В Ы Ч И С Л И Т Е Л Ь Н Ы Е С И С Т Е М Ы
1974 год Выпуск 60

УДК 681.142.2

СРЕДСТВА ПРОГРАММИРОВАНИЯ СИСТЕМЫ МИНИМАКС

В.Г.Кербель, Ю.И.Колосова, В.Д.Корнеев, Н.Н.Миренков
Е.В.Щербаков

Описываются средства параллельного программирования для однородной вычислительной системы (ОВС) МИНИМАКС [1,2], которые являются расширением математического обеспечения машины М-6000. Основное внимание уделяется языкам для записи параллельных (р-) алгоритмов, средствам отладки р-программ и языку директив.

1. Я з ы к и д л я з а п и с и р - а л г о р и т м о в основываются на языках МНЕМОКОД, ФОРТРАН, АЛГОЛ. Методика крупноблочного распараллеливания [3] и распараллеливание по циклам [4] позволяют получать р-алгоритмы в виде совокупности взаимодействующих ветвей. Разработка р-программы для р-алгоритма в основном сводится [5] к разработке последовательной программы одной ветви. Особенность такой программы – обменные взаимодействия с другими ветвями [6]. Введение операторов (процедур), реализующих эти взаимодействия, в известные языки программирования расширяет их до языков для записи р-алгоритмов. Указанное расширение впервые было предложено в [7] и реализовано для ОВС "Минск-222" [8-10].

Рассматриваются пять типов системных взаимодействий: трансляционный и дифференцированный обмены, обмен сдвигом, обобщенный условный переход, ввод/вывод для системных внешних устройств.

Т р а н с л я ц и о н н ы й обмен – передача информации из одной ветви во все остальные:

$$BEXC(N1, N, A, B, N2)$$

где BEXC – Broadcasting EXChange; $N1$ – номер передающей ветви (число или идентификатор); A и B – идентификаторы передаваемого и принимающего массивов (простых переменных); N – количество передаваемых слов; $N2$ – целое положительное число (метка), отличающее данный оператор от остальных. (Для параллельных программ с неидентичными ветвями системные операторы, реализующие одно и то же обменное взаимодействие, должны иметь одинаковое значение $N2$ во всех ветвях.)

Дифференцированный обмен – передача информации из одной ветви в некоторые выделенные:

$$DEXC\ (N1, N, A, B, N2, Z1, Z2, \ldots, ZK),$$

где $DEXC$ – Defferent EXChange; $Z1, Z2, \ldots, ZK$ – номера принимающих ветвей.

Обмен сдвигом – передача информации из каждой ветви в соседнюю с большим (меньшим) номером:

$$MEXC(X, A, B, N2),$$

где MEXC – Moving EXChange; $X = 0$ – сдвиг в ветвь с большим номером, $X = 1$ – с меньшим номером.

Обобщенный условный переход – изменение естественного порядка реализации операторов во всех ветвях при выполнении обобщенного условия:

$$GCP\ (F, M, N2),$$

где GCP – General Condition Passage; F – идентификатор простой переменной, участвующей в выработке обобщенного признака: если значения F отрицательны во всех ветвях, то управление передается на метку M, иначе – на следующий оператор.

Ввод/вывод для системных внешних устройств – обмен информацией между элементарными машинами системы и системными внешними устройствами:

$$IN/OUT\ (X, N, C, L, N2),$$

где X – определяет устройство ввода/вывода и адрес внешней памяти; N – количество вводимых/выводимых слов (байтов); C – идентификатор вводимого/выводимого массива; Z – номер ветви, через которую осуществляется обмен; при $L = 0$ обмен реализуется всеми ветвями.

