CZU 004.42

ПАРАЛЛЕЛЬНОЕ ПРОГРАММИРОВАНИЕ И ПРОБЛЕМА РАЗРАБОТКИ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

Анастасия МАЙДАЧЕНКО, *студентка,факультет реальных наук, экономики и окружающей среды, Бельцкий государственный университет имени Алеку Руссо*
Научный руководитель: **Корина НЕГАРА,** *доктор, конференциар*

Rezumat: *Acest articol se axează pe noțiunea de „programare paralelă”. Prin programare paralelă se înțelege controlul a două sau mai multe operații care sunt executate simultan și fiecare dintre operații implică o serie de instrucțiuni. Acest lucru poate fi realizat de un singur computer: fie echipându-l cu mai multe unități de control, fie permițând partajarea timpului unei unități de control între mai multe activități. A doua variantă pare a avea un interes practic mai mare.*

Cuvinte-cheie: *paralelism, programare paralelă, problema dezvoltării, programe paralele, paralelizarea sarcinilor, flux, sincronizare, sisteme de calcul, multitasking, controlul fluxului.*

Параллельное программирование по праву считается наиболее перспективным, многообещающим и востребованным направлением в области разработки

программного обеспечения. Основная причина - эффективное выполнение кода за счет снижения количества затрачиваемого времени. Как следствие, параллельное программирование часто масштабируется с размером проблемы и, таким образом, может производить более крупные расчеты. Соответственно можно максимизировать вычислительную мощность многоядерных процессоров и повысить производительность, ведь это программирование выходит за рамки, налагаемые последовательными вычислениями, которые часто ограничиваются физическими и практическими факторами.

Огромное увеличение вычислительной мощности, которым мы наслаждаемся на протяжении десятилетий, стало основой многих самых значительных достижений в таких разнообразных областях, как наука, интернет и развлечения. Например, расшифровка генома человека, все более точные медицинские изображения, удивительно быстрый и точный поиск в Интернете и все более реалистичные компьютерные игры были бы невозможны без этих улучшений. По мере того как наши вычислительные мощности увеличиваются, количество проблем, которые мы можем рассмотреть, также растет. Приведем несколько примеров.

- *Моделирование климата*. Чтобы лучше понять изменение климата, нам необходимо создавать более точные компьютерные модели, модели, которые включают взаимодействие между атмосферой, океанами, твердой землей и ледяными шапками на полюсах. Мы также должны иметь возможность проводить подробные исследования того, как различные вмешательства могут повлиять на глобальный климат.
- *Сворачивание белков*. Считается, что неправильно свернутые белки могут быть вовлечены в такие заболевания, как болезнь Паркинсона и Альцгеймера, но наша способность изучать конфигурации сложных молекул, таких, как белки, сильно ограничена нашими текущими вычислительными возможностями.
- *Открытие лекарств*. Есть много способов, использующих увеличенную вычислительную мощность в исследованиях новых медицинских методов лечения. Например, существует множество лекарств, которые эффективны при лечении относительно небольшой части людей, страдающих каким-либо заболеванием. Возможно, мы сможем разработать альтернативные методы лечения путем тщательного анализа геномов людей, для которых известное лечение неэффективно. Однако это потребует обширного компьютерного анализа геномов.
- *Энергетические исследования*. Повышенная вычислительная мощность позволит программировать гораздо более сложные модели технологий, таких, как ветряные турбины, солнечные элементы и инкубаторы. Эти программы могут предоставить информацию, необходимую для создания гораздо более эффективных источников чистой энергии.

Любая вычислительная система, насколько бы мощной она ни была, обладает ограниченными ресурсами. Оперативная память, устройство чтения и записи информации, процессор – все это имеет некоторые пределы. Из-за

этого возникает острая необходимость использовать эти ресурсы наиболее эффективно, и именно параллельная работа программ является одним из ключевых факторов, позволяющих выполнить поставленную задачу.

Мозг человека воспринимает информацию за счет анализа временных характеристик потока импульсов, проходящего по миллиарду новых клеток. Наш мозг мощнее, чем Google, и работает лучше всех роботов вместе взятых. Мы можем моментально перебирать в уме огромное число событий и эмоций. Мы способны немедленно узнавать родителей, супругов, детей, друзей или домашних животных, днем и ночью, в любом положении. Даже наиболее продвинутые компьютерные системы с трудом справляются с подобными задачами. Нам несложно одновременно заниматься несколькими делами, например, доставать телефон и проверять время, складывать оригами во время разговора. Робот же, способный справиться с такой простой комбинацией действий, еще не создан.

