УДК 681.3

САДЕГ РАСТГУ

МОДЕЛИ ВЫЧИСЛИТЕЛЬНЫХ ПРОЦЕССОВ В КЛАСТЕРНЫХ СИСТЕМАХ С МНОГОЯДЕРНОЮ АРХИТЕКТУРОЮ

В работе рассматриваются подходы к разработке математических моделей, описывающих поведение кластерной системы с многоядерною архитектурою. Проанализированы особенности организации вычислений в таких системах, связанных с двухуровневой параллельной обработкой. Представлена математическая модель кластерной системы, основанная на теории последовательных взаимодействующих процессов.

In paper a problem of creation a model of cluster systems with multicore architecture is considered. An analyze of computing in such systems related with two level of parallelisms is presented. Mathematical model of cluster system based on theory of sequential communicating processes is presented.

1. Введение

Развитие высокопроизводительных компьютерных систем в настоящее время связано с распределенными (кластерными) системам. Последняя редакция рейтинга суперкомпьютеров TOP500 (31-я редакция, июнь 2008 года) включает 61 кластерную систем в первых ста позициях. Количество процессоров в этих кластерных системах варьируется в диапазоне от 1992 до 122400, максимальная производительность – в диапазоне от 18,81 до 1026 Tflop/s, пиковая производительность – в диапазоне от 24,58 до 1375,78 Tflop/s. На сегодня наиболее мощной является кластерная система Roadrunner компании IBM, установленная в Лос-Аламосской национальной лаборатории (США).

Современная кластерная система (КС) включает набор вычислительных узлов, соединенных специальной быстродействующей системой связей, основанной на сетевых технологиях. Узел КС обеспечивает обработку и хранение данных, система связей – передачу данных между узлами КС.

Система связи узлов (ССУ) является важнейшей составляющей КС, так как эффективность КС в значительной мере определяется скоростью обмена данными между узлами системы. Основу всех реализаций систем связей составляют принципы, заимствованные их сетевых технологий, поэтому в КС широко используются технологии типа Ethernet (GigabitEthernet), которые на сегодня обеспечивают пропускную способность сети до 10Gbit/sec. Наряду с Ethernet технологией в современных КС используются специализированные системы связей типа Myrinet, Infiniband, cLan, SCI, Quadrics. Анализ кластерных систем, представленных в первых 100 позициях 31-го рейтинга TOP500 показал, что наиболее используемой коммуникационной технологией является InfiniBand (42 позиция). За ней следует технология GigEthernet (10). В списке представлены также технологии Myrinet (7) и QsNet (2).

Рост производительности КС традиционно связывался с увеличением количества узлов системы. Однако наращивание числа узлов ограничено из-за повышенного требования к ССУ при увеличении узлов и нагрузке на систему связей. Имеется второй путь повышения производительности КС, основанный на увеличении вычислительной мощности непосредственно самого узла. Такой подход обеспечивается использованием в узлах мощных процессоров, также реализацией в узлах КС принципов параллельной обработки за счет применения многопроцессорных систем. Использование в узлах многопроцессорных систем позволяет перейти в КС к двухуровневой параллельной обработке. Первый уровень обеспечивается параллельной обработкой на множестве узлов КС, второй – на множестве процессоров внутри узла.

Появление многоядерных процессоров открывает перспективы увеличения мощности узлов за счет более эффективной реализации второго уровня параллельной обработки в КС. Простая замена в узле обычного процессора на двухядерный процессор позволяет удвоить вычислительную мощность КС без изменения ее структуры. Появление на рынке четырехядерных процессоров и анонсированное в ближайшем будущем появления процессоров с восьмью и более ядрами, открывает возможности резкого увеличения производительности КС при построении систем на основе мно-

гоядерных процессоров. В этой связи следует отметить, что 31-я редакция TOP500 характеризуется широким использованием в компьютерных системах четырехядерных процессоров нового поколения (258 систем).

Целью настоящей работы является анализ особенностей организации вычислительных процессов в кластерных системах с многоядерною архитектурою (КСМА) и построение математических моделей, позволяющих формализовать эти особенности, которые позволят в полной мере реализовать все преимущества структурной организации КСМА.

2. Модели вычислительных процессов в КСМА

Рассмотрим КСМА как систему, поведение которой можно описать с помощью алгебры процессов, предложенной в теории взаимодействующих последовательных процессов (CSP) Ч.Хоара [1]. Выбор теории Ч.Хоара объясняется тем, что она позволяет описать любой объект с помощью процессов. Модель, основанная на понятии процесс, в свою очередь может быть эффективно реализована с помощью современных языков параллельного программирования, базирующихся на механизме потоков (легких процессов).

