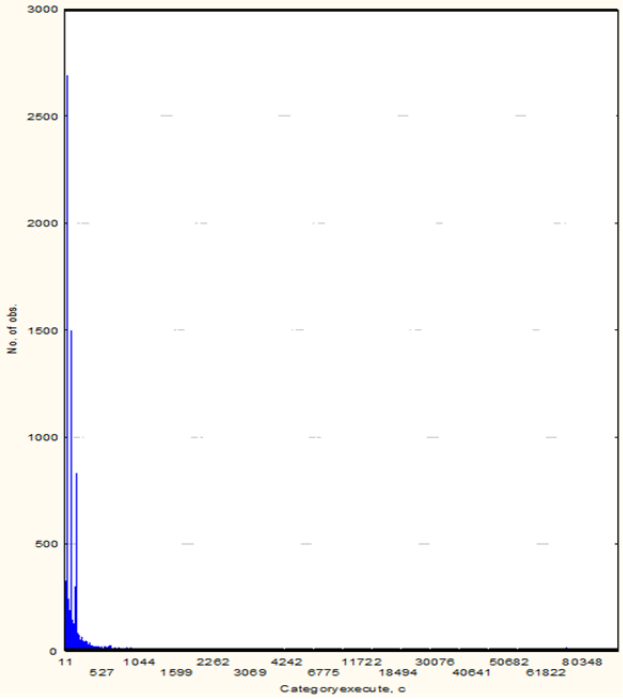


Рис. 4. Экспоненциальный закон распределения длительности заданий

Проведенный анализ показал, что длительности заданий подчиняются экспоненциальному закону распределения, среднее значение составляет 1308 сек. (22 мин.), т.е. большинство заданий – короткие.

Для исследования результатов применения стратегий планирования SJF (или LJF) в данной работе использованы синтетические тестовые данные для интенсивностей поступления заданий и их длительностей, сгенерированных по экспоненциальному закону распределения.

При генерации синтетических заданий приняты следующие допущения:

интенсивность и закон распределения сгенерированных входных заданий должен соответствовать интенсивности и закону распределения заданий на кластере ИСМА НАН Украины за анализируемый период (см. рис. 1, 2, 4);

для оценки влияния длительности заданий на стратегии планирования предполагается осуществить следующие генерации распределений синтетических заданий по длительности:

а) генерация длительностей заданий на основе среднего значения, но с меньшей долей коротких заданий в тестовой выборке, т.е. увеличение в выборке доли средних и длинных заданий на 10 %;

б) генерация длительности на основе среднего значения и закона распределения длительностей заданий кластера (рис. 4);

время завершения задания рассчитается как время его старта, увеличенное на его длительность.

Предварительный анализ реальных данных интенсивности поступления заданий на кластер показал, что среднее значение количества заданий по часам (группировка выполнена по часам суток для генеральной выборки) составляет 113 заданий (2 задания в мин.). Таким образом, при среднем времени решения заданий на узлах кластера 22 мин. в очереди на все узлы в среднем может находиться 44 задания.

Генерация синтетических длительностей заданий и их интенсивностей осуществлена в пакете STATISTICA 7.1 на основе экспоненциального закона распределения.

При этом экспоненциальное распределение имеет вид:

$$f(x)=\frac{1}{\lambda}\cdot e^{-\frac{x}{\lambda}}, \qquad x\in[0;\infty),\ \ \lambda>0,$$

где среднее значение ряда определяется как отношение x к λ (в нашем случае – 113).

В проведенных экспериментах для генерации интенсивности и интервалов времени поступления заданий на кластер были использованы следующие характеристики x=2, λ=0,017544, которые позволили получить среднее значение синтетического ряда равным 113.

Для генерации синтетических последовательностей поступления заданий в пакете STATISTICA 7.1 использована формула Vexpon(rnd(2); 0,017544). Статистический анализ интенсивности поступления заданий на кластер в различные интервалы времени показал, что она имеет экспоненциальный закон распределения (рис. 5, 6).

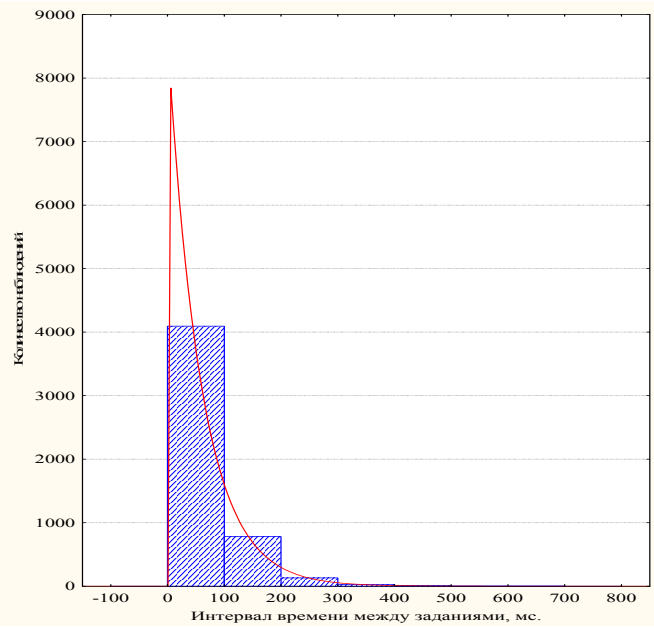


Рис. 5. Экспоненциальный закон распределения интенсивности поступления заданий в различные интервалы времени

Category	K-S d=,16159, p<,01 ; Lilliefors p<,01					
	Count	Cumulative Count	Percent of Valid	Cumul % of Valid	% of all Cases	Cumulative % of All
-100,000<x<=0,00000(	0	0	0,00000	0,0000	0,00000	0,0000
0,000000<x<=100,000	4094	4094	81,26241	81,2624	40,9400(	40,9400
100,0000<x<=200,000	781	4875	15,50218	96,7646	7,81000	48,7500
200,0000<x<=300,000	129	5004	2,56054	99,3251	1,29000	50,0400
300,0000<x<=400,000	26	5030	0,51608	99,8412	0,26000	50,3000
400,0000<x<=500,000	6	5036	0,11909	99,9603	0,06000	50,3600
500,0000<x<=600,000	1	5037	0,01985	99,9802	0,01000	50,3700
600,0000<x<=700,000	1	5038	0,01985	100,000(	0,01000	50,3800
Missing	4962	10000	98,49146		49,6200(	100,0000

Рис.6. Частотная таблица распределения интенсивности заданий в различные интервалы времени

Для генерации случайных последовательностей для длительностей заданий (времени выполнения) в пакете STATISTICA 7.1 со средним значением 1400 сек. были использованы следующие параметры: для варианта а) x=0,9, λ= 0,55; варианта б) – x=1, λ= 0,000714.

Вариант а) был сгенерирован при помощи функции 1000*Vexpon(rnd(0,9); 0,55) (см. рис. 7). Вариант б) был сгенерирован при помощи функции Vexpon(rnd(1); 0,000714) (см. рис. 8).

Варианты генерации синтетических данных отличаются тем, что вариант а) использует константу равную 1000 для увеличения доли средних и длинных заданий, а вариант б) использует генерацию стандартного экспоненциального распределения (без константы).

Для проверки отличия синтетических данных, сгенерированных в соответствии с вариантами а) и б), построены частотные таблицы, в которых отражены частоты поступления заданий в различные интервалы времени (см. рис. 9, 10). Таким образом, в отличие от работы [10], где генерируется одна генеральная тестовая выборка для получения двух различных тестовых последовательностей с одинаковыми характеристиками, в данном исследовании использован более обоснованный, со статистической точки зрения, подход к выбору стратегии планирования заданий входной очереди.

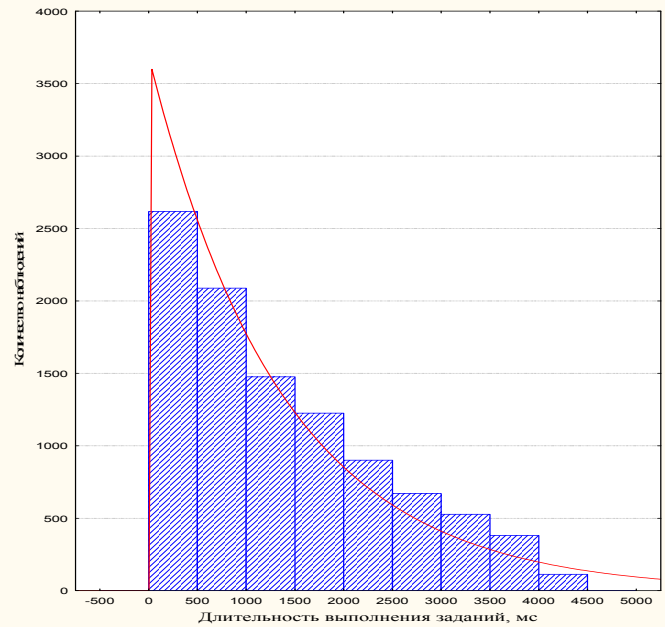


Рис. 7. вариант а) генерации длительностей

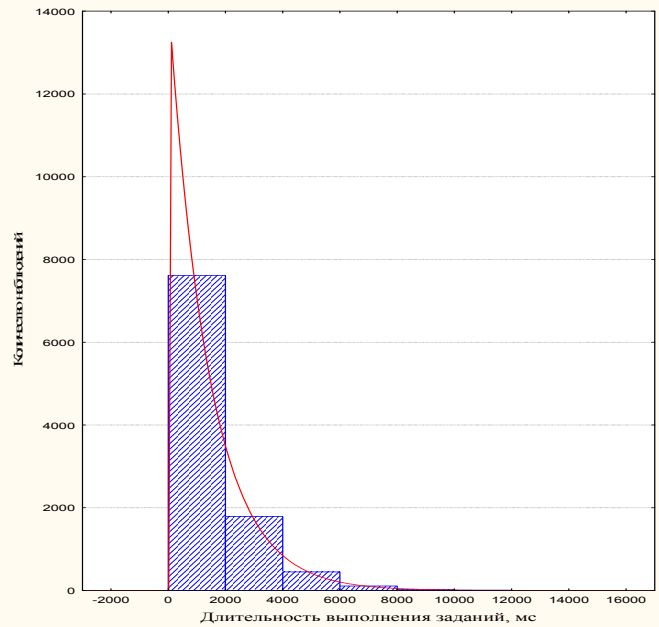


Рис. 8. Вариант б) генерации длительностей

Category	Frequency table: K-S d=,10338, p<,01 ; Lilliefors p<,01					
	Count	Cumulative Count	Percent of Valid	Cumul % of Valid	% of all Cases	Cumulative % of All
-1000,00<x<=0,00000(	0	0	0,00000	0,0000	0,00000	0,0000
0,000000<x<=1000,00	4705	4705	47,0500(	47,0500	47,0500(	47,0500
1000,000<x<=2000,00	2702	7407	27,0200(	74,0700	27,0200(	74,0700
2000,000<x<=3000,00	1572	8979	15,7200(	89,7900	15,7200(	89,7900
3000,000<x<=4000,00	908	9887	9,08000	98,8700	9,08000	98,8700
4000,000<x<=5000,00	113	10000	1,13000	100,000(	1,13000	100,0000
Missing	0	10000	0,00000		0,00000	100,0000

Рис.9. Частотная таблица распределения длительности выполнения заданий для варианта а)

Category	K-S d=,16062, p<,01 ; Lilliefors p<,01					
	Count	Cumulative Count	Percent of Valid	Cumul % of Valid	% of all Cases	Cumulative % of All
-2000,00<x<=0,00000(	0	0	0,00000	0,0000	0,00000	0,0000
0,000000<x<=2000,00	7615	7615	76,1500(	76,1500	76,1500(	76,1500
2000,000<x<=4000,00	1790	9405	17,9000(	94,0500	17,9000(	94,0500
4000,000<x<=6000,00	449	9854	4,49000	98,5400	4,49000	98,5400
6000,000<x<=8000,00	111	9965	1,11000	99,6500	1,11000	99,6500
8000,000<x<=10000,0	23	9988	0,23000	99,8800	0,23000	99,8800
10000,00<x<=12000,0	10	9998	0,10000	99,9800	0,10000	99,9800
12000,00<x<=14000,0	2	10000	0,02000	100,000(	0,02000	100,0000
Missing	0	10000	0,00000		0,00000	100,0000

Рис.10. Частотная таблица распределения длительности выполнения заданий для варианта б)

3. Сценарии проведения экспериментов и анализ полученных результатов

Для проведения экспериментов были использованы результаты предварительной статистической обработки данных о входных потоках заданий, приведенные в п. 2, и методы планирования заданий, имеющиеся в пакете GridSim – FCFS, Backfill и интегрированный в него и программно реализованный метод планирования, основанный на методе наименьшего покрытия МС [5]. Организация и проведение экспериментов осуществлялось следующим образом: для анализа была определена контрольная выборка заданий за весь период – 9200 заданий; для определения текущих показателей функционирования системы были определены периоды времени (интервалы планирования) таким образом, чтобы среднее количество поступающих заданий за эти периоды было равным 600.

Для оценки влияния исследуемых методов (стратегий) планирования заданий на функционирование кластера в соответствии с полученными п. 2 результатами реализованы следующие сценарии.

Сценарий 1. Планирование и расчет показателей функционирования системы для реальных данных.

Сценарий 2. Планирование и расчет показателей функционирования для сгенерированных синтетических данных для двух предложенных вариантов генерации длительностей заданий. В качестве метрик работы кластера выбраны следующие: коэффициент использования ресурсов, коэффициент балансировки загрузки, характеристика качества обслуживания, определяемая как отношение среднего времени ожидания заданий в очереди к среднему времени их выполнения $\frac{t_{wait}}{t_{run}}$. Для корректного сравнения полученных результатов для всех методов планирования выбран один и тот же период. Далее на графиках использованы следующие обозначения: КИ – коэффициент использования ресурса; КБ – коэффициент балансировки загрузки; количество ресурсов (КР) и количество процессорных элементов (КПЭ) на ресурсе обозначается как КР КПЭ. На рис. 11, 12 приведены результаты моделирования в пакете GridSim за выбранный период для сортированных по методу LJF заданий и без сортировки для 50 ресурсов, на рис.13, 14 – для 100 ресурсов.

Анализ приведенных на рис. 11–14 результатов показывает, что сортировка заданий по методу LJF практически не влияет на величину КИ как для 50, так и для 100 ресурсов. При этом наилучший результат показывает метод МС (30–33 % – для 50 ресурсов и 19–20% – для 100), затем из наиболее используемых следует backfill, наихудший результат – у метода FCFS.

На рис. 15, 16 приведены результаты моделирования и полученные значения КБ за выбранный период, которые показывают, что этот показатель также практически не изменяется, при этом наибольшее значение имеет метод МС (92 %), наименьшее – метод FCFS (для сортированных заданий 2%, для несортированных – 20%). Это указывает на то, что данный метод может иметь низкий коэффициент использования в случае, когда большинство заданий – длинные, или они отсортированы по убыванию их длительности.

На рис. 17, 18 приведены результаты расчетов для показателя качества обслуживания $\frac{t_{wait}}{t_{run}}$ для 50 и 100 ресурсов, соответственно. Из графиков следует, что, в целом, с увеличением количества ресурсов его величина уменьшается, а лучшие результаты – у методов МС и backfill.

