СРЕДСТВА РЕАЛИЗАЦИИ ПАРАЛЛЕЛЬНОСТИ В ЯЗЫКАХ ПРОГРАММИРОВАНИЯ

Дудник Л. О.
Кафедра ПМиИ, ДонГТУ
tolyan@r5.dgtu.donetsk.ua

Abstract

Dudnik L. Tools for realization of parallelism in programming languages. *The article describe a various tools of inter-process communication, such as semaphores, pipelines, message queues. The article describe tools for realization of parallelism for shared memory machines and nonshared memory machines. Shared memory machines support communication between parallel processes through shared variables whose access is controlled using primitives: semaphores, monitores and path expressions. Nonshared memory machines support another kind of inter-process communication - message passing. Moreover, the article describe a parallel programming systems for different types of parallel architectures.*

Введение

Идея распараллеливания вычислительных процессов привела к появлению многопроцессорных вычислительных машин, многомашинных комплексов и сетей, все шире используемых в различных областях науки и техники [1]. Для реализации параллельных процессов в этих системах потребовалось расширение возможностей программного обеспечения: языковых средств описания параллельных процессов, методов построения трансляторов, операционных систем и прикладных программ.

Параллельные процессы могут быть реализованы:

- на векторных и матричных процессорах, способных осуществлять параллельную синхронную обработку информации. К ним относятся ILLIAC-IV, GRAY-1, Cyber - 205 и др;
- в многопроцессорных вычислительных системах с общей оперативной памятью, имеющих несколько устройств управления и арифметико-логических устройств. Такие системы могут обрабатывать одновременно несколько потоков команд над различными данными. К ним относятся системы Sequent Symmetry, Convex, Cray 6400;
- на многомашинных комплексах и сетях, состоящих из многих вычислительных машин, объединенных общими каналами связи, по которым осуществляется обмен информацией (IBM-SP1/SP2, Parsytec GC, Cray T3D, nCUBE).

Разнообразие архитектур вычислительных систем обусловило и разнообразие средств реализации параллельности (Табл. 1).

Целью данной работы является обзор средств реализации параллельности в языках программирования, ориентированных на различные виды параллельной архитектуры. Рассмотрены системы программирования для архитектур следующих видов: векторных процессоров и мультипроцессорных систем с общей и с распределенной памятью.

1. Способы реализации параллельности

Так как архитектура параллельных вычислительных систем весьма разнообразна, различны методология программирования и средства, вводимые в язык, но все же можно выделить общие способы реализации параллельности в языках программирования [2].

Таблица 1

Влияние архитектуры вычислительных систем на языковые средства параллельного программирования

Типы вычислительных систем	Операции над массивами (синхронная параллельная обработка)	Описание процессов	Синхронизирующие примитивы типа "семафор"	Инициализация, завершение процессов и их синхронизация	Синхронизация доступа к разделяемым ресурсам	Обмен сообщениями и синхронизация сообщений
Векторные и матричные процессоры	используются	—	—	—	—	—
Многопроцессорные системы с общей памятью	—	используется	используются	используется	используется	—
Многопроцессорные системы с раздельной памятью и локальные сети	—	используется	могут быть использованы, но обычно не используются	используется	используется	используется

Использование библиотек. При появлении новой архитектуры, параллельность вначале часто реализуется с помощью специальной библиотеки, содержащей подпрограммы поддержки параллельности, а программы на Фортране или Си должны содержать обращения к подпрограммам этой библиотеки. Преимуществом использования библиотек является неизменность исходного языка и компилятора. В дальнейшем наиболее оправдавшие себя средства могут быть включены в языковые расширения. Но этот способ имеет и недостатки. Использование библиотек менее удобно для программиста, чем использование языковых расширений, так как список параметров обычно велик и не соответствует программированию на языке высокого уровня. Так как исходная программа содержит только обращения к библиотечным подпрограммам, то компилятор не может выполнить некоторые виды оптимизации. Недостатком этого подхода является и то, что не исключены дополнительные накладные расходы на обращения к подпрограммам библиотек.