При разработке модели будем исходить из того, что КСМА является сложной системой и имеет два уровня. Первый уровень – система узлов, второй уровень – система ядер внутри узла. Тогда модель КСМА имеет вид иерархической структуры, состоящей из моделей двух уровней (КМСА1 и КСМА2). Модель первого уровня описывает поведение системы, связанные с узлами, а модель второго уровня – с поведением системы внутри узла.

2.1 Модель КСМА1. Для модели первого уровня КСМА1 используем модель клиент-сервер, которая включает объект – сервер и множество объектов – клиентов.

В теории Т.Хоара процесс определяет поведение объекта и может быть описан с помощью ограниченного набора событий, выбранного в качестве его алфавита.

Набор событий, связанных с объектом сервер (алфавит сервера) C можно представить в виде

$$\alpha C=\{\ вводC, выводC,\ принятьСК_i,\ передатьСК_i\ \ счетC\ \}$$

где события: $вводC$ – ввод данных на сервере, $выводC$ – вывод результат на сервере, $принятьСК_i$ – прием сервером данных от клиента, $передатьСК_i$ – передача данных от сервера клиенту $К_i$, $счетC$ – счет на сервере.

Набор событий, связанных с i-м объектом клиентом ($К_i$) можно представить в виде

$$\alpha К_i=\{принятьК_iC,\ передатьК_iC,\ счетК_i,\ принятьК_iК_j,\ передатьК_iК_j\ \}$$

где события: $принятьК_iC$ – прием данных от сервера, $передатьК_iC$ – передача данных серверу, $счетК_i$ – счет на клиенте, $принятьК_iК_j$ – прием данных от клиента $К_j$, $передатьК_iК_j$ – передача данных клиенту $К_j$.

Поведение объекта сервера используя алфавит αC, можно описать как процесс

$$C=(вводC \rightarrow передатьСК_i \rightarrow счетC \rightarrow принятьСК_i \rightarrow выводC\ )$$

Для множества объектов клиентов $К_i$ их поведение (клиентский процесс) описывается следующим образом

$$К_i =(\ принятьК_iC \rightarrow передатьК_iК_j \rightarrow счетК_i \rightarrow принятьК_iК_j \rightarrow передатьК_iC\ )$$

Поведение КСМА1 можно описать как параллельную комбинацию поведения процессов C и $К_i$:

$$КЛИЕНТЫ=(К1||К2||\ .\ .\ .||КР),\ СЕРВЕР=(C)$$
$$КСМА1\ =\ (СЕВЕР\ ||\ КЛИЕНТЫ\ )$$

Взаимодействие процессов в модели КСМА1

Взаимодействие узлов в КСМА1 осуществляется с помощью посылки сообщений. Будем рассматривать взаимодействие как событие. Взаимодействие состоит в передаче сообщения и является событием, требующим участия двух процессов.

Для описания взаимодействия процессов введем понятие объекта коммуникации ($ОК$). Алфавит $ОК$ включает следующие события:

$$\alpha ОК\ =\ \{\ записьОК,\ \ чтениеОК\ \}$$

Процесс, описывающий поведение объекта $ОК$:

$$ОК\ =((\ записьОК \rightarrow чтениеОК\ |\ чтениеОК \rightarrow записьОК\ ) \rightarrow ОК\ )$$

Составляющей $ОК$ является спецификация C, который определяет тип и объем передаваемых данных ($ОК.C$) .

В модели КСМА1 взаимодействие объектов связано с событиями *принять* и *передать*. При этом взаимодействие должно быть связано с $ОК$, куда один объект передаст данные, а другой объект произведет считывание данных. Дополним события принять и передать параметром, определяющим используемый

объект коммуникации: *передатьСК*(*ОК*), *принятьСК*(*ОК*). Процесс *ОК* можно рассматривать как сопроцесс [1].

2.2 Модель КСМА2. Узел будем рассматривать как систему, включающую набор объектов, связанных с действиями, выполняемыми в многоядерном процессоре. При организации процессов внутри узла используем модель взаимодействия хост-рабочий [2]. Хост объект отвечает за связь с остальными узлами КСМА, и, в частности, – с серверным узлом КСМА. Хост объект (*X*) принимает исходные данные от сервера КСМА, распределяет их в узле по объектам – рабочим, собирает результаты счета в узле и отсылает результат в серверный узел КСМА. Объект рабочий (*P*) принимает данные от хост объекта, выполняет счет и возвращает результат хосту.