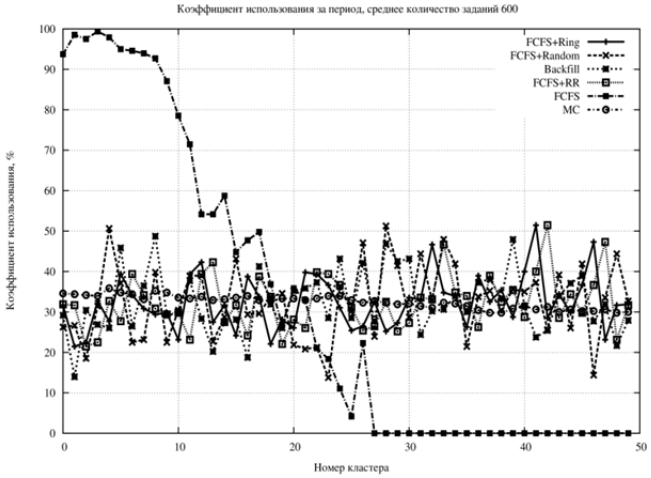


Рис. 11. Зависимость КИ для 50 ресурсов (с LJF)

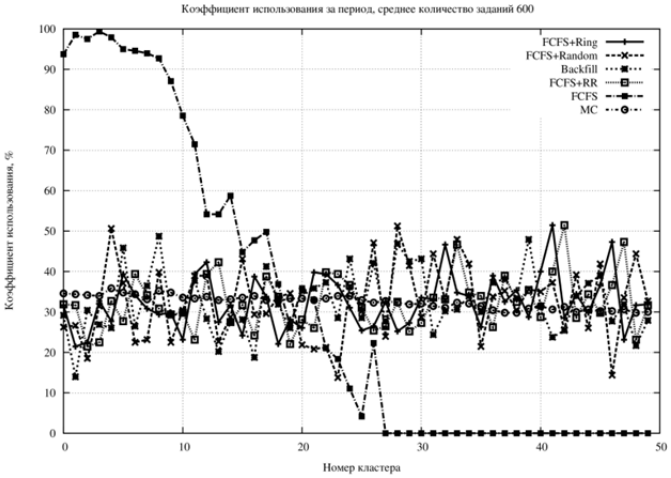


Рис. 12. Зависимость КИ для 50 ресурсов

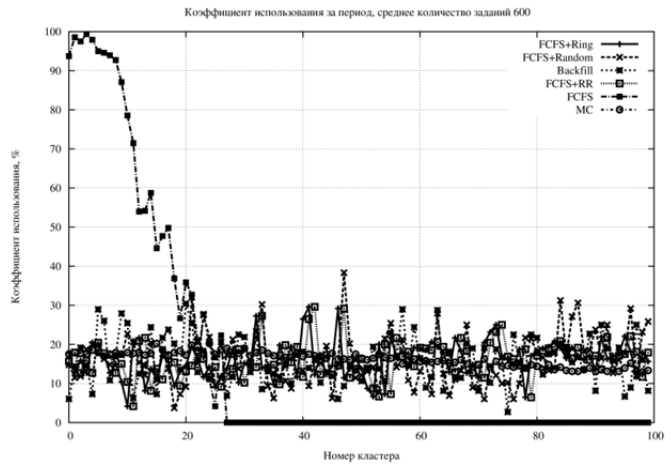


Рис. 13. Зависимость КИ 100 ресурсов (с LJF)

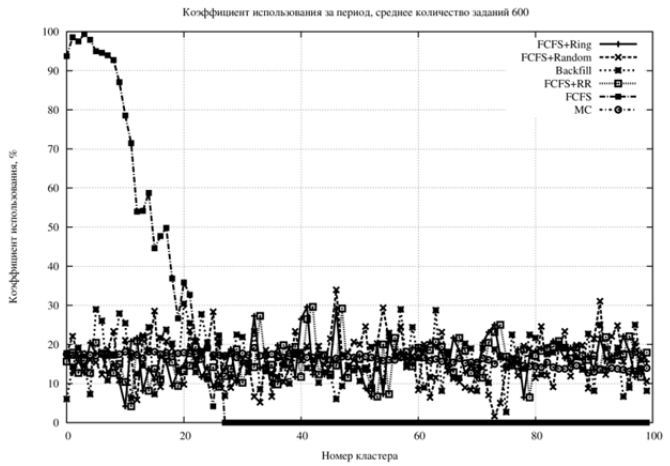


Рис. 14. Зависимость КИ для 100 ресурсов

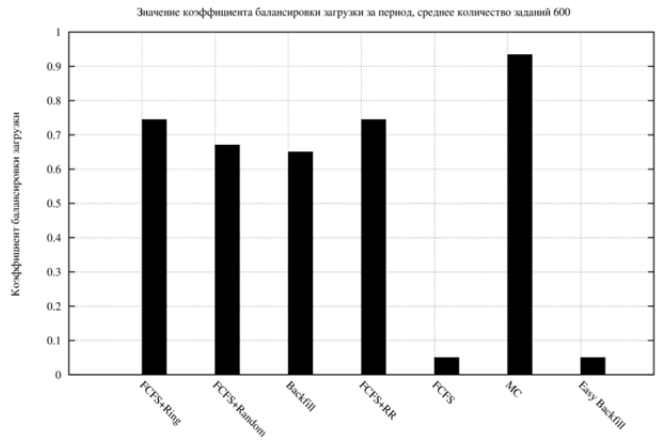


Рис. 15. Зависимость КБ для 70 ресурсов (с LJF)

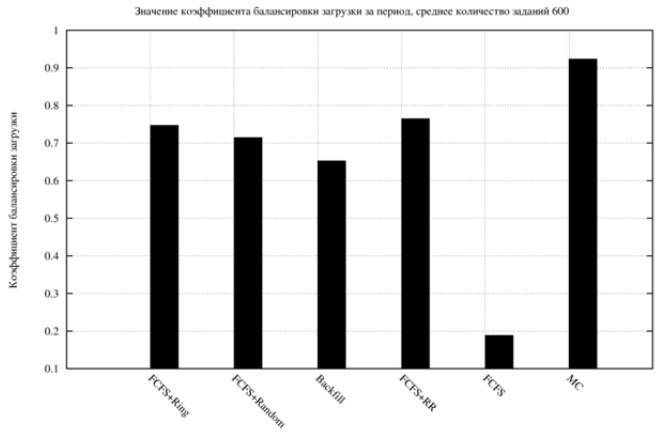


Рис. 16. Зависимость КБ для 70 ресурсов

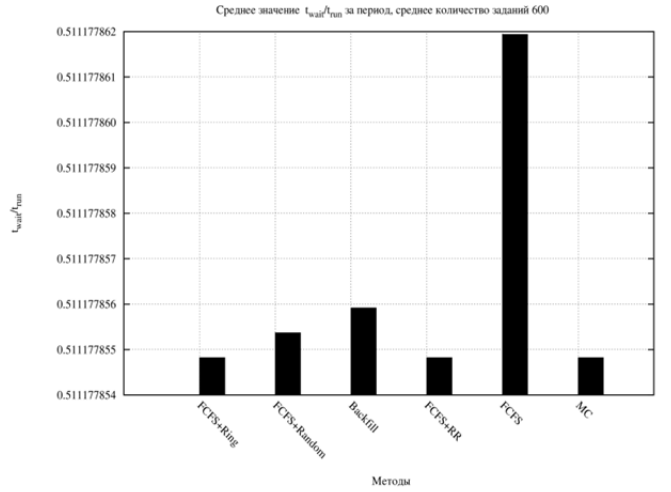


Рис. 16.Зависимость $\frac{t_{wait}}{t_{run}}$ для 50 ресурсов

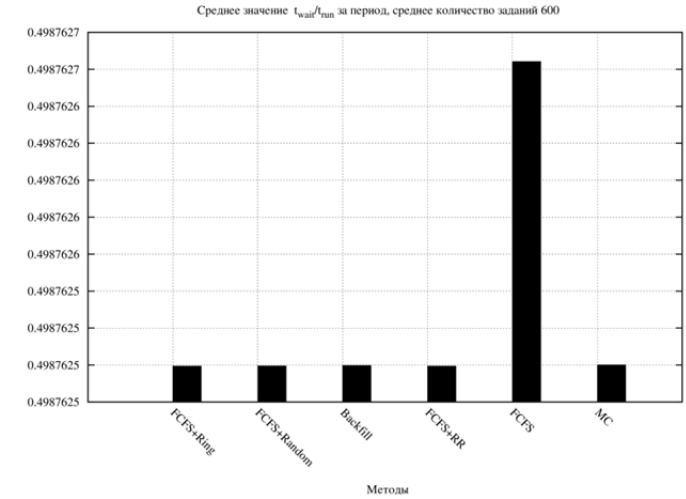


Рис. 17.Зависимость $\frac{t_{wait}}{t_{run}}$ для 100 ресурсов

Моделирование на сгенерированных синтетических данных проводилось в соответствии с описанными ранее сценариями.

Для обеспечения требуемого уровня достоверности полученных результатов сгенерированы 2 множества синтетических данных длительностей заданий – по 10 различных массивов с приведенными ранее параметрами экспоненциального закона распределения, каждый из которых включает 10000 наблюдений. Полученные в результате расчетов значения метрик работы системы усредняются по количеству сгенерированных синтетических длительностей. На рис. 18–23

приведены результаты расчетов для различных показателей для данных, сгенерированных по вариантам а) и б), соответственно, для 50 ресурсов. Как следует из графиков, коэффициент использования с увеличением доли длинных заданий в генеральной тестовой выборке увеличился для всех методов, при этом лучшие результаты показывают методы MC (улучшение составляет в среднем с 23% до 26%) и backfill (улучшение составляет в среднем с 22% до 25%); улучшение для метода FCFS составляет с 16% до 20%, но при этом ресурсы загружаются крайне неравномерно.

На рис. 24–29 приведены результаты моделирования для рассматриваемых вариантов для 100 ресурсов.

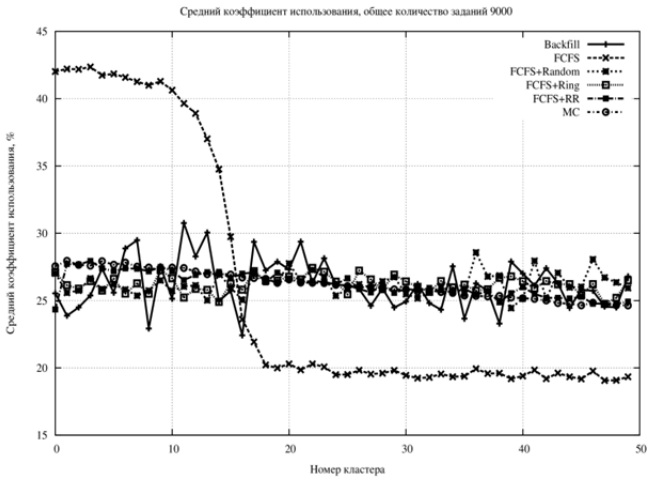


Рис. 18. Среднее значение КИ для варианта а)

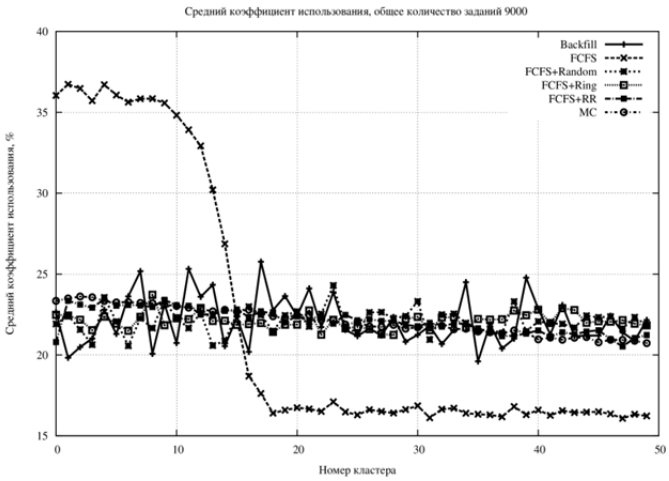


Рис. 19. Среднее значение КИ для варианта б)

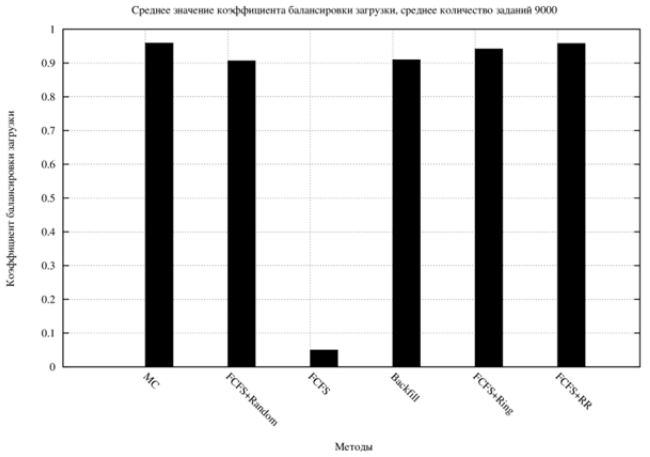


Рис. 20. Среднее значение КБ для варианта а)

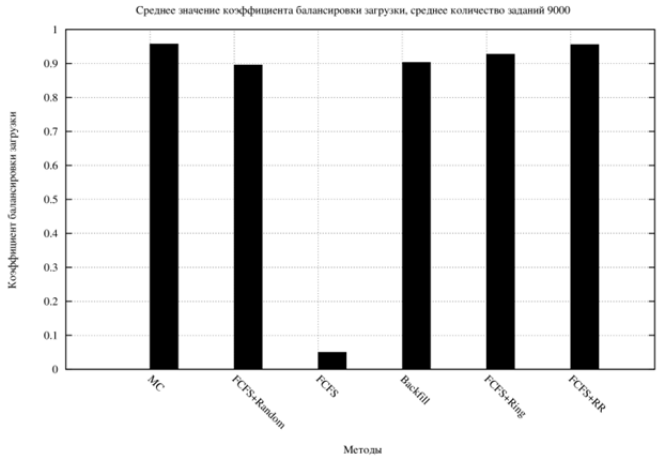


Рис. 21. Среднее значение КБ для варианта б)

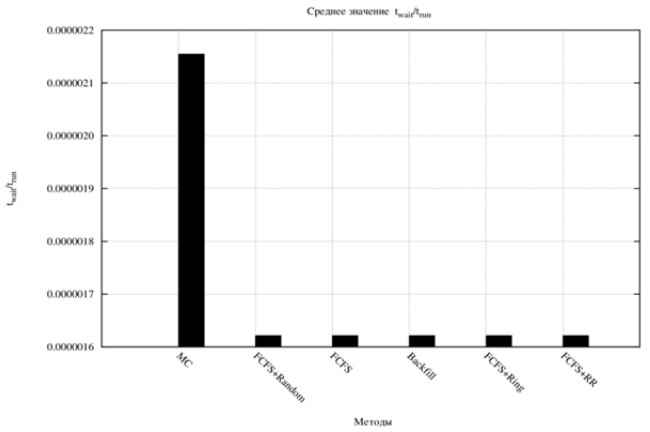


Рис. 22. Среднее значение $\frac{t_{wait}}{t_{run}}$ для варианта а)

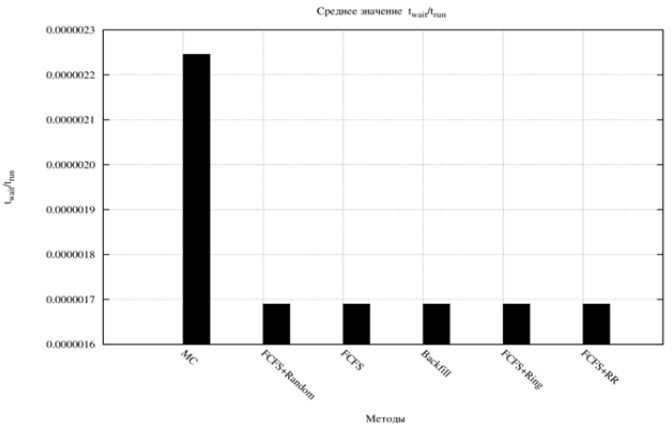


Рис. 23. Среднее значение $\frac{t_{wait}}{t_{run}}$ для варианта б)

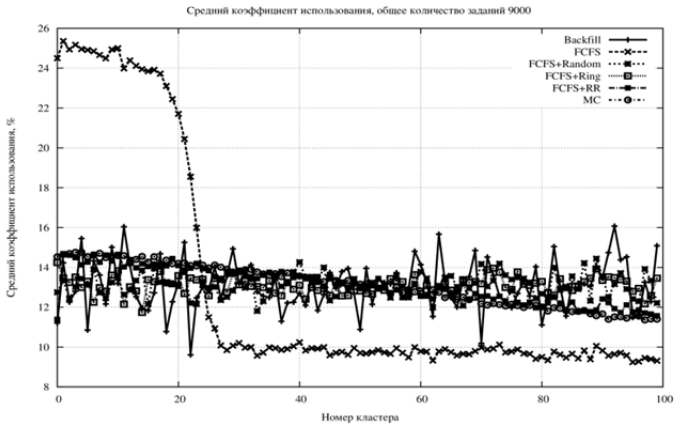


Рис. 24.Среднее значение КИ для варианта а)

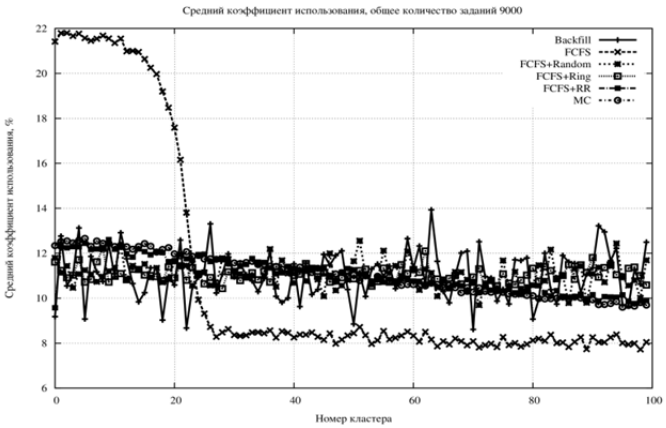


Рис. 25. Среднее значение КИ для варианта б)

Приведенные на рис. 24, 25 результаты показывают, что для 100 ресурсов величина КИ для всех методов уменьшилась практически в 2 раза, при этом улучшение для отдельных методов снизилось до уровня: для MC – с 12% до 13%, для backfill – с 11% до 13%, для FCFS – 8% до 10%. Приведенные на рис. 26, 27 результаты показывают, что среднее значение КБ для различных вариантов не изменилось, а среднее значение $\frac{t_{wait}}{t_{run}}$ несколько увеличилось, что предсказуемо для случая длинных заданий и постоянных интервалов планирования.