Помимо приведенных основных системных операторов выделяются еще два. Трансляционно-циклический обмен есть трансляционный обмен в цикле по всем ветвям:

$$CBEXC\ (N, A, B, N2).$$

Коллекторный обмен – передача информации из всех ветвей в одну выделенную:

$$CEXC\ (N1, N, A, B, N2).$$

Описанные операторы обменных взаимодействий реализуются на основе обращений к модулям диспетчера элементарной машины (ЭМ) [II]:

$$CALL\ EXEC(ICODE, ICON, \ldots).$$

$$ICODE = DEC \begin{cases} 1 - IN \\ 2 - OUT \\ 101 - BEXC \\ 102 - DEXC \\ 103 - MEXC \\ 104 - CBEXC \\ 105 - CEXC \\ 106 - GCP \end{cases},$$

$ICON$ – первый параметр из списка в операторе.

Модули, соответствующие операторам обменных взаимодействий, реализуют следующие основные функции:

1) проверку возможности использования системных рабочих каналов для обменных операций; проверка связана с блокированием обменов в подсистеме, если одна из её ЭМ занята операциями ввода/вывода для системных внешних устройств;

2) выработку обобщенного условия и выполнение обобщенного условного перехода (связанного с аварийными ситуациями в ветвях параллельной программы) перед выполнением данного обменного оператора;

3) выявление семантических ошибок, связанных с неправильным использованием операторов;

4) обменное взаимодействие, заданное в операторе.

Взаимодействия между ветвями параллельных программ выполняются в подсистемах режима параллельной обработки автономно

(относительно Старшего диспетчера режима параллельной обработки и Главного диспетчера [II]) по каналам, принадлежащим этим подсистемам. Управление каналами осуществляется с помощью операции "Настройка элементарной машины".

2. С р е д с т в а а н а л и з а и о т л а д к и р – п р о г р а м м базируется на "отладчике" элементарной машины, обеспечивающем в режиме диалога проверку программ, выработанных трансляторами с МНЕМОКОДа, ФОРТРАНа и АЛГОЛа. Средства ориентированы на проведение анализа и отладки р-программ на одной машине, хотя имеется возможность использования нескольких ЭМ [I2-I4].

Особенности функционирования средств анализа и отладки р-программ заключаются:

I) в проверке правильности взаимодействий ветвей р-программы;

2) в выдаче информации, помогающей пользователю совершенствовать р-программу в сторону максимального использования технических возможностей, представляемых системой.

О т л а д о ч н ы е д е й с т в и я распространяются на блоки р-ветви и всю р-программу.

I. Б л о к и р-программы, не содержащие системных операторов и параметров, зависящих от этих операторов, отлаживаются как обычная последовательная программа с помощью любого оператора "Отладчика".

2. Р – в е т в ь проверяется посредством "отладчика", расширенного новым блоком, который при появлении системного оператора может вывести на терминал тип этого оператора, данные, предназначенные для других р-ветвей, запрос на ввод той или иной информации.

3. Р – п р о г р а м м а отлаживается с помощью "Отладчика", расширенного блоком, регулирующим взаимодействие р-ветвей. Этот блок организует выполнение каждой р-ветви (до появления в них системного оператора). В результате выявляется набор операторов, характеризующих системное взаимодействие. Анализ этих взаимодействий выполняется по методике предложенной в [I3].

Предусмотрены два режима отладки р-программы: первый режим моделирует параллельный процесс, когда расширенный "Отладчик" и все р-ветви с их массивами находятся в оперативной памяти машины, второй – когда для запоминания состояния р-ветвей и их массивов используется внешняя память. Для каждого из режимов удобно иметь свой вариант "Отладчика", вызываемого по имени.

А н а л и з и р у ю щ и е д е й с т в и я рассматриваемых средств связаны с выдачей информации, используя которую можно совершенствовать р-программу и р-алгоритм [I2].

– Время простоев. Простои появляются из-за ожидания какой-либо р-ветвью нужных операндов, из-за образования очередей при обращении к одной и той же ЭМ и т.д. Информация о простоях позволяет по-другому распределять данные между ветвями, видо-изменять р-алгоритм.