Макрорасширения. В ряде реализаций используются макрорасширения Фортрана и Си. Директивы макрорасширения позволяют пользователю специфицировать некоторые виды параллельности. Макрорасширения либо оформляются в виде специальных комментариев, либо записываются заглавными буквами, а основной текст записывается строчными буквами. Директивы-макрорасширения обычно переводятся препроцессором в операторы Фортрана или Си с вызовами специальных подпрограмм, которые и реализуют параллельную обработку. Описанный подход не очень удобен при программировании на языке высокого уровня. Кроме того, использование макрорасширений добавляет промежуточный код, что осложняет отладку программ.

Расширения в языке. Более удобным и более естественным для программиста представляется подход, основанный на введении расширений непосредственно в язык. Разработчики расширений для уже существующих языков программирования обычно сталкиваются с двумя проблемами: выбор новых средств и отображение выбранных средств в конкретную языковую среду.

Довольно часто расширения в языке ориентированы на конкретную архитектуру, что делает параллельные программы немобильными.

2. Языковые средства реализации синхронных параллельных процессов и обработки массивов

Для решения большинства задач требуется интенсивная обработка массивов. Эти задачи могут быть успешно решены с помощью векторизующих компиляторов. Большинство векторизующих компиляторов разработаны для Фортрана, но имеются векторизующие компиляторы и для Си и для Паскаля [3]. Существует несколько вариантов расширения языка Фортран. Все они представляют собой параллельно-последовательный метод организации параллельных программ, основу которого составляет последовательность параллельных операторов или параллельно-исполняемые последовательные ветви (процессы), называемые синхронными параллельными процессами [4]. Если из такой программы удалить дополнительные средства распараллеливания, то программа становится обычной последовательной. В качестве описания параллельно-последовательных структур применяются различные средства, например, parbegin . . . parend, cobegin . . . coend, отмечающие начало и конец параллельного сегмента [5]. При этом обычно налагается требование того, что параллельный участок закрывается только тогда, когда все его ветви завершились. Каждая из ветвей может содержать свои участки параллельности, таким образом они образуют произвольную иерархию [6]. Для задания ветвления используются операторы fork и join. В языке IVTRAN, разработанного для системы ILLIAC-IV, задание параллелизма осуществляется с помощью оператора DO m FOR ALL (i1..in)/... .

Одним из первых и наиболее популярным расширением Фортрана был язык VECTRAN [7]. Массивы в нем описываются как параллельные объекты данных заданной размерности, и любое обращение к массиву подразумевает весь массив. Векторное выражение может быть присвоено строке или столбцу массива. В языке Фортран 90, тоже являющемся расширением языка Фортран, имеется большой набор стандартных функций для работы с массивами [5]. Операции над массивами, введенные в этот язык, неявно задают внутренний параллелизм действий над компонентами массивов. Функции (COUNT, MINVAL, MAXVAL, PRODUCT, SUM) выполняют арифметические, логические, счетные операции над массивами.

На основе описанных выше средств реализации параллельности можно выделить две особенности параллельно-последовательного метода программирования: отсутствие каких-либо взаимодействий между параллельными фрагментами; императивность управления вычислениями, состоящая в том, что после выполнения каждого параллельного оператора или сегмента явно и точно указывается, какие операторы или сегменты следует выполнять на следующем шаге.

3. Средства синхронизации доступа к разделяемым ресурсам в мультипроцессорных системах с общей памятью

Существует много важных задач, которые содержат ряд алгоритмов, допускающих параллельную обработку, но плохо векторизуемых. В таких случаях лучше распараллелить алгоритм на несколько процессов, выполняемых на различных процессорах. Эти процессы могут обмениваться данными или работать с общими данными, как, например, в мультипроцессорных системах с общей памятью. Обмен информацией в системах с общей памятью может быть выполнен традиционными средствами типа COMMON или передачей по параметрам. Для

синхронизации доступа к разделяемым ресурсам в системах с общей памятью чаще всего используются три типа языковых конструкций: *семафоры, мониторы и управляющие выражения* [8].

Простейшим языковым механизмом синхронизации является *семафор* [1], введенный Дейкстрой. Семафор - неотрицательная переменная, предназначенная специально для синхронизации. Начальное значение семафора указывается при его объявлении (по умолчанию - единица). Единственные операции, допустимые над семафорами - процедуры P и V. С формальной точки зрения, семафор - это очередь со счетчиком, который уменьшается и увеличивается на единицу при помощи операций P и V. Несмотря на универсальность, семафор обладает и некоторыми недостатками, а именно - плохой структуризацией и выразительностью. При работе с ним высока вероятность ошибки, которую трудно найти и исправить.