Справедливости ради стоит отметить, что речь идет не о настоящем параллельном процессинге, а о быстром переключении с одной задачи на другую, о грамотном распределении внимания и расстановке приоритетов. Но сама вероятность того, что компьютер может приблизиться по функциональным возможностям к человеческим уже давно не кажется такой уж и невозможной. Стремительное развитие технологий в последние десятилетия дает надежду на воплощение футуристического будущего.

Проблема создания высокопроизводительных вычислительных систем относится к числу наиболее сложных научно-технических задач современности и ее разрешение возможно только при всемерной концентрации усилий многих талантливых ученых и конструкторов, предполагает использование всех последних достижений науки и техники и требует значительных финансовых инвестиций. Тем не менее достигнутые в последнее время достижения в этой области впечатляют. Так, в рамках принятой в США в 1995 г. Программы "Ускоренной стратегической компьютерной инициативы" (Accelerated Strategic Computing Initiative - ASCI) была поставлена задача увеличения производительности супер-ЭВМ в 3 раза каждые 18 месяцев и достижение уровня производительности в 100 триллионов операций в секунду (100 терафлопс) в 2004 г. [1].

Данные за 2020 год таковы: самым мощным суперкомпьютером в июне 2020 года стала японская система Fugaku. Она имеет мощность 415,5 петафлопса, что в 2,8 раза выше, чем у предыдущего лидера Summit, который теперь расположился на втором месте. Новый компьютер также имеет высокую, хотя и не рекордную (девятое место в списке), энергоэффективность: она составляет 14,665 гигафлопс на ватт потребляемой мощности. Топ 5 самых мощных суперкомпьютеров к июню 2020 года представлены в таблице 1 [2].

Применение супер-ЭВМ налагает особые требования на вновь создаваемые программные средства, обеспечивающие надежную и экономичную реализацию алгоритма при решении прикладных задач. Эффективность использования супер-ЭВМ проявляется при создании сложных исследовательских комплексов и экспертных систем.

Таблица 1. *Самые мощные суперкомпьютеры к июню 2020 года*

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	Суперкомпьютер Fugaku - Supercomputer Fugaku, A64FX 48C 2.2GHZ, Tofu interconnect D, Fujitsu RIKEN Center for Computational Science Japan	72990072	415530.0	513854.7	28335
2	Summit - IBM Power System AC922, IBM POWER9 22C 3.07GHZ, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM DOE/SC/Oak Ridge National Laboratory United States	2414592	148600.0	200794.9	10096
3	Sierra - IBM Power System AC922, IBM POWER9 22C 3.1GHZ, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, IBM/ NVIDIA / Mellanox DOE/NNSA/LLNL United States	1572480	94640.0	125712.0	7438
4	Sunway TaihuLight - Sunway MPP, Sunway SW26010 260C 1.45GHZ, Sunway, NRCPC National Supercompuling Center in Wuxr China	10649600	93014.6	125439.9	15371
5	Tianhe-2A - TH-IVB-FEP Cluster, Intel Xeon E5-2692v2 12C 2.2GHZ, TH Express-2, Matrix-2000, NUDT National Super Computer Center in Guangzhou-China	4981760	61444.5	100678.7	18482

Примечания к Таблице 1.
Rank – порядковый номер в списке Top500.
System – название (тип) компьютера, указанное поставщиком.
Cores – количество вычислительных ядер.
Rmax – максимальная полученная производительность по Linpack (TFlop/s).
Rpeak – теоретическая пиковая производительность (TFlop/s).
Power – электропотребление системы в киловаттах (kW).

Написать эффективную параллельную программу намного труднее, чем последовательную. Создание программного обеспечения для параллельных компьютеров – это центральная проблема суперкомпьютерных вычислений [3].

Частично проблема выбора оптимального числа параллельных ветвей в соответствии с критерием минимума суммарных затрат времени может быть

решена в автоматах генерации параллельной программы. Частный случай разрешения этой проблемы для вычислительных систем с MIMD архитектурой рассмотрен в статье Костенко В.А. "К вопросу об оценке оптимальной степени параллелизма" [4].

Эффективность использования многопроцессорных вычислительных систем в значительной мере определяется качеством прикладных параллельных программ. Программа считается эффективной, когда во время ее выполнения загружены все процессоры, выделенные под процессы. Но практически это не реализуемо.