– Время выполнения блока программы. Такая информация особенно важна для реализации счета в случаях конвейерных схем и неидентичных р-ветвей.

– Точность. Потеря точности вычислений связана с округлениями при выполнении операций с плавающей запятой. Алгоритм определения точности опирается на формулы, приведенные в [I2]. Такая информация облегчает обнаружение функций, на которых имеют место наибольшие потери точности вычислений.

– Объем оперативной памяти и объем вычислений. Указанная информация позволяет судить об активности использования памяти разными блоками программы.

3. Я з ы к д и р е к т и в представляет собой набор предложений для задания работ системе. Директивы могут вводиться с клавиатуры телетайпа или устройств группового ввода элементарных машин и позволяют:

– инициировать, приостанавливать, завершать и выбрасывать работы;

– переключать ЭМ с режима "клавиатуры" на режим "группового ввода" и обратно;

– запускать приостановленные программы;

– создавать и уничтожать файлы пользователей;

– редактировать и распечатывать файлы;

– инициировать трансляцию исходных программ;

– печатать даты и комментарии;

– узнавать состояние системы (разбиение системы на подсистемы, состояние дорожек диска, таблиц устройств ввода/вывода и т.п.);

– реализовывать "почтовую" связь между пользователями;

– переводить ЭМ (и внешние устройства) в режим профилактики;

– организовывать подсистемы и назначать графики их функционирования.

Пользователю доступны директивы, связанные со всеми видами работ, за исключением двух последних. Оператору (привилегированному пользователю) доступны любые директивы. Директивы имеют одинаковый формат независимо от устройства, с которого они вводятся, и подразделяются на две группы. Первая управляет заданиями, связанными с режимом автономной работы, вторая – заданиями, связанными с режимом параллельной обработки и дополнительными возможностями ОВС. Директивы первой группы являются стандартными. Они присущи фактически любой дисковой операционной системе, поэтому на них мы останавливаться не будем и перейдем к директивам второй группы.

Запрос на запуск р-программы:

$$PRUN, R, NAME, У, Z, T, P ,$$

где *PRUN* – **Parallel RUN** ; *R* – ранг р-программы; *NAME* – имя р-программы;

$У =$ { I – счёт с повышенной надежностью; 2 – счет с дублированием р-программы на подсистемах; 3 – простой счет;

$Z =$ { I – ввод р-программы с устройства группового ввода; 2 – размещение р-программы на дисковом файле по имени, указанном в данной директиве;

T – предполагаемое время счета; *P* – приоритет задачи.

К о м м е н т а р и й. Этой директивой пользователь заказывает счет р-программы. Выделяются три вида реализации, связанные с возможностью выхода машин из строя.

I. Счет с повышенной надежностью означает тестирование машин через некоторые интервалы времени и замену вышедших из строя машин (при небольших значениях *T* тестирование производится до и после счета).

2. Счет с дублированием на подсистемах означает сравнение через некоторые интервалы времени результатов одинаковых вычислений, выполненных в двух подсистемах. При несовпадении результатов машины тестируются и счет продолжается на основании информации, сохраненной с предыдущего успешно реализованного шага.

3. Простой счет означает вычисления на подсистеме ОВС без принятия специальных мер.

Подготовка задачи к первым двум реализациям производится пользователем в процессе диалога со специальным блоком программного обеспечения.

Снятие запроса на запуск р-программы:

$$ARUN, NAME, D ,$$

где *ARUN* – **Abort RUN**;

$D =$ { I – снятие заказа без сохранения результатов возможного счета; 2 – снятие заказа с сохранением результатов.

К о м м е н т а р и й. Этой директивой пользователь снимает запрос на запуск параллельной программы с указанным именем. Поступление *ARUN* прерывает счет поименованной программы.