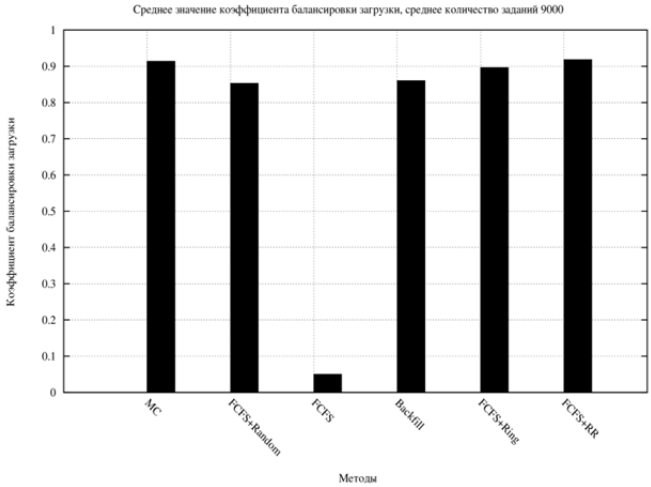


Рис.26. Среднее значение КБ для варианта а)

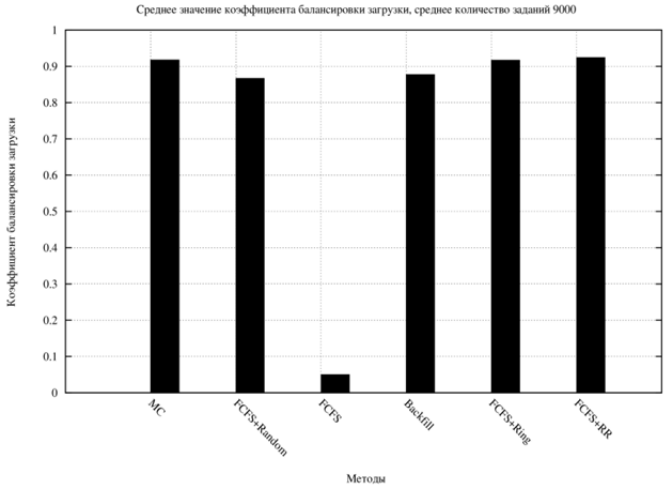


Рис.27. Среднее значение КБ для варианта б)

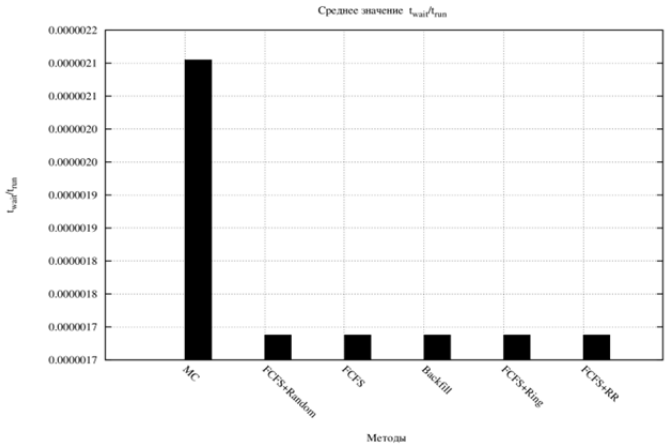


Рис. 28. Среднее значение $\frac{t_{wait}}{t_{run}}$ для варианта а)

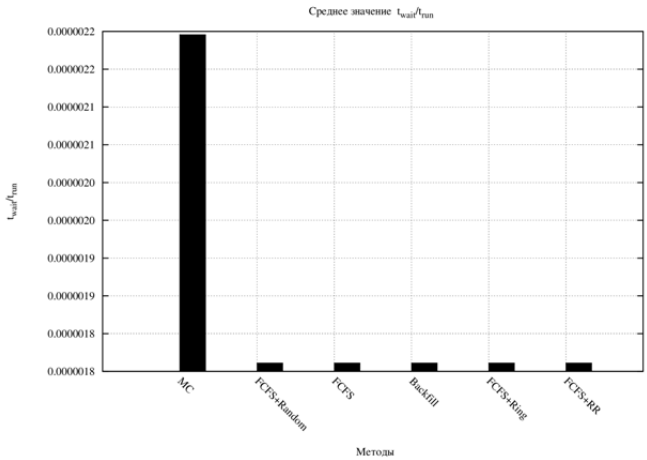


Рис. 29. Среднее значение $\frac{t_{wait}}{t_{run}}$ для варианта б)

4. Выводы

В данном исследовании получены результаты, позволяющие сформулировать следующие выводы относительно оценки влияния характеристик поступающих на вычислительный кластер заданий на выбор стратегии планирования и производительность системы.

1. Для обоснованного выбора координированной стратегии планирования заданий необходимо проводить статистический анализ интенсивности поступающих заданий, их длительности, а также динамики интенсивности заданий, показывающей, как при этом может измениться производительность системы в целом.

2. Для выбора стратегии планирования заданий ключевым является, как показали проведенные исследования, соотношение между количеством длинных и коротких заданий, влияющего на оценки метрики производительности вычислительной системы. Для подтверждения этой гипотезы в работе предложено использовать генерацию синтетических (искусственных) данных об интенсивностях и длительностях заданий, имеющих одинаковые средние значения, соответствующие характеристикам реальных данных. При этом используется частотный анализ, в соответствии с которым можно достаточно точно определить долю длинных заданий (их количество) в генерируемых случайных тестовых последовательностях.

3. Проведенные на синтетических данных исследования показали, что сортировка для массива с увеличенным количеством длинных заданий в соответствии со стратегией SJF практически не влияет на значения используемых метрик. Это, по-видимому, объясняется тем, что для определенных периодов (в исследовании они выбраны в среднем равным 600 заданиям), соотношение меду длинными и короткими заданиями можно рассчитать, а для каких-то – нет, в силу неравномерного поступления заданий по периодам. Поэтому необходимо использовать динамическое планирование: например, с увеличением интенсивности заданий или увеличением доли коротких заданий уменьшать интервал планирования или определять его по среднему времени решения заданий на ресурсе данной вычислительной системы.

4. Качество обслуживания с увеличением доли длинных заданий падает (отношение $\frac{t_{wait}}{t_{run}}$ увеличивается), поэтому для обоснования правильности выбора координированной стратегии планирования необходимо определять приоритеты для критериев эффективности работы вычислительной системы.

Список литературы

1. Симоненко В.П. Математическая постановка задачи динамического распределения работ в grid системах и оценки качества решения. // Вісник НТУУ «КПІ» Інформатика, управління та обчислювальна техніка. – 2011. – №53. – С.37–41.
2. Симоненко В.П. Теоретические основы проектирования динамических пространственных планировщиков неоднородных GRID систем. // Электронное моделирование. – 2011. – Т. 33. – №5. – С. 57–71.
3. Симоненко А.В. Система пространственного распределения заданий в распределенных вычислительных системах./ А.В. Симоненко, С.В. Пих, Н.В. Слуцкий, В.В. Воробйов. // Вісник НТУУ «КПІ» Інформатика, управління та обчислювальна техніка. – 2012. – №56. – С.94–97.
4. Петренко А.І. Гібридний алгоритм брокера для Nordugrid ARC 2.0. /А.І Петренко, С.Я. Свистунов, П.В. Свірін.// Міжнародна конференція "Високопродуктивні обчислення" HPC-UA'2012 (Україна, Київ, 8-10 жовтня 2012 року. – С. 275–277.
5. Листровой С.В. Модель и подход к планированию распределения ресурсов в гетерогенных ГРИД-системах / С.В. Листровой, С.В. Минухин. // Проблемы управления и информатики. – 2012. – №5. – С.120–133.
6. Rodero I. Evaluation of Coordinated Grid Scheduling Strategies / I. Rodero, F. Guim, J. Corbalan // HPCC '09 Proceedings of the 2009 11th IEEE International Conference on High Performance Computing and Communications, 2009, p. 1–10.
7. Tsafrir D. Backfilling Using Runtime Predictions Rather Than User Estimates / D.Tsafrir, Y. Etsion, D. G. Feitelson. // IEEE Transactions on Parallel and Distributed Systems. – 2007. – V.18. – Iss. 6. – p.789–803.
8. Srinivasan S. Characterization of Backfilling Strategies for Parallel Job Scheduling / S. Srinivasan, R. Kettimuthu, V. Subramani, P. Sadayappan.// Proceedings of the ICPP Workshop on Scheduling and Resource Management for Cluster Computing, 2002.

9. Guim F. Prediction *f* based Models for Evaluating Backfilling Scheduling Policies / F. Guim1, J. Corbalan, J. Labarta. // PDCAT '07 Proceedings of the Eighth International Conference on Parallel and Distributed Computing, Applications and Technologies, 2007, p. 9–17.
10. Guim F.A Job Self-Scheduling Policy for HPC Infrastructures / F. Guim, J. Corbalan. // JSSPP'07 Proceedings of the 13th international conference on Job scheduling strategies for parallel processing, 2007, p. 51–75.
11. Минухин С.В. Статистический анализ и исследование влияния методов работы с очередью входных заданий на функционирование распределенных вычислительных систем. / С.В. Минухин, А.В. Коровин. // Third International Conference "High Performance Computing"HPC-UA 2013 (Ukraine, Kyiv, October 7-11, 2013), 2013, С. 258–265.

УДК 004.7

ЛУЦКИЙ Г.М.,
МУХИН В.Е.

МЕТОДЫ МАРШРУТИЗАЦИИ В КОМПЬЮТЕРНЫХ СИСТЕМАХ С МНОГОКАНАЛЬНЫМИ СВЯЗЯМИ

В статье рассмотрены вопросы адаптации классических механизмов и методов маршрутизации, ориентированных на одноканальные системы связи, к многоканальной технологии передачи данных. Выполнено исследование параметров механизмов маршрутизации при их применении для многоканальных систем связи, выделены наиболее эффективные механизмы в зависимости от конкретных условий их реализации.

In this article there is discussed the adaptation of the classic routing mechanisms and methods oriented on the single-channel communication system for the multi-channel data transmission technology. There is performed the analisys of the routing mechanisms parameters for the multichannel communication systems, and are distinguished the most effective mechanisms depending on the specific conditions of their implementation.

1. Введение

Важным аспектом при эксплуатации распределенных компьютерных систем является повышение их пропускной способности по передаче данных. Одним из подходов к повышению пропускной способности данных систем является реализация каналов связи на основе технологии многоканальности. В частности, для поддержки данной технологии применяются оптоволокон-ные линии, что позволяет реализовать частотное и временное разделение каналов связи на подканалы, и тем самым обеспечить одновременную передачу значительных массивов данных. Таким образом важной задачей является адаптация классических механизмов и методов маршрутизации, ориентированных на одноканальные системы связи, к многоканальной технологии передачи данных.

2. Метод коммутации каналов при одноканальных и многоканальных связях

Основные идеи методов маршрутизации базируются на классических механизмах с их адаптацией для коммутируемых сетей, при этом задержки передачи данных вычисляются для L_M - битового сообщения. Размеры флита и фита будем считать эквивалентными и равными ширине канала передачи данных в L_{TC} бит, длина которого в свою очередь равна ширине внутреннего канала передачи данных в маршрутизаторах. Заголовок пакета для маршрутизации равен 1 флит, тем самым размер сообщения оказывается равен: $L_M + L_{TC}$ бит. Маршрутизатор принимает решение по маршрутизации за один цикл, и, следовательно, флит проходит через маршрутизатор или канал связи также за один цикл.

В методе коммутации каналов связи для передачи данных резервируется физический путь от интерфейса источника к интерфейсу приемника, после чего источник запускает т.н. называемый пробный пакет. Пробный пакет направляется к приемнику в соответствии с протоколом маршрутизации, и в процессе продвижения резервирует физические каналы и порты маршрутизатора на выделенном пути. Для подтверждения факта выделения пути исходному узлу-источнику возвращается подтверждение о доставке [1].

Недостатки данного метода заключаются в том, что выделенные каналы связи и порты маршрутизаторов на некоторый интервал времени блокируют другие потоки данных, которым требуются те же ресурсы, как и для выделенного канала связи, в частности, пробный пакет может быть заблокирован в процессе ожидании освобождения канала связи. Каналы связи, ранее занятые пробным пакетом аналогично блокируют создание других каналов [1].

Параметр времени передачи данных при коммутации каналов определяется как сумма времени создания путей (маршрутов) и времени передачи данных. Для канала, который проходит к приемникам через маршрутизаторы время передачи данных рассчитываются как:

$$t_{канал} = m * [t_c + 2(t_b + t_r + t_{lc}) + t_{данные}] \quad (1)$$

Здесь: t_c - время, затрачиваемое на определение выходного порта маршрутизатора для передачи сообщения, t_b - время входного буферирования, t_r - время передачи бит через маршрутизатор, t_{lc} - время передачи бит через канал

связи, соответственно, пробным пакетом и пакетом подтверждения, m – число узлов, на которые выполняется передача данных.

Процесс определения выходного порта взаимно совмещается с процессом запроса и выделения порта маршрутизатора и поэтому время формирования пути маршутизации не учитывается в выражении (1). Коэффициент 2 в выражении (1) отражает тот факт, что пробный пакет и пакет подтверждения передаются в обе стороны по одному и тому маршруту. Заметим, что пакет подтверждения не маршрутизируется, и, следовательно, для него отсутствует задержка маршрутизации на промежуточных маршрутизаторах.

Коммутация каналов, как правило, более эффективна в том случае, когда сообщения передаются с низкой интенсивностью и являются относительно длинными по сравнению с размером пробных пакетов. После первичного формирования пути следования пакетов, последующие сообщения в этом направлении передаются без задержек на формирование пути, при этом также повышается межузловая пропускная способность среды передачи данных, поскольку после создания пути следования сообщения передаются к приемнику не маршрутизируются и не блокируются в сети.

Вариант реализации коммутации каналов для многоканальных связей приведен на рис. 1. В этом случае единый канал связи подразделяется на N независимых подканалов, например, на основе частотного принципа разделения, и по каждому из них данные передаются независимо друг от друга (на определенной частоте).

Параметр времени передачи данных при коммутации каналов при многоканальных связях определяется как:

$$t_{канал} = t_c + 2(t_b + t_r + t_{lc}) + t_{данные}, \text{ при } m \le n \quad (2)$$

$$t_{канал} = \frac{m}{n}[t_c + 2(t_b + t_r + t_{lc}) + t_{данные}], \text{при } m > n \quad (3)$$

где m – число узлов, на которые выполняется одновременная передача данных, n – число одновременно открытых подканалов связи.