Набор событий, связанных с объектом хост *i*-го узла (*Xi*), можно представить в виде

$$\alpha X_i = \{\ принятьX_i,\ передать\ X_iP_{ij},\ счетX_i,\ принятьX_iP_{ij},\ передать\ X_i\ \}$$

где события: $принятьX_i$ – прием данных от другого узла, $передатьX_iP_{ij}$, – передать данные в узле объекта рабочий P_iP_{ij}, $счетX_i$, – счет, $принятьX_iP_{ij}$ – принять данные в узле от объекта рабочий P_{ij}, $передатьX_i$ – передать данные в другой узел.

Набор событий, связанных с объектом *j*-ий рабочий *i*-го узла (P_{ij}) можно представить в виде

$$\alpha P_{ij} = \{принятьP_{ij}X_i,\ передатьP_{ij}X_i, счетP_{ij},\ принятьP_iP_{iк},\ передатьP_{ij}P_{iк}\}$$

где события: $принятьP_{ij}X_i$ – прием данных от хост объекта, $передатьP_{ij}X_i$ – передача данных хост объекту, $счетP_{ij}$ – счет в хост объекте, $принятьP_iP_{iк}$ – прием данных от объекта рабочего $P_{iк}$, $передатьP_{ij}P_{iк}$ – передача данных объекту рабочему Рік..

Описание поведения объектов хост и рабочий будет зависеть от системы связей ядер в многоядерном процессоре, которая будет определять форму взаимодействия объектов хост и рабочий. Увеличение количества ядер ставит проблему организации связей между ядрами. Используемая в 2-4 ядрах шинная архитектура станет тормозом для эффективной связи ядер и неизбежным будет переход на распределенную архитектуру. Так, например, связывание ядер с помощью топологии типа решетка позволит отказаться централизованных ресурсов, а также решить проблему надежности при выходе из строя ядер или связей.

Если связь ядер в многоядерном процессоре реализована через распределенную архитектуру с помощью, например, топологии звезда, где все ядра связаны через хост ядро, то поведение объектов внутри узла можно описать следующим образом.

Поведение хост объекта X_i (процесс хоста) используя алфавит αX_i, можно описать как

$$X_i = \{\ принятьX_i \rightarrow передатьX_iP_{ij} \rightarrow счетX_i \rightarrow принятьX_iP_{ij} \rightarrow передатьX_i\ \}$$

Поведение объекта рабочий P_{ij}:

$$P_{ij}=(принятьP_{ij}X_i \rightarrow счетP_{ij}, \rightarrow передатьP_{ij}X_i\ )$$

Модель второго уровня для*i*-го клиентского узла можно описать как параллельную комбинацию поведения процессов X_i и P_{ij}:

$$ХОСТ_i = (X_i),\ РАБОЧИЕ_i = (P_i\ )$$

$$УЗЕЛ_i = (\ X_i \parallel РАБОЧИЕ_i\ )$$

Взаимодсействие процессов в модели КСМА2.

Организация взаимодействия процессов внутри узла зависит от выбранной системы связей ядер. При прямой связи ядер (распределенная архитектура) используется модель, основанная на посылке сообщений, для связи через память (шинная архитектура) – модель, основанная на общих переменных.

При использовании посылки сообщений справедлива схема взаимодействия через объекты коммуникации (*ОК*), рассмотренная для модели первого уровня КСМА1.

Рассмотрим подробнее организацию взаимодействия процессов при использовании модели, основанной на общих переменных. Данная модель связана с решением двух задач: задачи взаимного исключения и задачи синхронизации [4]. Введем понятие объекта синхронизации (*ОС*), который будет использоваться процессами при взаимодействии через общую память. Выделим три вида объектов синхронизации. Первый вид *ОС*(*А*) связан с неделимыми (атомик) операциями на переменными. Второй вид *ОС*(*S*) реализует концепцию низкоуровневых средств синхронизации, используемых в примитивах типа семафор, мютекс, событие. Третий вид *ОС*(*М*) реализует концепцию мониторов.

Объект синхронизации типа А

$$\alpha A = \{\ чтениеA,\ записьA,\ счетA\ \}$$

где *чтениеА* – чтение неделимой переменной, *записьА* – запись в неделимую переменную,

счетA – счет с использованием неделимой переменной.

Объект синхронизации типа S

αS ={ *установитьS, проверитьS, сигналS* }

где *установитьS* – установка начального несигнального состояния *S*, *проверитьS* – проверка состояния *S*; блокирование процесса при занятом (несигнальном) состоянии – *S*, *сигналS* – установка *S* в сигнальное состояние (при создании *ОС* принимает сигнальное состояние).

Объект синхронизации типа M

Набор событий, связанных с объектом-монитором *M* можно представить в виде

αM ={ *записьM, чтениеM, счетM, ждатьM, сигналM* }

где события: *записьM* – запись данных в монитор, *чтениеM* – чтение (копирование) данных из монитора, *счетM* – счет в мониторе, *ждатьM* – ожидание сигнала, *сигналM* – сигнал.