Создание параллельного файла:

$$PSTOR, C, NAME, R, LOGICAL\ UNIT ,$$

где *PSTOR* – **Parallel STORe**; *C* – тип р-файла (например, исходная программа, объектная программа, двоичные данные и т.д.); *NAME* – имя организуемого р-файла; *R* – ранг р-файла; *LOGICAL UNIT* – логическое устройство (или сектор диска), с которого предполагается считать данные для р-файла.

К о м м е н т а р и й. Этой директивой пользователь создает на диске р-файл, представляющий собой совокупность файлов, предназначенных для ветвей р-программы.

Уничтожение параллельного файла:

$$PPURG, NAME ,$$

где *PPURG* – **Parallel PURGe**.

К о м м е н т а р и й. Этой директивой пользователь уничтожает на диске р-файл.

Организация графика работы подсистем:

$$WORK, LIST1, LIST2, LIST3, LIST4, t ,$$

где *t* – время действия директивы; *LIST1*, *LIST2*, *LIST3* – списки ЭМ, назначаемых соответственно для режимов автономной работы,

параллельной обработки и диспетчера [II]; *LIST4* – список ЭМ, принадлежащих списку *LIST* I и предназначенных для динамического перевода в режим параллельной обработки при выполнении одного из следующих условий:

$$\sum_i T_i' R_i > k \cdot t' \cdot L_2, \quad \max_i R_i > L_2,$$

где

$$T_i' = \begin{cases} T_i, & \text{если } T_i < t'; \\ t', & \text{если } T_i \geq t'; \end{cases}$$

L_2 – число ЭМ в *LIST2*; T_i – предполагаемое время, необходимое для завершения i-й задачи, R_i – её ранг; t' – остаток времени действия директивы; k – некоторое положительное число, меньшее I.

К о м м е н т а р и й. Этой директивой оператор разбивает систему на подсистемы и назначает им режим работы. По истечении времени t оператору выдается сигнальное сообщение, а также информация о наличии запросов на параллельные программы. После чего действие директивы продолжается с $t' = 0$, пока не будет введена новая директива *WORK*. ЭМ, не вошедшие в списки директивы, считаются находящимися в режиме профилактики.

Перевод машин в режим профилактики:

MDOWN, LIST, H,

где *MDOWN* – **Machine DOWN**; при H = I ЭМ списка *LIST* исключаются из системы для ремонта; при H = 2 – тестируются программными средствами.

К о м м е н т а р и й. Директива доступна только оператору. Помимо оператора, перевод ЭМ в режим профилактики может осуществлять Главный диспетчер.

Возвращение машины в систему после профилактики:

MUP, LIST, X1, Z1,

где *MUP* – **Machine UP**; $X1$ = I означает перевод ЭМ списка *LIST* в режим автономной работы; $X1$ = 2 – в режим параллельной обработки;

$Z1 =$ { I – возвращение ЭМ в систему после ремонта;
2 – возвращение после тестирования программными средствами.

К о м м е н т а р и й. Директива доступна только оператору.

Организация "почтовой связи".

SPOST, NAME, 'TEXT',

где *SPOST* – **System POST**; *NAME* – имя пользователя, для которого предназначен *TEXT*.

К о м м е н т а р и й. С помощью этой директивы пользователи обмениваются сообщениями. *TEXT* помещается в "почтовый ящик", откуда выдается пользователю автоматически при входе его в систему или по директиве:

GPOST, NAME,

где *GPOS* – **Get POST**.

Запрос о состоянии системы:

SSTAT Y1[Y2,Y3],

где *SSTAT* – **System STATus**; при $Y1$ = I выдаются сведения о наличии незагруженных машин; при $Y2$ = I – списки машин по каждому режиму и график их функционирования; при $Y3$ = I – данные о состоянии обобществленных (системных) устройств.

К о м м е н т а р и й. С помощью этой директивы пользователь узнает о разбиении системы на подсистемы, о наличии вышедших из строя машин, о максимальном ранге р-программы, которая может быть реализована в данной ситуации и т.д. Квадратные скобки в операторе означают, что оба или один из параметров могут отсутствовать.