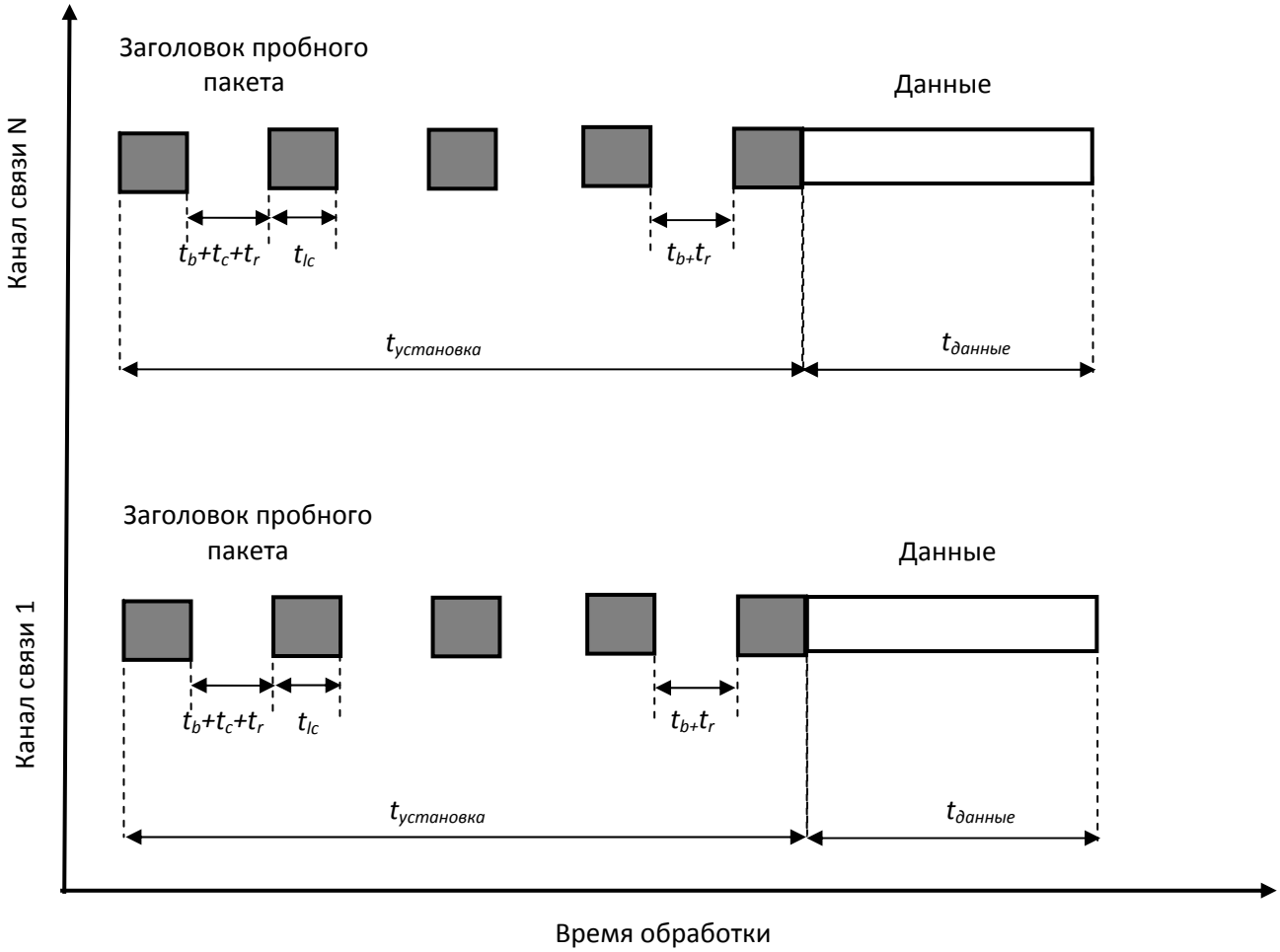


Рис.1. Метод коммутации каналов связи для многоканальных связей

Дополнительно снизить время передачи данных возможно путем применения технологии конвейерной передачи данных в одном подканале, на основе временного разделения трафика, что позволит фактически повысить степень заполнения подканала трафиком и снизит периоды простоя подканала (рис. 2).

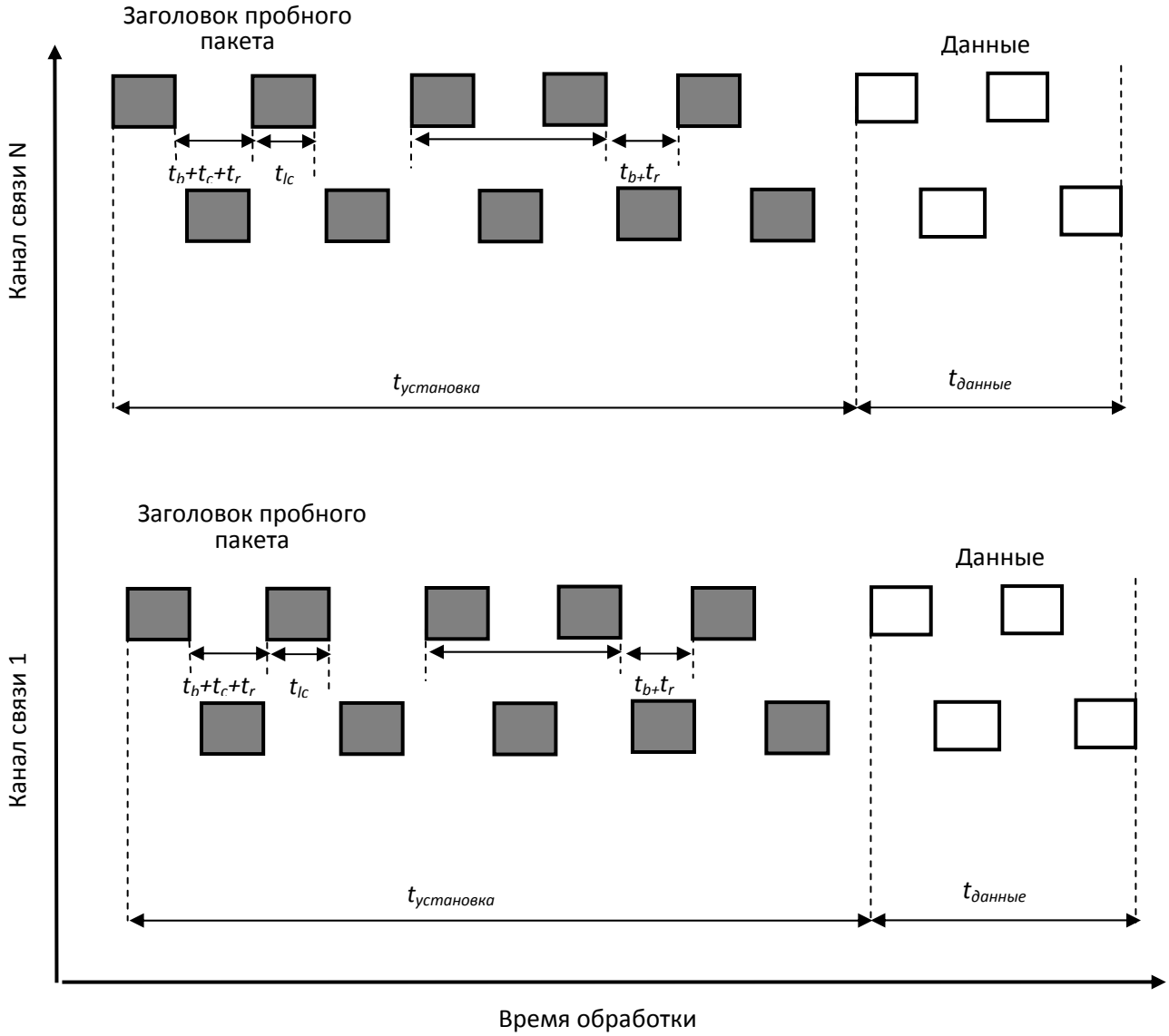


Рис.2. Метод коммутации каналов связи для многоканальных связей с конвейеризацией

Параметр времени передачи данных при коммутации каналов при многоканальных связях и при реализации конвейерной технологии передачи данных в одном подканале (рис. 2) определяется как:

$$t_{канал2} = t_c + 2(t_b + t_r + t_{lc}) + t_{lc} + 2t_{данные} \text{ при } m \le n \quad (4)$$

$$t_{канал2} = \frac{m}{n}[t_c + 2(t_b + t_r + t_{lc}) + t_{lc} + 2t_{данные}] \text{ при } m > n \quad (5)$$

На рис. 3 показаны графические зависимости времени передачи данных (в временных тактах *T*) по каналу связи от источника к приемнику в зависимости от числа *m* узлов-приемников для трех случаев: коммутация каналов при одноканальных связях, при многоканальных связях и при многоканальных связях с применением конвейерной передачи данных в одном подканале, на основе временного разде-ления трафика. При построении графических зависимостей принято, что: t_c = 1 временной такт (Т), t_b = 1 Т, t_r = 1 Т, t_{lc} =1 Т, t_d =5 Т и результат представлен в временных тактах.

Как видно из рис. 3 наиболее эффективной оказывается маршрутизация при многоканальных связях с использованием конвейерной передачи данных, которая обеспечивает снижение времени передачи одинаковых массивов данных для данного случая в среднем на 500% по сравнению маршрутизацией при одноканальных связях и на 33% без конвейе-ризации.

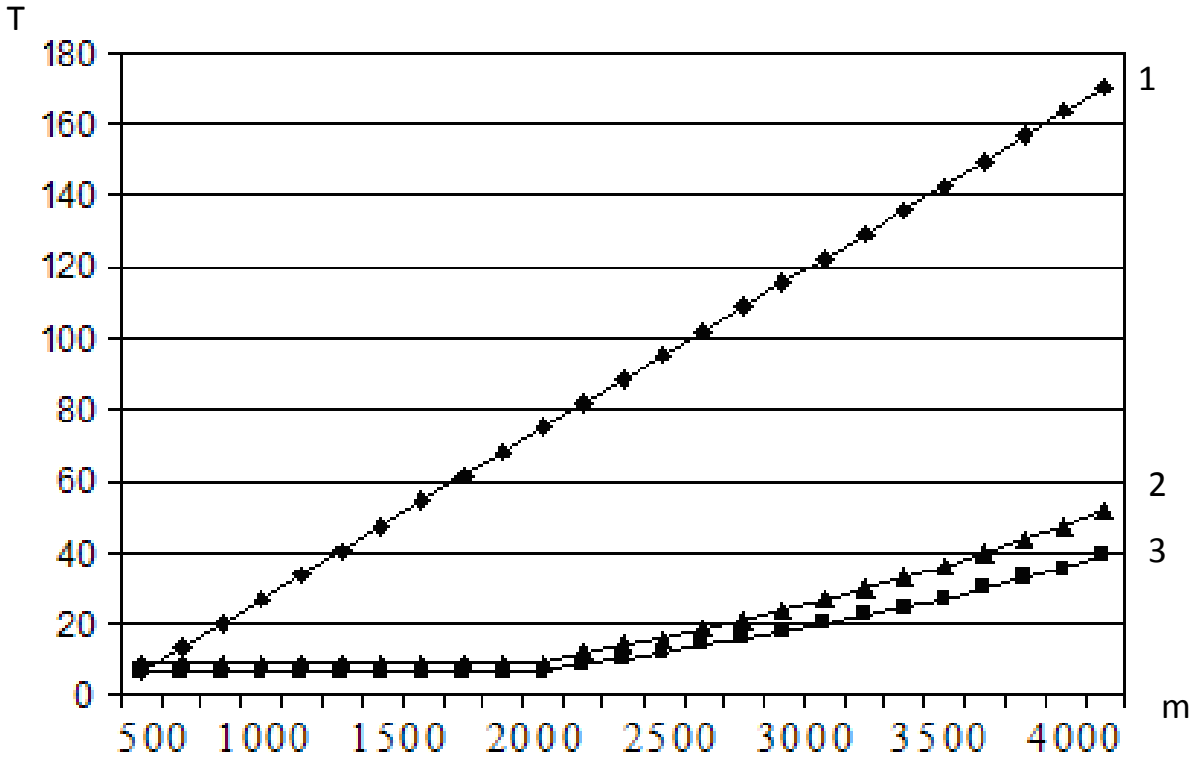


Рис. 3. Время передачи данных в зависимости от числа узлов-приемников для маршрутизации по методу коммутации каналов при одноканальных связях (1), при многоканальных связях (2) и при многоканальных связях с применением конвейерной передачи данных в одном подканале (3)

3. Метод коммутации пакетов при одноканальных и многоканальных связях

В методе коммутации каналов процессы маршрутизации и передачи данных четко разделены. Все маршрутизаторы на пути следования пакета от источника к приемнику определяются заранее и все передающиеся в пакетах пользовательские данные полностью занимают полосу пропускания канала. В методе коммутации пакетов сообщения разделяются на блоки одинаковой длины. Первые несколько байт формируют заголовок пакета, который содержит информацию по маршрутизации и управляющую информацию, а пользовате-льские данные находятся в теле пакета. Каждый пакет передается по сети отдельно, а управ-ление потоком данных между маршрути-заторами выполняется на уровне полных пакетов [2].

Пакет не может быть переслан в том случае, если буферное пространство всего пакета не доступно на следующем маршрутизаторе, в противоположном случае, полный пакет передается по каналу связи во входной буфер следующего маршрутизатора. Время передачи данных через маршрутизаторы в этом случае оценивается как:

$$t_{пакетная} = m*[t_c + (t_b + t_r + t_{lc})*(L_M + L_{TC})/L_{TC}] \quad (6)$$

если $L_M = L_{TC}$, то:

$$t_{пакетная} = m*[t_c + 2*(t_b + t_r + t_{lc})] \quad (7)$$

Вариант реализации коммутации каналов для многоканальных связей приведен на рис. 4. В этом случае единый канал связи подразделяется на N независимых подканалов, например, на основе частотного принципа разделения, и по каждому из них данные передаются независимо друг от друга (на определенной частоте).

Параметр времени передачи данных при коммутации каналов при многоканальных связях определяется как:

$$t_{пакетная} = t_c + 2*(t_b + t_r + t_{lc}) \quad \text{при } m \le n \quad (8)$$

$$t_{пакетная} = \frac{m}{n}[t_c + 2*(t_b + t_r + t_{lc})] \quad \text{при } m > n \quad (9)$$

где m – число узлов, на которые выполняется одновременная передача данных, n – число одновременно открытых подканалов связи.

Дополнительно снизить время передачи данных возможно путем применения технологии конвейерной передачи данных в одном подканале, на основе временного разделения трафи-

ка, что позволит фактически повысить степень заполнения подканала трафиком и снизит периоды простоя подканала (рис. 5).

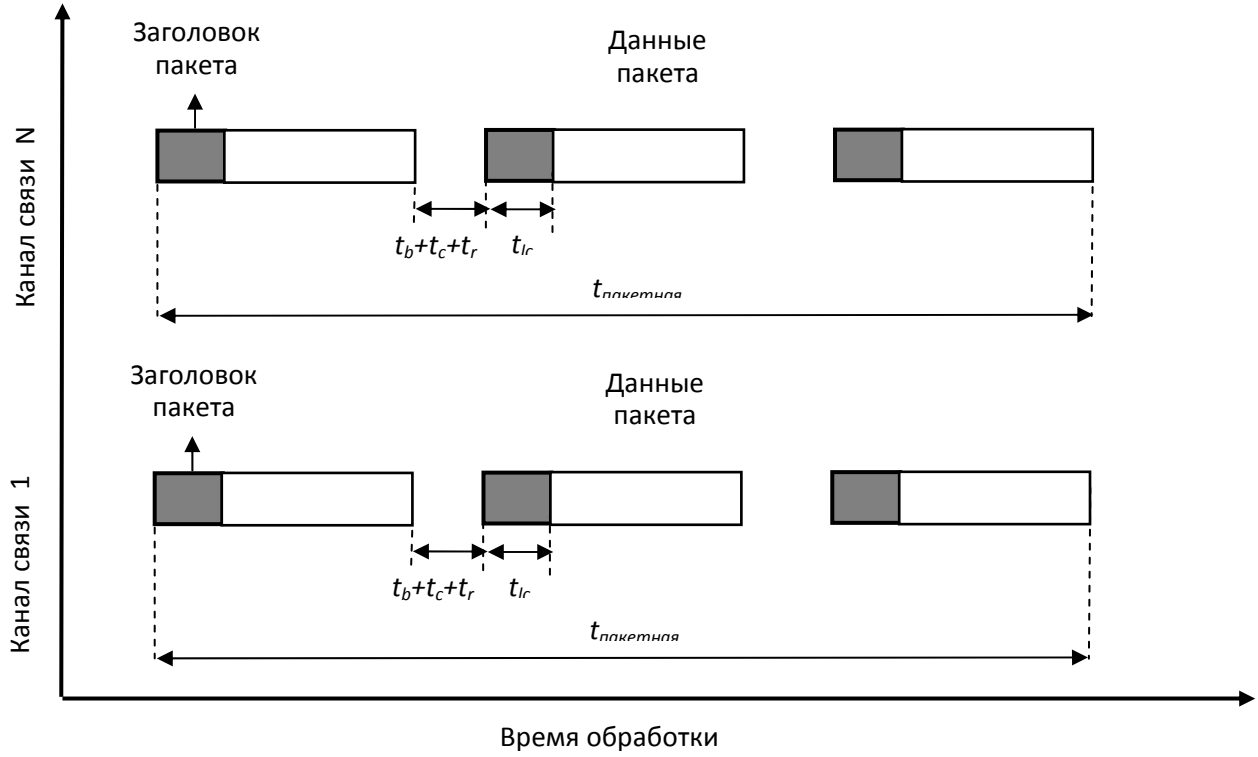


Рис.4. Метод коммутации пакетов для многоканальных связей

Параметр времени передачи данных при коммутации пакетов при многоканальных связях и при реализации конвейерной техно-логии передачи данных в одном подканале (рис. 5) определяется как:

$$t_{пакетная3} = t_c + 2*(t_b + t_r + t_{lc}) + 2t_{lc} \text{ при } m \le n \quad (10)$$

$$t_{пакетная\,3} = \frac{m}{n}[t_c + 2*(t_b + t_r + t_{lc}) + 2t_{lc}] \text{ при } m > n \quad (11)$$

На рис. 6 показаны графические зависимости времени передачи данных по каналу связи от источника к приемнику в зависимости от числу узлов-приемников для трех случаев: коммутация пакетов при одноканальных связях, при многоканальных связях и при многоканальных связях с применением конвейерной передачи данных в одном подканале, на основе временного разделения трафика.