Другим наиболее распространенным средством синхронизации является *монитор*. Монитор - это языковая конструкция, состоящая из структур данных и набора подпрограмм. В каждый момент времени внутри монитора может находиться только один процесс. Если внутри монитора находится какой-либо процесс, то все остальные процессы, обращающиеся к этому монитору задерживаются и ставятся в очередь. Монитор осуществляет управление доступом к разделяемым ресурсам посредством использования переменных типа "событие". Хотя монитор и является средством синхронизации более высокого уровня, чем семафор, но он обладает и существенным недостатком: синхронизация с помощью монитора сильно снижает производительность системы.

Другой подход синхронизации доступа к разделяемым ресурсам реализован *в управляющих выражениях (*Path expressions) [10]. В управляющих выражениях предусмотрена возможность организации различных дисциплин синхронизации доступа к процедурам.

4. Средства реализации параллельности в многопроцессорных системах с распределенной памятью

Для систем с распределенной памятью используется механизм задач и передачи сообщений. Исходная программа разбивается на задачи, которые могут выполняться параллельно на разных процессорах. Связь между задачами осуществляется с помощью передачи сообщений [11,12]. Во многих системах (PVM, PARMACS и др.) механизм задач реализуется с помощью библиотечных вызовов. Накоплен большой опыт использования подобных систем. Разработан проект MPI (Message Passing Interface), представляющий собой стандартный набор библиотечных интерфейсов для передачи сообщений [2]. MPI включает большой набор средств, в число которых входят операции передачи сообщений от одного источника (процессора) к другому. MPI поддерживает все типы данных, имеющиеся в Фортране и Си, в нем имеются и собственные типы данных. Хотя MPI предлагает большой набор средств, все же, для прикладных программистов более предпочтительными являются системы, основанные на расширении языков. Расширения Фортрана реализованы, например, в языке Фортран Ryan-McFarland. Фортран RYAN-MCFARLAND был разработан в Oregon Graduate Center для машины Intel iPSC Concurrent Computer d5, имеющей 16 процессорных узлов, соединенных в топологии гиперкуба (рис.1).

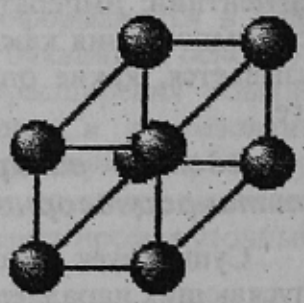

Рис. 1 Гиперкуб из 8 узлов

Программа, написанная на этом языке, содержит один начальный процесс и может содержать другие процессы. Обмен информацией между процессами осуществляется с помощью передачи сообщений. Предусмотрено два вида передачи сообщений: с блокировкой и без блокировки. В язык введены операторы передачи сообщений SEND и RECEIVE. Языковые средства этого языка обеспечивают более высокий уровень программирования, так как в операторах передачи сообщений можно указывать произвольный список передаваемых данных, аналогичный списку ввода/вывода Фортрана, что гораздо удобнее, чем средства, имеющиеся в системах, основанных на библиотечном подходе, в которых требуется, чтобы либо все передаваемые данные находились в непрерывной области памяти, длина которой часто указывается в байтах. Выборка сообщений из очереди в системах с распределенной памятью происходит по типу сообщения или по приоритету. В общем случае операторы посылки и приема сообщений содержат соответствующие служебные слова языка, адреса отправителя и адресата и список переменных, значения которых пересылаются от одного процесса другому.

5. Средства реализации параллельности в языках, ориентированных на различные виды параллельной архитектуры

Одной из важных характеристик языков параллельного программирования является мобильность. Кроме рассмотренных выше способов, предпринимались попытки создания языковых средств, ориентированных одновременно и на системы с общей памятью и на системы с распределенной памятью. В некоторых системах в язык включались как средства реализации механизма задач и передачи сообщений, так и конструкции, задающие параллельное выполнение итераций цикла, параллельное выполнение ветвей и средства синхронизации (семафоры, события, барьеры и т.п.). Так, например, программы, написанные на языке Charm могут выполняться на машинах с или без совместно используемой памятью. Язык Charm был создан в 1995 г. в университете Иллинойс [13]. Charm является синтаксическим расширением языка С. Основная единица параллельного программирования в CHARM программах - chare. Chare экземпляры могут посылать сообщения друг другу используя вызовы SendMsg и получают сообщения используя вызов RecvMsg. В языке Charm можно определять глобальные общие переменные, значения которых могут изменяться процессами.