Использование объектов синхронизации в задачах взаимного исключения и синхронизации.

Объект A. Обращение процессов к общему ресурсу связано со следующими событиями в объектах *X* и *P*:

X: (*записьA*), *P*: (*чтениеA*)

Объект S. Обращение процессов к общему ресурсу связано со следующими событиями в объектах *X* и *P*:

X: (*проверитьS* $\rightarrow$ *счетX* $\rightarrow$ *сигналS*)

P: (*проверитьS* $\rightarrow$ *счетP* $\rightarrow$ *сигналS*)

Синхронизация связано со следующими событиями (хост посылает сигнал рабочему):

X: (*сигналS*), *P:* (*проверитьS*)

Взаимное исключение. **Объект M.** Обращение процессов к общему ресурсу связано со следующими событиями в описании объектов *X* и *P*:

X:(*записьM,чтениеM,счетM, вводM, выводM*)

P: (*чтениеM, записьM, счетM*)

Синхронизация связано со следующими событиями в описании объектов *X* и *P*:

X:(*ждатьM,сигналM*),*P*:(*сигналM, ждатьM*)

3. Применение моделей

Предложенные модели позволяют применить математический аппарат теории алгебры процессов Ч.Хоара для анализа вычислительных процессов в КСМА.

Рассмотрим анализ взаимодействия процессов на возможность возникновения тупиковой ситуации при синхронизации двух процессов через объект синхронизации типа *ОС*. Пусть *P* и *Q* – процессы, *D* – объект синхронизации типа *ОС*(*S*). Процессы синхронизируются по событию (ввод данных) в процессе *P* (процесс *Q* ждет сигнала от процесса *P*), после чего процессы начинают счет.

Алфавиты процессов будут иметь следующий вид:

αP = { *ввод, счет, установитьD, сигналD*}

αQ = { *счет, проверитьD, установитьD* }

αD = { *проверитьD, установитьD, сигналD* }

Поведение процессов:

P = (*установитьD* $\rightarrow$ *ввод* $\rightarrow$ *сигналD* $\rightarrow$ *счет*)

Q = (*проверитьD* $\rightarrow$ *счет*)

Процесс *D* следует рассматривать как подчиненный процесс по отношению к процессам *P* и *Q*. При этом $\alpha D \subseteq (\alpha P \cup \alpha Q)$. В комбинации (*P* || *Q* || *D*) каждое действие *D* может произойти, если это позволяют процессы *P* и *Q*.

Тупиковая ситуация при синхронизации процессов *P* и *Q* может возникнуть в двух случаях: а)объект *D* не установлен изначально в несигнальное состояние; б)процесс *P* не посылает сигнал о событии, которое ожидает процесс *Q*. Эти действия в процессах связаны с событиями *проверитьD, сигналD, установитьD*. Для борьбы с тупиком необходимо выполнить анализ алфавитов процессов и поведения процессов. Необходимое условие отсутствия тупика – наличие этих событий в алфавитах процессов:

{*проверитьD, сигналD, установитьD* } $\subseteq$ ($\alpha P \cup \alpha Q$)

Достаточные условия – наличие события *сигналD* в алфавите процесса *P* и события *проверитьD* в алфавите процесса *Q*:

{ *сигналD*} $\subseteq$ (αP), { *проверитьD*} $\subseteq$ (αQ)

Для процессов – необходимо присутствие события *сигналD*, события *проверитьD* и *установитьD* при описании поведения процессов *P*, *Q* и *D* соответственно:

(*проверитьD* $\rightarrow$ *Q*) || (*сигналD* $\rightarrow$ *P*) || (*установитьD* $\rightarrow$ *D*)

Выводы

Предложены математические модели, основанные на теории взаимодействующих процессов Ч.Хоара, позволяющие описать орга-

низацию вычислений в кластерных системах с многоядерною архитектурою. Модели отображают двухуровневую организацию параллельных процессов в КСМА, а также систему взаимодействия процессов на основе посылки сообщений или общих переменных. Модель может быть использована для анализа параллельных процессов в КСМА, оптимизации взаимодействия процессов, связанной с передачей данных, а также выявления тупиковых ситуаций, возникающих при взаимодействии процессов в КСМА.

Список литературы

1. Хоар Ч. Взаимодействующие последовательные процессы: Пер с англ. – М.: Мир, 1989.
2. El - Rewini H, Lewis T., Distributed and Parallel Computing. – Manning Pub. Co., NY, 1998.
3. Burns A., Welling A. Real-Time Systems and Programming Languages, Third Edition, Addison-Wesley, NY, 2001.
4. Жуков І., Корочкін О. Паралельні та розподілені обчислення. – Корнійчук, К., 2006.