При построении графических зависимостей принято, что: $t_c = 1$ временной такт (Т), $t_b = 1$ Т, $t_r = 1$ Т, $t_{lc} = 1$ Т и результат в временных тактах.

Как видно из рис. 6 наиболее эффективной оказывается маршрутизация при многоканальных связях с использованием конвейерной передачи данных, которая обеспечивает снижение времени передачи одинаковых массивов данных в среднем в 5 раз по сравнению маршрутизацией при одноканальных связях в 1.7 раза для случая без конвейеризации.

4. Метод червячного механизма маршрутизации при одноканальных и многоканальных связях

Необходимость полной буферизации пакетов в маршрутизаторе создает определенные проблемы при построении компактных и быстродействующих маршрутизаторов. В ранних версиях в коммутируемых сетях для буферизации блокированных пакетов в качестве хранилища использовалась память локальных узлов. Процесс повторного запуска пакетов существенно повышает величину задержки передачи пакетов, следовательно, целесообразно сохранять пакеты в сети.

Однако оказалось недостаточно буферировать пакеты внутри лишь отдельных маршрутизаторов. Червячная маршрутизация представляет собой вариант оптимизации виртуальной конвейерной маршрутизации, с использованием буферов [3]. Размер буфера в каждом маршрутизаторе позволяет хранить несколько флит. В том случае, если заголовок пакета блокирован, сообщение занимает буферы в нескольких маршрутизаторах [3].

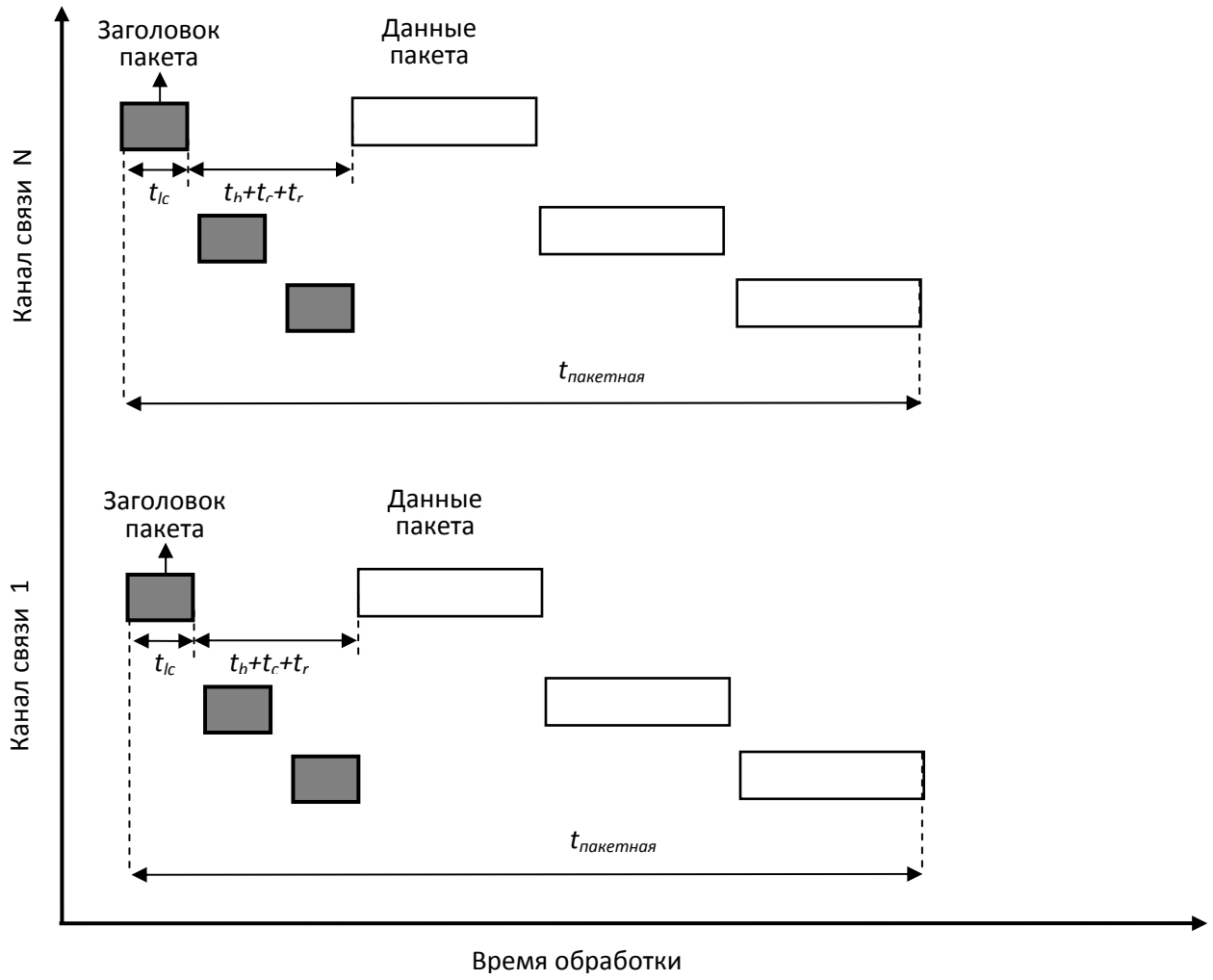


Рис. 5. Метод коммутации пакетов для многоканальных связей с конвейеризацией

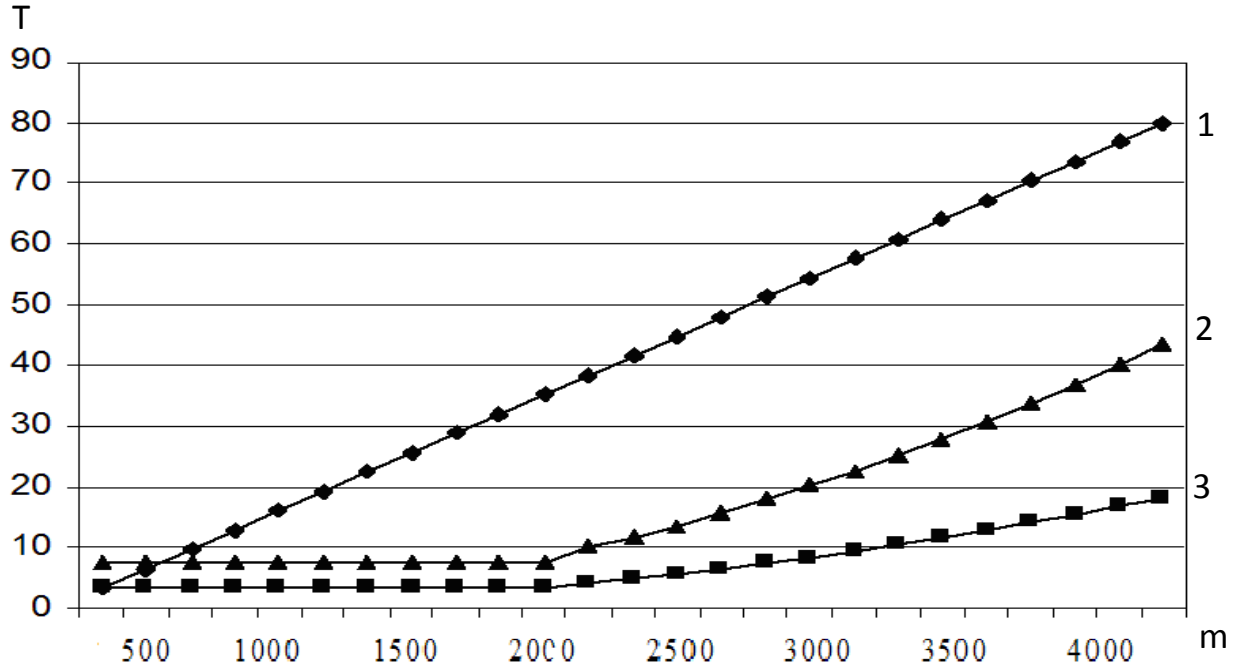


Рис. 6. Время передачи данных в зависимости от числа узлов-приемников для маршрутизации по методу коммутации пакетов при одноканальных связях (1), многоканальных связях (2) и при многоканальных связях с применением конвейерной передачи данных в одном подканале (3)

Использование буферов малого размера также приводит к двум важным результатам. Во-первых, снижаются задержки доступа к буферу и сокращается время процесса конвейеризации. Снижение задержек конвейе-ризации позволяет повысить тактовую частоту работы маршрутизатора, и, следовательно, повысить пропускную способность маршрутизатора, например, до уровня 5 ГГц. [4]. Во-вторых, малые буферы снижают стати-ческое энергопотребление, что особенно важно для встроенных маршрутизаторов. Однако, при увеличении загрузки канала связи, блокированное сообщение занимает буферы на нескольких маршрутизаторах и каналы связи между ними, что приводит к повторному блокированию сообщений, разделяющих эти каналы связи, что, в свою очередь, распространяет блокировки далее по сети.

Базовое время передачи сообщений, передаваемых по методу червячной маршрутизации через маршрутизатор с размером флита, равного размеру фита вычисляется следующим образом:

$$t_{червячная} = m*[(t_c + t_b + t_r + t_{lc}) + t_b*(L_M)/L_{TC}] \quad (11)$$

если $L_M = L_{TC}$, то:

$$t_{червячная} = m*(t_c + t_b + t_r + t_{lc}) + t_b \quad (12)$$

Недостатком метода червячной маршрутизации, реализуемого классическими маршрутизаторами является то, что заблокированные сообщения захватывают физические ресурсы канала связи. Информация по маршрутизации -- несколько флит в заголовке пакета, а флиты с данными пользователя не содержат маршрутной информации. Следовательно, когда пакет заблокирован, передачи других пакетов по физическому каналу связи не выполняются. В результате при увеличении интенсивности трафика возникает быстрое насыщение каналов связи заблокированными пакетами.

Вариант реализации червячной коммутации для многоканальных связей приведен на рис. 7. В этом случае единый канал связи подразделяется на N независимых подканалов, например, на основе частотного принципа разделения, и по каждому из них данные передаются независимо друг от друга (на определенной частоте).

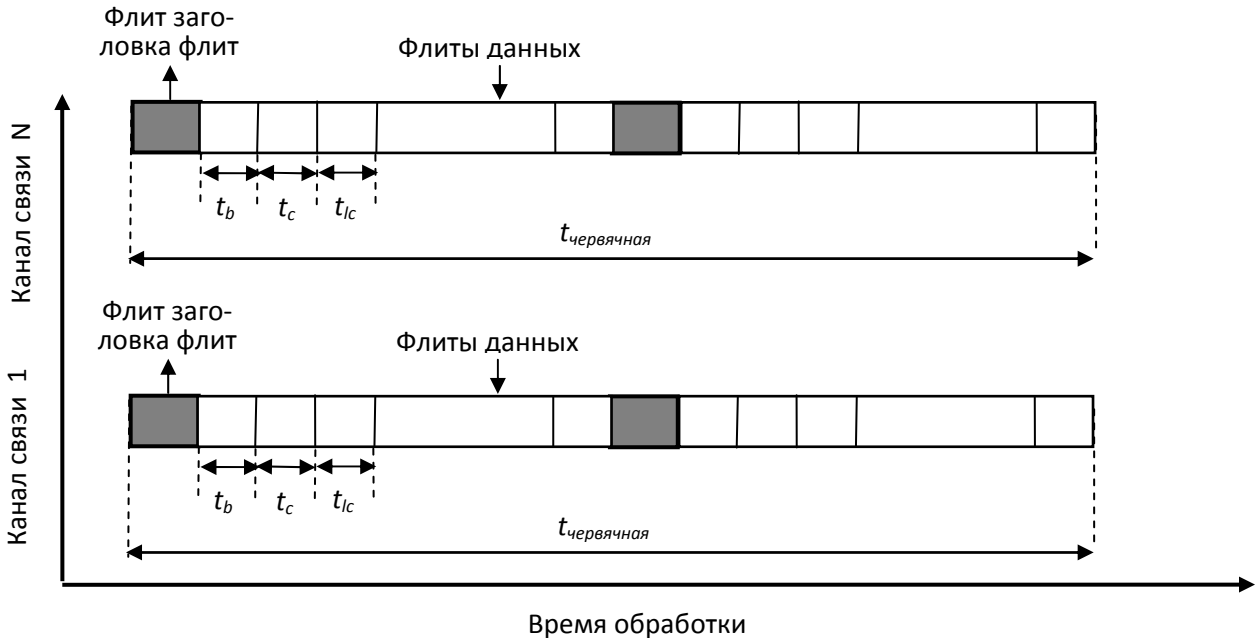


Рис.7. Метод червячного механизма маршрутизации для многоканальных связей

Параметр времени передачи данных при червячной коммутации при многоканальных связях определяется как:

$$t_{червячная} = (t_c + t_b + t_r + t_{lc}) + t_b \quad \text{при } m \le n \quad (13)$$

$$t_{червячная} = \frac{m}{n}*[(t_c + t_b + t_r + t_{lc}) + t_b] \quad \text{при } m > n \quad (14)$$

где m – число узлов, на которые выполняется одновременная передача данных, n – число одновременно открытых подканалов связи.

Дополнительно снизить время передачи данных возможно путем применения технологии конвейерной передачи данных в одном подканале, на основе временного разделения трафика, что позволит фактически повысить степень заполнения подканала трафиком и снизит периоды простоя подканала (рис. 8).

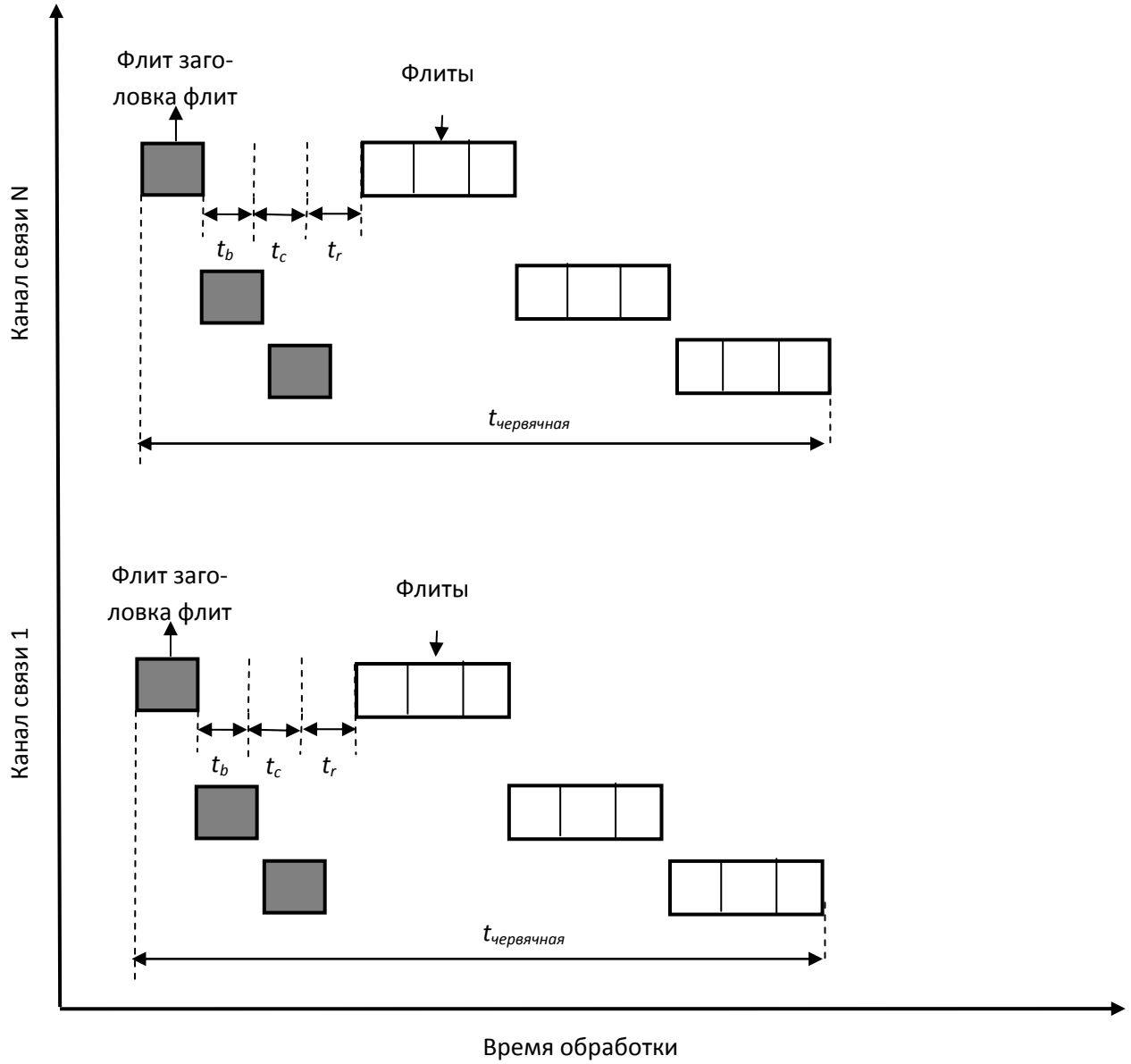


Рис.8. Метод червячного механизма маршрутизации для многоканальных связей с конвейеризацией

Параметр времени передачи данных при червячной коммутации при многоканальных связях и при реализации конвейерной технологии передачи данных в одном подканале (рис. 8) определяется как:

$$t_{червячная3} = (t_c + t_b + t_r + t_{lc}) + t_b + 2t_{lc} \quad \text{при } m \le n \quad (15)$$

$$t_{червячная3} = \frac{m}{n}[(t_c + t_b + t_r + t_{lc}) + t_b + 2t_{lc}] \quad \text{при } m > n \quad (16)$$

На рис. 9 показаны графические зависимости времени передачи данных по каналу связи от источника к приемнику в зависимости от числу узлов-приемников для трех случаев: червячная коммутация при одноканальных связях, при многоканальных связях и при многоканальных связях с применением конвейерной передачи данных в одном подканале, на основе временного разделения трафика.