Рассмотренные средства программирования вместе с методикой крупноблочного распараллеливания предоставляют широкие возможности для создания р-программ, обеспечивают преемственность основных методов последовательного программирования и могут служить основой математического обеспечения общего назначения при создании автоматизированных систем управления на базе многомашинных комплексов из мини-ЭВМ.

Литература

I. ВИНОКУРОВ В.Г., ДИМИТРИЕВ Ю.К., ЕВРЕИНОВ Э.В., КОСТЕЛЯНСКИЙ В.М., ЛЕХНОВА Г.М., МИРЕНКОВ Н.Н., РЕЗАНОВ В.В, ХОРОШЕВСКИЙ В.Г. Однородная вычислительная система из мини-машин. -В кн.: Вычислительные системы. Вып.51, Новосибирск, 1972, стр. 127-146.

2. МИРЕНКОВ Н.Н. МИНИМАКС - вычислительная система коллективного пользования. -Настоящий сборник, стр.II5-I28.

3. ЕВРЕИНОВ Э.В., КОСАРЕВ Ю.Г. Однородные универсальные вычислительные системы высокой производительности.Новосибирск, "Наука", I966.

4. КОСАРЕВ Ю.Г. Распараллеливание по циклам. -В кн.: Вычислительные системы. Вып.24, Новосибирск, I967, стр. 3-I9.

5. МИРЕНКОВ Н.Н. Параллельные алгоритмы для решения задач на однородных вычислительных системах. -В кн.: Вычислительные системы. Вып.57, Новосибирск, I973, стр. 3-34.

6. КОСАРЕВ Ю.Г. О схемах обмена между ветвями параллельных алгоритмов. -В кн.: Вычислительные системы. Вып.5I. Новосибирск, I972, стр. 70-75.

7. КОСАРЕВ Ю.Г. Об автоматизации программирования для однородных вычислительных систем. -В кн.: Труды I-ой Всесоюзной конференции по вычислительным системам. Вып.4.Новосибирск,I967, стр. 23-26.

8. ГОЛОВЯШКИНА Л.В., КОЛОСОВА Ю.И., КОСАРЕВ Ю.Г., МИРЕНКОВ Н.Н. Автоматизация программирования для системы на основе существующих трансляторов. -В кн.: Вычислительные системы. Вып. 30. Новосибирск, I968, стр. 63-69.

9. ГРИШАЕВА Н.К., КЕРБЕЛЬ В.Г., КОЛОСОВА Ю.И., КОНСТАНТИНОВ В.И., МИРЕНКОВ Н.Н. Язык параллельных алгоритмов. -В кн.: Вычислительные системы.Вып.57. Новосибирск,I973, стр. 33-54.

I0. ГРИШАЕВА Н.К., КЕРБЕЛЬ В.Г., КОЛОСОВА Ю.И., КОНСТАНТИНОВ В.И., МИРЕНКОВ Н.Н.Транслятор с языка параллельных алгоритмов. -Там же, стр. 55-73.

II. КЕРБЕЛЬ В.Г., КОРНЕЕВ В.Д.,МИРЕНКОВ Н.Н.,ЩЕРБАКОВ Е.В. Управляющая программа системы МИНИМАКС. -Настоящий сборник, стр. I29-I42.

I2. КОЛОСОВА Ю.И. Комплекс средств производства параллельных программ. -В кн.: Вычислительные системы. Вып.57. Новосибирск, I973, стр. 98-II4.

I3. КОЛОСОВА Ю.И., МИРЕНКОВ Н.Н. Исследование взаимодействий ветвей параллельных программ. -Там же, стр. II5-I24.

I3. КОЛОСОВА Ю.И. Об отладочных программах для вычислительной системы "Минск-222". -В кн.: Вычислительные системы.Вып.5I. Новосибирск, I972, стр. 76-8I.

Поступила в ред.-изд.отд.
9 июля I973 года