Наиболее перспективным в настоящее время является подход к расширениям последовательных языков программирования, ориентированный на разбиение данных. В язык вводятся средства, позволяющие пользователю специфицировать размещение и распределение данных по процессорам, всю остальную работу по распараллеливанию программы выполняет компилятор. Предлагаемые средства могут быть использованы как для машин с векторными процессорами, так и для разных типов многопроцессорных систем. Так, например, в 1992 г. был разработан язык HPF (High Performance Fortran), являющийся расширением Фортрана 90 [14]. Помимо векторных операций Фортрана 90, HPF содержит директивы для описания способов разбиения данных между параллельно работающими процессорами и средства явного указания параллельности. Директивы HPF имеют вид комментария, который начинается с символов !HPF$. С помощью директивы TEMPLATE объявляется некоторое абстрактное пространство - шаблон. Директива ALIGN специфицирует элементы различных массивов, которые должны быть распределены на одном и том же процессоре. Директива DISTRIBUTE специфицирует соответствие элементов шаблона и процессоров, т.е. определяет способ распределения по процессорам. Из средств явного указания параллельности можно отметить оператор FORALL, допускающий большее разнообразие видов секций, чем в Фортране 90, и

директиву INDEPENDENT, которая указывает компилятору, что итерации в следующем DO-цикле или операторе FORALL могут выполняться независимо.

Заключение

Проведенный анализ средств реализации параллельности в языках программирования позволяет сделать следующие выводы:

- для векторных процессоров используется последовательно-параллельный метод программирования, основу которого составляет последовательность параллельных операторов или параллельно-исполняемые последовательные ветви;
- в языках программирования, ориентированных на системы с общей памятью, для синхронизации доступа к разделяемым ресурсам чаще всего используются три типа языковых конструкций: семафоры, мониторы и управляющие выражения;
- в языках программирования, ориентированных на системы с распределенной памятью, используется модель обмена сообщениями. В настоящее время широко используется проект MPI (Message Passing Interface), представляющий собой стандартный набор библиотечных интерфейсов для передачи сообщений;
- в языки программирования, ориентированные одновременно и на системы с общей памятью и на системы с распределенной памятью, включаются как средства реализации механизма задач и передачи сообщений, так и конструкции, задающие параллельное выполнение итераций цикла, параллельное выполнение ветвей и средства синхронизации. Наиболее перспективным в настоящее время является создание языков программирования, ориентированных на разбиение данных.

Литература

1. Томас Бройнль "Параллельное программирование". Начальный курс. перевод В. А. Святного. Киев. 1997 г., 200 с.

2. Открытые системы. № 2, 1995 г.

3. "Программирование на параллельных вычислительных системах". ред. Бэбб Р., М.: 1991 г.

4. Вальковский В.А., Малышкин В.Э. Элементы современного программирования и суперЭВМ. Новосибирск. Наука, 1990. С. 54-72.

5."Элементы параллельного программирования". ред. Котов В.Е., М.: Радио и связь, 1983, С. 68-89.

6. Миренков Н. Н. Параллельное программирование для многомодульных вычислительных систем. М.: Радио и связь, 1989.

7."Алгоритмы, математическое обеспечение и архитектура многопроцессорных вычислительных систем". ред. Ершов А. П., М.: Наука, 1982.

8. Барский А. Б. Параллельные процессы в вычислительных системах. ., М.: Радио и связь, 1990.

9. Хокни Р., Джесссхоул К. Параллельные ЭВМ. Перевод с англ. М.: Радио и связь, 1986, С. 209-215, 240-242.

10. Трахтенгерц Э. А. Программное обеспечение параллельных процессов. М.: Наука, 1987. С. 17 - 34.

11. Поспелов Д.А. Введение в теорию вычислительных систем. М.: Советское радио, 1982.

12. "Системы параллельной обработки" ред. Д. Ивенс. М.: Мир, 1985.

13. Charm: http://charm.cs.uius.edu

14. High Performance Fortran: http://www.erc.msstate.edu/hpff/report.html