При построении графических зависимостей принято, что: $t_c = 1$ временной такт (Т), $t_b = 1$ Т, $t_r = 1$ Т, $t_{lc} = 1$ Т и результат представлен в временных тактах.

Как видно из рис. 9 наиболее эффективной оказывается червячная маршрутизация при многоканальных связях с использованием конвейерной передачи данных, которая обеспечивает снижение времени передачи одинаковых массивов данных в 5 раз по сравнению маршрутизацией при одноканаль-ных связях и в 1.5 раза для случая без конвейеризации.

На рис.10 и рис.11 представлены сравнительные зависимости времени передачи данных в зависимости от числа узлов для различных

видов методов маршрутизации: коммутации каналов, пакетов, червячной маршрутизации при многоканальных связях при реализации конвейерной передачи данных

На рис.10 рассмотрен случай, когда передаются малые массивы данных 10 байт, а на рис. 11 – 100 байт.

Как видно из рис. 10 и рис. 11 при передаче данных малого размера (10 байт) наиболее эффективной оказывается червячная маршрутизация, а для передаваемых данных большей длины (100 байт) – маршрутизация на основе коммутации каналов.

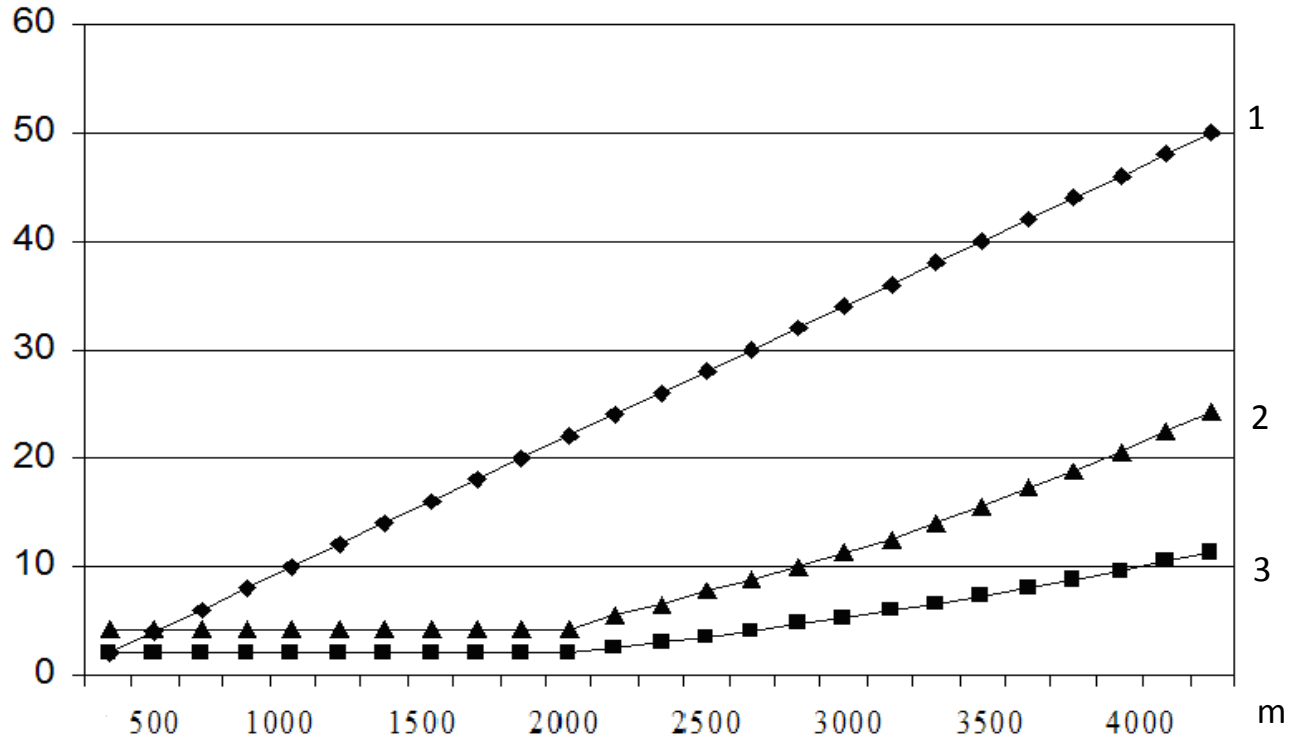


Рис. 9. Время передачи данных в зависимости от числа узлов-приемников для маршрутизации по методу червячной коммутации при одноканальных связях (1), многоканальных связях (2) и при многоканальных связях с применением конвейерной передачи данных в одном подканале (3)

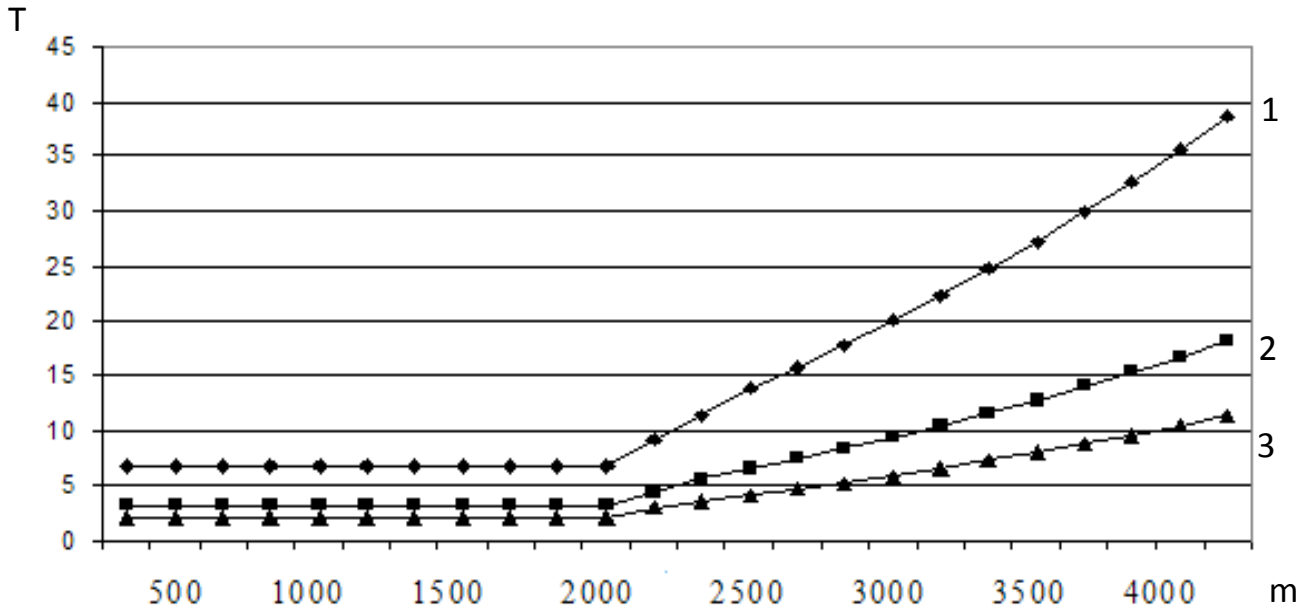


Рис.10. Время передачи данных в зависимости от числа узлов для коммутации каналов (1), пакетов (2), червячной маршрутизации (3) при многоканальных связях при реализации конвейерной передачи данных длиной 10 байт

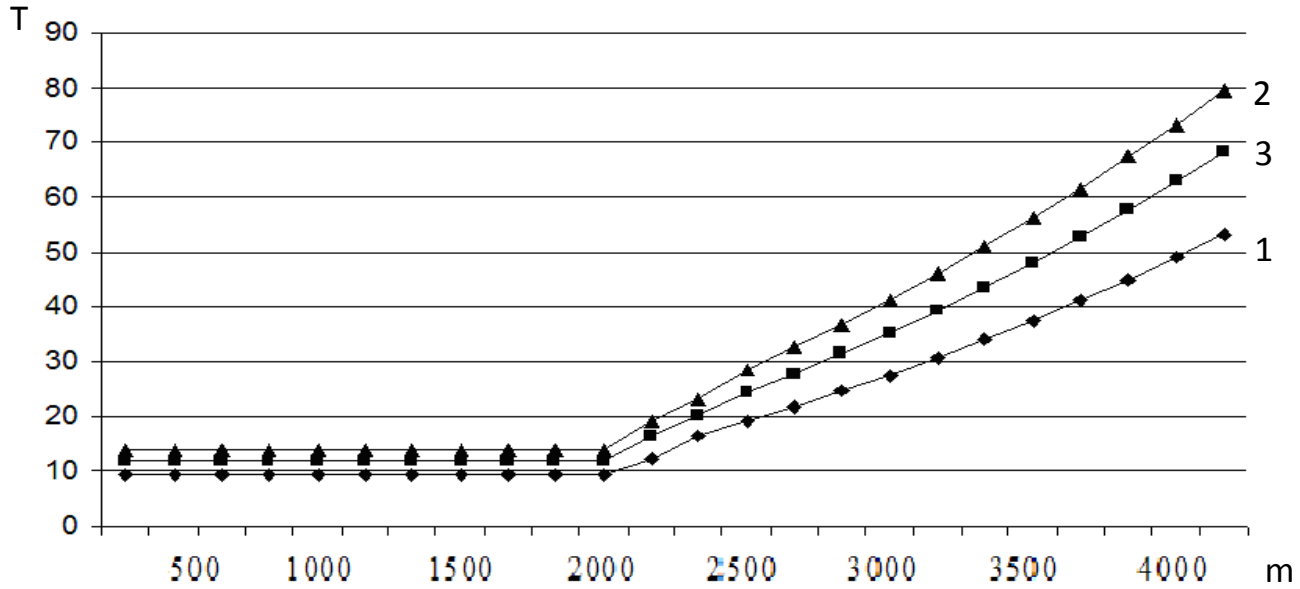


Рис. 11. Время передачи данных в зависимости от числа узлов для коммутации каналов (1), пакетов (2), червячной маршрутизации (3) при многоканальных связях при реализации конвейерной передачи данных длиной 100 байт

5. Заключение

Исследование классических механизмов и методов маршрутизации показали, что применение технологии многоканальных связей, например, на основе оптоволоконных связей позволяет в несколько раз (на 500 - 600% для рассматриваемых случаев) снизить время передачи эквивалентных объемов данных в условиях масштабирования компьютерных систем. Для повышения эффективности использования многоканальных связей целесообразно применить технологию конвейеризации при передаче как служебных, так и пользовательских данных.

Также исследования показали, то при передаче данных малого размера (порядка 10 байт) наиболее эффективной оказывается червячная маршрутизация, а для передаваемых данных большей длины (свыше 100 байт) – маршрутизация на основе коммутации каналов

Список литературы

1. Пепелкин Д.А., Перепелкин А.И. Разработка алгоритма динамической маршрутизации на базе протокола OSPF в корпоративных вычислительных сетях // Вестник Рязанского государственного радиотехнического университета. N2 (вып. 28), 2009. - с. 68- 72.
2. Doverspike N., Jha V. Comparison of Routing Methods for DCS-Switched Networks.// Interfaces, Vol. 23, No. 2 (Mar. - Apr., 1993), pp. 21-34
3. Mohapatra P. Wormhole Routing Techniques in Multicomputer Systems//spirit.cs.ucdavis.edu/pubs/journal/wsurvey.pdf
4. Washington N., Perros H. Performance Analysis of Traffic-Groomed Optical Networks Employing Alternate Routing Techniques// Lecture Notes in Computer Science, Vol. 4516, 2007, - pp. 1048-1059

УДК 004.451.24

СТІРЕНКО С.Г.

МОДЕЛЬ ТЕХНОЛОГІЇ ПРОГРАМУВАННЯ ПАРАЛЕЛЬНИХ СИСТЕМ

Рассматриваются существующие модели организации вычислений в параллельных системах, применяемые в различных инструментальных средствах. Предложены подходы к оценке єффективности параллельных систем с локальной памятью. Приведен математический формализм и семантика операций, программная реализация которых позволит соответствовать математической модели.

Different models used in modern libraries and frameworks to organize computations in parallel systems are studied and reviewed. An improved approach, which allows consideration of data locality in distributed system. The mathematical formalism of this model and semantics of operations on the model are described.

Порівняльний аналіз технологій програмування паралельних систем

На сьогоднішній день існує велика кількість технологій та моделей програмування паралельних обчислювальних систем, що підтримуються більшістю сучасного апаратного забезпечення таких систем. Переважна кількість моделей програмування, що застосовуються в вищеперелічених технологіях заснована на особливостях організації паралельних обчислень в тому чи іншому апаратному забезпеченні. Зокрема виділяють два великих класи обчислювальних систем: обчислювальні системи зі спільною пам'яттю, та обчислювальні системи з локальною пам'яттю, — кожна з яких має свої особливості організації роботи з даними при паралельних обчисленнях. Так для систем зі спільною пам'яттю найбільш часто застосовується модель, заснована на спільних даних, які можуть бути розташовані в спільній пам'яті обчислювальної системи та доступні всім учасникам паралельних обчислень. При цьому виникає ряд задач, які необхідно розв'язати для забезпечення коректності доступу до даних в пам'яті, зокрема задачі взаємного виключення та синхронізації. Перша з них необхідна щоб запобігти накладення одночасних записів в одну комірку пам'яті, яке може призвести, наприклад, до некоректного результуючого значення, тобто такого, яке не було записано жодним з одночасно виконуваних потоків. В той час як друга дозволяє організовувати очікування певної події в паралельно виконуваному потоці. З іншого боку, для обчислювальних систем з локальною пам'яттю типовою є модель взаємодії, заснована на передачі повідомлень, при використанні якої необхідно розв'язувати проблему підтримання непротирічності копій даних, що знаходяться в локальній пам'яті різних обчислювальних вузлів, а також проблему синхронізації.

Особливості організації доступу до даних зведені до табл. 1. Можна відмітити, що більшість розглянутих широко застосовуваних у високопродуктивних обчисленнях технологій програмування орієнтовані лише на обчислювальні системи зі спільною пам'яттю або лише на обчислювальні системи з локальною пам'яттю. Також слід зазначити, що ряд технологій додатково підтримують використання так званих акселераторів для виконання довільних обчислень.

Розглянемо паралельне виконання обчислень в гетерогенній, тобто неоднорідній, обчислювальній системі з локальною пам'яттю, що містить K вузлів та N процесорів. Один вузол може містити від 1 до $(N - K + 1)$ процесорів. Гетерогенність системи може полягати в різній продуктивності обчислювальних вузлів або у різній пропускній здатності мережевих зв'язків між ними, при чому мережа організована таким чином, щоб забезпечити можливість зв'язку між будь-якою парою вузлів.

Виконання паралельних обчислень в системі з локальною пам'яттю передбачає пересилання даних, які необхідні для виконання певної частини обчислень, з локальної пам'яті одного вузла в локальну пам'ять іншого, на якому будуть виконуватися обчислення. Позначимо M_i обсяг пам'яті, що використано в i-тому вузлі, де $i \in$ – номер вузла. Позначимо m_k розмір k-го блоку даних. Позначимо $t^i_{rdy,k}$ час готовності k-го блоку даних в і-тому вузлі, необхідних для обчислення в певному іншому j-тому вузлі, такому що $i \neq j$ Причому $t^i_{rdy,k} = 0 \ \forall i \in \overline{(1,K)}$ для вхідних даних, які не обчислюються в процесі роботи (вважаємо, що вхідні дані перенесені засобами операційної системи на всі вузли, в яких вони необхідні для обчислень) Позначимо $t^j_{req,k}$ час запиту k-го блоку даних для виконання обчислень з його використанням в j-тому вузлі. Відмітимо, що для у загальному випадку, обсяг

пам'яті, що буде використано, може залежати від значень вхідних даних та від результатів обчислень на попередніх кроках, тому його необхідно враховувати динамічно, тобто в процесі виконання обчислень. [1,2] Тоді необхідно визначити моменти часу для початку передачі даних $t_{comm,k}^{i\to j}$, такі що задовольняють нерівність $t_{comm,k}^{i\to j} \geq t_{rdy,k}^{i}$, максимізують ефективність використання обчислювальної системи $K_E \to \max$ та мінімізують обсяг пам'яті, що використано в усіх вузлах $\sum_{i=1}^{K} M_i \to \min$, з урахуванням можливості багаторазової зміни вузла j на j' у проміжок часу $\left[t_{rdy,k}^{i}, min\{t_{req,k}^{j}, t_{req,k}^{j'}\}\right]$. У якості критерію ефективності використовується коефіцієнт ефективності K_E, який визначається відношенням прискорення паралельної реалізації обчислень у порівнянні з послідовною реалізацією до кількості процесорів N. [3] При визначенні $t_{comm,k}^{i\to j}$ також врахувати час, необхідний на безпосередню передачу даних між вузлами $t_{T,k}^{i\to j}$, який залежить від структури системи та обсягу даних.

Необхідно розглянути три наступні випадки співвідношень між часом запиту даних $t_{req,k}^{j}$ і часом готовності даних $t_{rdy,k}^{i}$.

а) Дані обчислені раніше, ніж вони використовуються вперше: $t_{rdy,k}^{i} < t_{req,k}^{j}$. В цьому випадку моделі, що забезпечують підтримку негайної передачі даних виконають переміщення даних у момент готовності останніх $t_{comm,k}^{i\to j} = t_{rdy,k}^{i}$, а моделі, що забезпечують підтримку відкладеної передачі даних виконають переміщення у момент запиту $t_{comm,k}^{i\to j} = t_{req,k}^{j}$. При цьому, незалежно від підтримуваного способу передачі даних, обсяги пам'яті, що використовується в обох процесах, мають підпорядковуватися обмеженням

$$\begin{cases} M_i\left(t_{comm,k}^{i\to j} + t_{T,k}^{i\to j} + 1\right) = M_i\left(t_{comm,k}^{i\to j}\right) - m_k \\ M_j\left(t_{comm,k}^{i\to j} + t_{T,k}^{i\to j}\right) = M_j\left(t_{comm,k}^{i\to j} - 1\right) + m_k \\ M_i\left(t_{comm,k}^{i\to j}\right) \leq M_i(t) \leq M_i\left(t_{comm,k}^{i\to j} + t_{T,k}^{i\to j} + 1\right) \\ \forall t: 0 \leq \left(t - t_{comm,k}^{i\to j}\right) \leq t_{T,k}^{i\to j} \\ M_j\left(t_{comm,k}^{i\to j}\right) \leq M_j(t) \leq M_j\left(t_{comm,k}^{i\to j} + t_{T,k}^{i\to j} + 1\right) \\ \forall t: 0 \leq \left(t - t_{comm,k}^{i\to j}\right) \leq t_{T,k}^{i\to j} \end{cases} \quad (1)$$

де перші два рівняння показують, що пам'ять, яка була використана для зберігання даних в i-тому вузлі може бути звільнена після завершення передачі, а в j-тому вузлі після передачі має бути виділено достатньо пам'яті для зберігання отриманих даних; а останні дві нерівності показують, що виділення та звільнення пам'яті може відбуватися будь-яким чином в процесі передачі, наприклад лінійно або стрибкогодібно на початку та у кінці передачі. В даному випадку необхідно визначити такий час початку передачі $t_{comm,k}^{i\to j}$, $t_{rdy,k}^{i}$, $\leq t_{comm,k}^{i\to j}$, $\leq t_{req,k}^{j}$, що максимізує ефективність та мінімізує максимальний обсяг використаної пам'яті на проміжку часу $\left[t_{rdy,k}^{i},\ t_{req,k}^{j} + t_{T,k}^{i\to j},\right]$:

$$\max_{t_{rdy,k}^{i} \leq t \leq t_{req,k}^{j} + t_{T,k}^{i\to j}} (M_i(t) + M_j(t)) \to min$$

Слід зазначити, що обсяг пам'яті, що використана в кожному з процесів, може залежати від значень вхідних даних або результатів попередніх обчислень, тому недоцільно виконувати оптимізацію статично.

В цьому випадку також необхідно врахувати можливість зміни вузла j, на якому фактично будуть виконані обчислення, у час $t_{migr,k}^{i\to j'}$, такий що $t_{rdy,k}^{i} < t_{migr,k}^{i\to j'} < \min\ \{t_{req,k}^{j}, t_{req,k}^{j'}\}$ внаслідок роботи методів балансування навантаження. При цьому, якщо k-тий блок даних вже було передано у локальну пам'ять j-того вузла, необхідно передати його із локальної пам'яті вузла j у локальну пам'ять вузла j'. Для мінімізації часу, необхідного на передачу даних, необхідно також мінімізувати різницю між часом передачі та часом запиту даних

$$t_{req,k}^{j} - t_{comm,k}^{i\to j} - t_{T,k}^{i\to j} \to min,$$

Оскільки рішення про балансування навантаження приймаються під час роботи програми, необхідно перерозрахувати $t_{comm,k}^{i\to j'}$ після нього.

б) Дані готові в той самий момент, у який вони мають бути використані вперше $t_{rdy,k}^{i} = t_{req,k}^{j}$. В цьому випадку, оскільки розглядаються лише різні вузли $i \neq j$, також необхідно виконати передачу даних. При цьому на безпосередню передачу буде витрачено час $t_{T,k}^{i\to j'}$. Розпочати передачу даних у час $t_{comm,k}^{i\to j'} < t_{rdy,k}^{i}$ неможливо, оскільки дані ще не були обчислені. Розпочати передачу даних із затримкою t_{ω} недоцільно, оскільки під час цієї обчислення виконуватись не будуть, звідки випливає, що відношення часу безпосередніх обчислень до загального часу виконання задачі зменшиться, тому загальна ефективність використання системи під час обчислень знизиться. Таким чином, в даному випадку необхідно ініціювати передачу даних у момент часу $t_{comm,k}^{i\to j'} = t_{rdy,k}^{i} = t_{req,k}^{j}$.

Оскільки запит на використання даних відбувається одночасно із закінченням їх підготовки, ані вузол *i*, який виконує підготовку, ані вузол *j*, якому необхідні ці дані для подальших обчислень, не можуть бути змінені через балансування навантаження: підготовка даних щойно закінчено, тому переміщення обчислень щодо підготовки не відбуватиметься; дані для подальшого використання вже запитані, тому будь-яке переміщення наступних неможливе до отримання цих даних, інакше дані можуть бути передані у невірний вузол.

в) Запит на використання даних відбувся раніше, аніж вони були підготовлені $t^{i}_{rdy,k} > t^{j}_{req,k}$. Цей випадок найбільш суттєво впливає на ефективність обчислень, оскільки система буде простоювати впродовж $(t^{i}_{rdy,k} - t^{j}_{req,k})$. А також на безпосередню передачу буде витрачено час $t^{i\to j}_{T,k}$. Розпочати передачу даних у час $t^{i\to j\prime}_{comm,k} < t^{i}_{rdy,k}$ неможливо, оскільки дані ще не були обчислені. Розпочати передачу даних із затримкою t_{ω} недоцільно за міркуваннями, аналогічними до пункту (б).

В цьому випадку необхідно врахувати можливість зміни вузла і, на якому виконується підготовка даних. Якщо на момент часу $t^{j}_{req,k}$ підготовка даних ще не розпочалась, у час t < $t^{i}_{rdy,k}$внаслідок роботи алгоритмів балансування навантаження може бути змінено вузол, на якому фактично будуть виконані обчислення щодо підготовки *k*-го блоку даних. Завдяки подібній заміні може бути змінений час отримання даних, запит на використання яких вже видано, $t^{i\prime}_{rdy,k} \leq t^{i}_{rdy,k}$, що в свою чергу може змінити час ініціювання передачі даних.

Вироджений випадок $i = j$ не розглядається, оскільки виконувати передачу даних між вузлами для нього не потрібно, та вони стають доступні відразу після їх обчислення. Для узгодженості приймемо у цьому випадку $t^{i\to j}_{comm,k} = t^{i}_{rdy,k}$.

Питання визначення оптимального маршруту для передачі даних між вузлами обчислювальної системи з точки зору швидкості такої передачі відноситься до розв'язання задачі маршрутизації, для якої запропоновано ряд ефективних алгоритмів. [4]

Питання керування ресурсами обчислювальної системи та виділення ресурсів для виконання обчислень, а також розподілу виділених ресурсів системи для виконання певних частин обчислень відносяться до функцій операційної системи. Підходи до розв'язання цієї задачі глибоко вивчені та існує велика кількість способів її розв'язання, [5] тому в даному досліджені ці питання не розглядаються, а використовуються відомі та реалізовані підходи.

Оцінка ефективності

Будемо вважати, що обчислення виконуються на паралельній обчислювальній системі з локальною пам'яттю, вузли якої можуть містити спільну для певної невеликої кількості процесорів пам'ять. Обчислювальна система з локальною пам'яттю також може містити акселератори певного виду, які розглядаються як окремі вузли з локальною пам'яттю, оскільки вимагають явного перенесення необхідних даних між пам'яттю акселератора та основною пам'яттю вузла. Нехай система містить K вузлів, при чому *i*-тий вузол в свою чергу містить ni процесорів. В цьому разі загальна кількість процесорів, що доступні в обчислювальній системі $N = \sum_{i=1}^{K} n_i$. Для спрощення вважаємо, що в системі виконуються лише обчислення що розглядаються, а також складові частини операційної системи та система керування паралельними обчисленнями. При цьому обсяги обчислень, що виконуються двома різними процесорами за одиницю часу, можуть відрізнятися, а також певна частина обчислень може виконуватись лише на певних заздалегідь визначених процесорах, що має бути коректно забезпечено операційною системою.

Позначимо час виконання деякої паралельної реалізації певного алгоритму на обчислювальній системі, що розглядається, T. Час виконання розраховується від моменту початку обчислень до моменту отримання кінцевого результату. При цьому, час активного використання *j*-го процесора даними обчисленнями складає t_j, таке що

$$t_j \leq T \qquad (2)$$

через необхідність очікування введення та виведення даних, комунікації з іншими процесами, отримання доступу до спільного ресурсу тощо. Таким чином, час послідовних обчислень тієї ж самої реалізації алгоритму на даній обчислювальній системі складався б з суми часу обчислень кожним процесором окремо $\sum_{j=1}^{N} t_j$. Тоді коефіцієнт прискорення визначається відношенням оціненого часу обчислень послідовно до часу обчислень паралельно

$$K_{\text{п}} = \frac{\sum_{j=1}^{N} t_j}{T} \quad (3)$$

при чому, з урахуванням (2), можна відмітити, що коефіцієнт прискорення не має перевищувати кількості наявних процесорів

$$K_{\text{п}} = \frac{\sum_{j=1}^{N} t_j}{T} \leq \frac{\sum_{j=1}^{N} T}{T} = \frac{NT}{T} = N \quad (4)$$

В такому разі коефіцієнт ефективності визначається як відношення коефіцієнту прискорення до кількості наявних процесорів

$$K_E = \frac{K_{\text{п}}}{T} \quad (5)$$

при чому, з урахуванням (4), коефіцієнт ефективності може приймати значення в проміжку [0, 1], оскільки

$$0 \leq K_E = \frac{K_{\text{п}}}{T} \leq \frac{N}{N} = 1$$

Для подальшої оптимізації необхідно більш детально розглянути фактори, що впливають на сумарний час виконання обчислень. Зокрема слід проаналізувати складові сумарного часу обчислень, що складається з

$$T = t_c + t_{i/o} + t_w + t_s$$

де $t_c = \sum_j t_j$ – час безпосереднього виконання обчислень (англ. computations); $t_{i/o}$ – час введення та виведення даних, в тому числі з використанням мережі для обміну даних з іншими процесами (англ. input and output); t_w – час очікування (англ. wait); t_s – час неактивності через роботу системних потоків (англ. sleep). Можна відмітити, що коефіцієнт ефективності обмежуються відношенням часу безпосереднього виконання обчислень до суми всіх інших компонентів сумарного часу виконання

$$K_E = \frac{K_{\text{п}}}{N} = \frac{\sum_{j=1}^{N} t_j}{TN} = \frac{t_c}{(t_{i/o} + t_w + t_s)N}$$

що відповідає обмеженням на коефіцієнт прискорення, встановленими законом Амдала [6]. Також необхідно більш детально розглянути складові часу очікування, які можуть бути зменшені в рамках запропонованого підходу

$$t_w = t_{i/o} + t_{w,swap} + t_{w,e}$$

де $t_{w,i/o}$ — час очікування введення або виведення даних, в тому числі з використанням мережі для зв'язку з іншими процесами, $t_{w,swap}$ – час очікування підкачки даних в основну пам'ять в рамках одного вузла, $t_{w,e}$ – час очікування інших подій, зокрема взаємного виключення. Оцінка часу очікування підкачки даних в основну пам'ять є вкрай важливою характеристикою, оскільки за умови передачі даних у вузол значно раніше, аніж вони будуть оброблені, виникає необхідність їх зберігати. При цьому збільшується загальний обсяг використовуваної пам'яті а також на певний час після завершення передачі змінюється статистика звернень до сторінок пам'яті, що може негативно вплинути на ефективність роботи системи підкачки даних у віртуальній організації пам'яті. Аналогічний вплив може мати і зберігання даних на певному вузлі, після їх обчислення, тому необхідно розглядати роботу з основною пам'яттю більш детально. Для цього позначимо загальний обсяг використаної основної пам'яті

$$M(t) = \sum_{i=1}^{k} M_i(t) = \sum_{i=1}^{K} \sum_{l=1}^{n_i} M_{i,l}(t) + M_{i,s}(t) \quad (6)$$

де M_i – обсяг пам'яті, використаної в і-тому вузлі зі спільною пам'яттю, $M_{i,l}$ – обсяг пам'яті, використаної в l-тому процесі, що виконується на i-тому вузлі, а $M_{i,s}$ – спільна пам'ять, доступна одночасно всім процесам, що виконуються на i-му вузлі. Такий поділ є доцільним, оскільки в обчислювальній системі, що розглядається, вузли мають спільну пам'ять, і вплив від зберігання даних в одному з процесів, що виконується на даному вузлі, розповсюджується також на інші процеси, що виконуються на даному вузлі. Тому недостатньо розглядати використання пам'яті лише в одному процесі. Обсяг пам'яті, що використовується, є функцією часу. Для спрощення аналізу, можна ввести статистичні показники M_Σ — сумарний обсяг пам'яті, який було виділено під час виконання обчислень; $\widehat{M}$ – середній обсяг пам'яті, який було виділено під час виконання обчислень; M^* – піковий обсяг пам'яті, який одночасно було виділено під час виконання обчислень. Для характеристики роботи системи підкачки даних в основну пам'ять також необхідно аналізувати кількість сторінок пам'яті, які були збережені поза основною пам'яттю та пізніше були завантажені за запитом. Це типово характеризується кількістю так званих *кеш-промахів* (англ. cache miss), тобто кількістю звернень до сторінок пам'яті, які не були розміщені в основній пам'яті на момент звернення. Під час такого промаху виконується очікування підкачки даних в основну пам'ять, яке впливає на час очікування tw,swap. Позначимо загальну кількість кеш-промахів C, та кількість кеш-промахів за одиницю часу $C' = \frac{C}{T}$.

Ряд характеристик варто також розглядати не лише для обчислювальної системи в цілому, а також для кожного процесу окремо. До таких

характеристик відносяться час очікування введення та виведення $t^{j}_{w,i/o}$ в j-тому процесі, час очікування підкачки даних $t^{j}_{w,swap}$ в j-тому процесі, а також час виконання введення та виведення $t^{j}_{i/o}$ та час безпосередньо обчислення t^{j}_{c} в кожному процесі. Це також дозволяє розглядати ряд статистичних значень, які загалом характеризують процес виконання обчислень, таких як середні за процесами значення часу очікування введення та виведення $\hat{t}^{j}_{w,i/o}$часу очікування підкачки даних $\hat{t}^{j}_{w,swap}$ або максимальні значення цих характеристик $t^{*}_{w,i/o}$ $t^{*}_{w,swap}$. Перші впливають на загальний час виконання обчислень в цілому, в той час як останні відіграють роль обмежуючого фактора.

Розглянуті вище характеристики мають опосередкований вплив на результуючий коефіцієнт ефективності, шляхи цього впливу показані на рис. 1. Тобто обсяг використаної в кожному вузлі пам'яті Мі впливає на кількість кеш-промахів С, однак не є єдиним фактором, що її визначає. Кількість кеш-промахів впливає на час очікування підкачки даних $t_{w,swap}$, який в свою чергу впливає на загальний час виконання обчислень. З іншого боку, час очікування введення та виведення $t_{w,i/o}$ безпосередньо впливає на загальний час виконання обчислень. На рис. 1 рамкою виділені дві характеристики, M_i та $t_{w,i/o}$, які можна змінювати при різних підходах до організації пересилання даних.

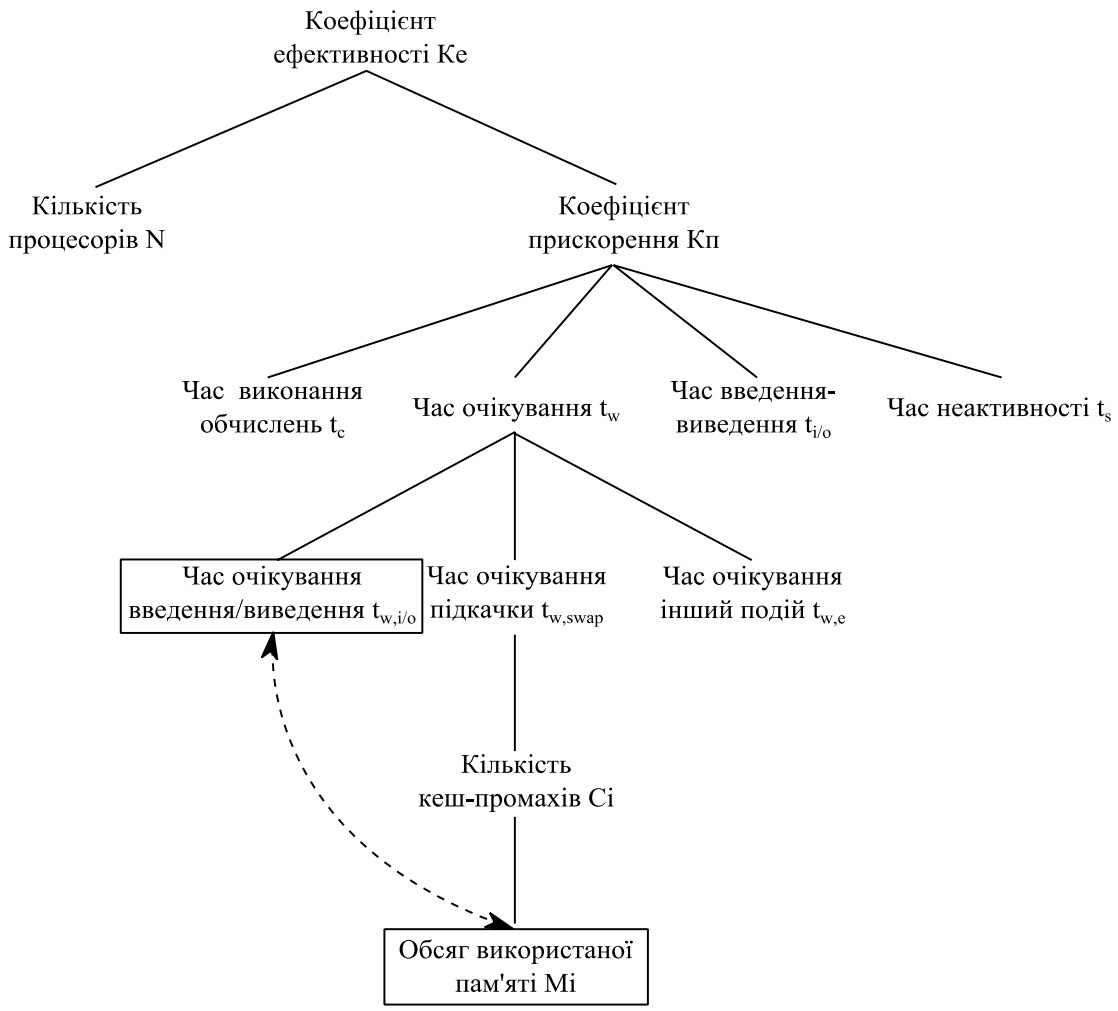


Рисунок 1. Вплив динамічних характеристик на коефіцієнт ефективності

Однак, дані характеристики також впливають одна на одну, що відмічено пунктирною лінією на рис. 1. Так, зменшення часу очікування передачі даних, тобто введення та виведення, може бути досягнуте завдяки буферизації k-го блоку даних на вузлі j, в якому будуть виконуватись обчислення, перед тим, як вони будуть запитані

$$\left(t^{i\to j}_{comm,k} + t^{i\to j}_{T,k}\right) < t^{i}_{req,k} \Rightarrow \left(t^{j}_{w,\frac{i}{o}} - t^{j'}_{w,\frac{i}{o}}\right) \geq t^{i\to j}_{T,k}$$

однак це призведе до збільшення обсягів використаної пам'яті

$0 \leq M_i(t)' - M_j(t) \leq m_k \forall t > t_{comm,k}^{i \to j}$.

Таким чином, характеристики вступають у протиріччя, покращення однієї з них може мати погіршити іншу та навпаки, тому необхідно досліджувати їх сумарний вплив на коефіцієнт ефективності

$$\begin{cases} \sum_{j=1}^{N} t_{w,i/o}^{j} \to min \\ \max_j t_{w,i/o}^{j} \to min \\ \sum_{i=1}^{K} \widehat{M_i} \to min \\ \max_i \widehat{M_i} \to min \\ K_E \to max \end{cases} \quad (7)$$

Відповідно до вищесказаного, найбільш важливим критерієм є коефіцієнт ефективності, який характеризує виконання обчислень в цілому. Однак для його підвищення з використанням автоматизованих засобів, необхідно проаналізувати ряд характеристик, що впливають на його значення. До них відносяться час очікування введення та виведення даних, час очікування підкачки даних в основу пам'ять, обсяг використаної пам'яті та кількість кеш-промахів. Застосування певних засобів мінімізації використаної пам'яті безпосередньо впливає на час введення та виведення даних та навпаки, тому для збільшення коефіцієнту ефективності необхідно розглядати та покращувати перелічені характеристики одночасно.

Таблиця 1

Порівняння обраних моделей та технологій програмування паралельних обчислювальних систем

Технологія	Цільова система	Особливості доступу до даних
Вказівки компілятору (OpenMP)	Обчислювальна система зі спільною пам'яттю	Копіювання автоматичне відповідно до вказівок; додаткові вказівки для взаємного виключення; лише спільна пам'ять.
Низькорівневий інтерфейс передачі повідомлень (MPI)	Обчислювальна система з локальною пам'яттю	Обов'язкове явне копіювання даних між вузлами; можливість програмованої буферизації під час передачі.
Низькорівневий інтерфейс передачі повідомлень з оптимізацією (MPI-2/3)	Обчислювальна система з локальною пам'яттю, вузли якої містять спільну пам'ять	Те ж саме, що MPI, та додатково використання спільної пам'яті в рамках вузла за можливості, технології прямого дос-тупу у віддалену пам'ять.
Інтерфейс програмування графічних акселераторів CUDA	Обчислювальна система зі спільною пам'яттю та застосуванням графічних акселераторів nVidia	Обов'язкове явне копію-вання даних в пам'ять акселератора. Багато-рівнева ієрархія пам'яті.
Низькорівневий інтерфейс передачі повідомлень з урахуванням графічних акселераторів (CUDA-MPI)	Обчислювальна система з локальною пам'яттю та застосу-ванням графічних акселераторів nVidia	Те ж саме, що CUDA, та додатково можливість явного копіювання у пам'ять акселератора в іншому вузлі.
Технологія Array Building Blocks	Обчислювальна система зі спільною пам'яттю	Автоматичне взаємне виклю-чення в екземплярах спе-ціально структурованих типів; лише спільна пам'ять.
Технологія Threading Building Blocks	Обчислювальна система зі спільною пам'яттю та застосу-ванням акселераторів типу Xeon Phi	Автоматичне копіювання на акселератор та взаємне вик-лючення в екземплярах спе-ціально структурованих типів; лише спільна пам'ять.
Стандартна бібліотека Futures мови C++	Обчислювальна система зі спільною пам'яттю	Можливість відкладеного обчислення та доступу (за без-посереднім запитом), зокрема паралельного. Лише спільна пам'ять.
Стандартна техно-логія відкладених обчислень мови Haskell	Обчислювальна система зі спільною пам'яттю	Можливість відкладеного обчислення та доступу. Лише спільна пам'ять.

Список литературы

1. Стиренко, С.Г. Модель организации вычислений в распределјнной системе / С.Г. Стиренко, А.И. Зиненко, Д.В. Грибенко // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка. – 2012. –Т. 57. – С. 101–109.
2. Stirenko, S.G. Dynamic analysis of MPI applications / S.G. Stirenko, A.I. Zinenko, D.V. Gribenko // Proceedings of the 1st international conference on High-performance computing (HPC-UA), Kiev, 2011. – K. : ВЕК+, 2011. – P. 152–153.
3. Hennessy, John L. Computer architecture: a quantitative approach/ John L Hennessy, David A Patterson. – Amsterdam, Noord-Hooland, Netherlands : Elsevier, 2011. – 824 p.
4. Tanenbaum, A.S. Computer Networks/ A.S. Tanenbaum, D.J. Wetherall. – London, UK : Pearson Education, Limited, 2010. – 1240 p.
5. Симоненко, В. П. Организация вычислительных процессов в ЭВМ, комплексах, сетях и системах / В. П. Симоненко. – К. : ВЕК+, 1997. –304 с.
6. Amdahl, Gene M. Validity of the single processor approach to achieving large scale computing capabilities / Gene M. Amdahl // Proceedings of the April 18-20, 1967, spring joint computer conference. – AFIPS '67 (Spring). – New York, NY, USA : ACM, 1967. – P. 483–485.

Відомості про авторів

Амонс О.А.	к.т.н., доцент, докторант НТУУ «КПІ»
Аракелян Г.А.	заступник декана ФІОТ по науковій роботі НТУУ "КПІ", інженер.,
Бойко В.В.	НТУУ «КПІ»
Болдак А.А.	к.т.н., доцент, докторант, ННК «ІПСА»
Воротников В. В.	к.т.н., доцент кафедри АСУ Житомирского военного института им. С. П. Королева
Голков В.Б.	студент-магістрант НТУУ «КПІ»
Гуменюк И. В.	инженер кафедры АСУ Житомирского военного института им. С. П. Королева
Дьяконова С.В.	аспірант НТУУ «КПІ»
Єфремов К.В.	директор Світового центру даних з геоінформатики та сталого розвитку, ст. викладач НТУУ «КПІ»
Жук С.В.	магістр НТУУ «КПІ»
Забаринская Л.П.	зав. лабораторією, с.н.с., Федеральна державна бюджетна установа науки Геофізичний центр РАН
Зайченко Ю.П.	д.т.н., профессор кафедры ММСА УНК «ИПСА» НТУУ «КПИ»
Згуровский М.З.	академік НАН України, директор, ННК «ІПСА»
Знахур С.В.	к.э.н. доц.кафедры информационных систем Харьковского национального экономического университета
Знахуренко В.П.	НТУУ «КПІ»
Коган А.В.	асистент кафедри ОТ НТУУ «КПІ»
Коровин А.В.	вед. программист ПраТ "Альтрон"
Кулаков Ю. А.	д.т.н., проф. кафедри ОТ НТУУ «КПІ»
Лемешко В.А.	НТУУ «КПІ»
Лисецкий Т.Н.	аспирант кафедры АСОИУ НТУУ «КПИ»
Луцкий Г.М.	д.т.н., проф., завідувач кафедри ОТ НТУУ «КПІ»
Мельник Т.П.	НУ "Львівська політехніка"
Минухин С.В.	к.т.н., проф. кафедры информационных систем Харьковского национального экономического университета
Мисюра Е.Б.	к.т.н, с.н.с. кафедры АСОИУ НТУУ «КПИ»
Молчановський О.І.	старший викладач кафедри АСОІУ НТУУ «КПІ»
Мухин В.Е.	к.т.н., доцент кафедри ОТ НТУУ «КПІ»
Надеран Э.	аспірант НТУУ «КПІ»
Нисилевич М.В.	зав. лабораторією, н.с., Федеральна державна бюджетна установа науки Геофізичний центр РАН
Павлов А.А.	д.т.н., проф. кафедры АСОИУ, декан Факультета информатики и вычислительной техники НТУУ «КПИ»
Півень О.Б.	кандидат фізико-математичних наук, доцент каф. СП Черкаського державного технологічного університету
Полторак В.П.	к.т.н., доцент НТУУ «КПІ»
Приймак А.В.	аспирант Харьковского национального университета им. В.Н.Каразина
Прошин С.Ю.	асистент кафедри комп'ютерних наук Тернопільського національного технічного університету ім. І. Пулюя
Пустоваров В.И.	к.т.н., доцент кафедри ОТ НТУУ «КПІ»
Ролик А.И.	к.т.н., доцент кафедри АУТС НТУУ «КПІ»
Сергеева Н.А.	зав. лабораторією, к.ф.-м.н., Федеральна державна бюджетна установа науки Геофізичний центр РАН
Сергієнко А.М.	д.т.н.,с.н.с. кафедри обчислювальної техніки НТУУ "КПІ"
Сперкач М.О.	ассистент кафедры АСОИУ НТУУ «КПИ»
Стіренко С.Г.	к.т.н., доцент кафедри ОТ НТУУ «КПІ»
Теленик С.Ф.	д.т.н., проф., завідувач кафедри АУТС НТУУ «КПІ»
Федотова-Пивень И.Н.	к.т.н., ст. викладач каф. системного програмування Черкаського державного технологічного університету
Халус Е.А.	ассистент кафедры АСОИУ НТУУ «КПИ»
Шестопалов И.П.	зав. лабор., пров.н.с., к.ф.-м.н., Федеральна державна бюджетна установа науки Геофізичний центр РАН
Шовгун Н.В.	аспірант НТУУ «КПИ»

SUMMARY

Zgurovsky M.Z., Boldak A.A., Efremov K.V., Sergeeva N.A., Zabarinskaya L.P., Shestopalov I.P., Nisilevich M.V. Application of Data Mining Methods for Empirical Studies on the Interrelationship of Helio- and Geophysical Processes…….. 4

Pavlov A.A., Misjura E.B., Lisetsky T.N., Sperkach M.O., Khalus E.A. The four-level model of planning, decision making and operational control in the network systems with limited resources……..... 11

Pavlov A.A., Misjura E.B., Lisetsky T.N. Scheduling independent tasks on the identical parallel machines which starting times are smaller than a common due date…… 24

Telenik S.F., Amons O.A., Efremov K.V., Zhuk S.V. Semantic Integration of Heterogeneous Information Resources…… 29

Kyselova A. Context Data Analysis For Microgrid Control System……. 46

Dyakonova S.V., Zaychenko Y.P. Approach to The Problem of Automated Detection of Buildings on Satellite Images…… 51

Naderan E., Zaychenko Y.P. An Approach to Structural Analysis of Handwritten Mathematical Expressions in Real Time…….. 56

Molchanovsky O.I., Znahurenko V.P. Inductive Learning Method Based on Gifts Recommendation System…... 61

Kulakov Y.A., Vorotnikov V.V., Gumenuk I.V. Analysis of Telecommunication Network by Bringing its Topology to the Form Prefractal Graph…… 68

Kulakov Y.A., Kogan A.V. Determining the Optimal Number of Edge Routers in Secure Multipath Routing of Large Mobile Network…… 73

Rolik A.I. Decomposition-Compensation Method of Service Level Management of Corporate IT Infrastructures. 78

Shovgun N.V. Increasing the Efficiency of Fuzzy Neural Networks in the Problem of Assessing the Creditworthiness of the Borrower……. 89

Poltorak V.P., Golkov V.B. Electronic Digital Systems Signature Comparison And Their Implementation Features On Elliptic Curves Basis……. 95

Pustovarov V.I., Boiko V.V., Lemeshko V.A. Functional Expansion of Industrial Automation and Control Systems…….. 104

Proshyn S. Transition Matrices Estimation of Power Supply Based on Periodic Markov Chain…… 109

Arakelyan G.A., Sergiyenko A.M. Coal Ashes Estimation Using the Simulated Annealing…… 115

Melnyk T.P. Improving Dynamic Simulation Technology Stochastic System For Determining Flood Zones Subject to Dam Diking…….. 119

Pryjmak A.V. Optimization of Computing With CUDA in Simulation of Instability of Langmuir Waves in Plasma……... 125

Fedotova-Piven I.N., Piven O.B. Software Modeling of Coincident in Time Addition of Twenty Positive Integers in the Redundant Recurrent Numeration System of the Third Order…….. 131

Minukhin S.V, Znakhur S.V., Korovin A.V. Coordinate Strategies Job Scheduling Based On Generalized Statistical Analysis of Their Characteristics in the Distributed Computing Systems……... 137

Lutsk G.M., Muhin V.E. Routing Methods in Computer Systems With Multi-Channel Links……. 148

Stirenko S.G. Model of Parallel Programming Technology……. 158

Information about authors…… 165

Summary……..… 166