Несмотря на большое разнообразие конструкций, которые можно найти в современных электронных цифровых компьютерах, есть одна основная черта, которая является общей для всех существующих машин. Это тот факт, что они обычно продолжали выполнение ряда операций, каждая из которых указывается инструкцией, по одной за раз. Этим они отличаются от некоторых более ранних машин, таких как Hollerith Tabulator и ENIAC (Goldstine and Goldstine, 1946), конструкция которых позволяла производить сравнительно большое количество фактически независимых операций добавления или переноса данных одновременно. С появлением современных компьютеров с хранимыми программами от этой гибкости отказались в пользу выполнения операций по одной за раз.

Однако утверждение о том, что современные компьютеры выполняют только одну операцию за один раз, заслуживает внимательного изучения. Ведь, на самом деле, это не совсем так. Операции, которые ограничены одной специальной частью машины и занимают время, длительное по сравнению со средним временем для операции, часто разрешается выполнять независимо от выполнения других операций. Самый распространенный пример – это работа механических устройств, подключенных к компьютеру, таких, как устройства ввода и вывода.

Однако, прежде чем перейти к тому, как именно осуществляется параллельная работа программ, необходимо разобрать базовые вещи. Таковыми являются понятия: процесс и поток. Они довольно похожи, ведь и то, и то является не чем иным, как определенной последовательностью команд некоторой программы. Но, тем не менее, они имеют существенные различия. Так, процессы изолированы друг от друга, они используют разные адресные пространства и не могут получить доступа к переменным и структурам других процессов. В свою очередь, потоки выполняются в рамках некоторого процесса, используют одно адресное пространство и спокойно взаимодействуют между собой, при этом никоим образом не мешая работе друг друга. Именно благодаря потокам можно заметно повысить скорость выполнения некоторого приложения, просто разбив его на несколько последовательных потоков [5].

Отметим, что идеальная параллельная программа должна обладать следующими свойствами:

1. Длины параллельно выполняемых ветвей равны между собой.
2. Полностью исключены простои из-за ожидания данных, передачи управления и возникновения конфликтов при использовании общих ресурсов.

3. Обмен данными полностью совмещен с вычислениями.

Основные этапы самого процесса адаптации программ к архитектуре параллельных компьютеров, а также описание задач, возникающих на каждом из этих этапов, представлены в статье Антонова А.С. "Эффективная адаптация последовательных программ для современных векторно-конвейерных и массивно- параллельных супер-ЭВМ" [6]. На некоторые из проблем, с которыми столкнулись авторы проведенного исследования, хотелось бы обратить особое внимание. Среди них:

1. исследование общей структуры программы;
2. выделение основного вычислительного ядра, локализация ввода-вывода;
3. определение потенциального параллелизма фрагмента;
4. выделение последовательных фрагментов вычислений и попытка использования альтернативных алгоритмов для таких фрагментов;
5. определение и минимизация точек перераспределения данных;
6. преобразование традиционных циклов под параллельные алгоритмы;
7. минимизация числа и объема временных массивов для оптимизации работы с кэш-памятью;
8. переход от исходной программы, работающей с полными массивами, к программе, обрабатывающей только локальную порцию, распределенную на процессор: изменение размеров массивов и соответствующее преобразование текста программы.

Отметим, что решение этих задач позволяет осуществить эффективный перенос последовательной программы на параллельную архитектуру.

Процесс разработки параллельной программы очень длителен и трудоемок, несмотря на то, что, как правило, на момент ее создания уже имеется реализация ее "последовательного" аналога. Обычно программа разрабатывается на машине с одной архитектурой, а ее практическое применение производится на другой, с отличной от первой топологии, но при этом более мощной. Такой подход позволяет экономить машинное время на более мощных супер-ЭВМ, число которых на порядок меньше по сравнению с более дешевыми моделями суперкомпьютеров.

Преимущества параллельных вычислений заключаются в том, что компьютеры могут выполнять код более эффективно, что может сэкономить время и деньги, сортируя «большие данные» быстрее, чем когда-либо. Параллельное программирование также может решать более сложные задачи, увеличивая объем ресурсов. Это помогает с приложениями, начиная от улучшения солнечной энергии и заканчивая изменением работы финансовой отрасли.

1. Параллельные вычисления моделируют реальный мир

Окружающий нас мир не последователен. Вещи не происходят по одному, ожидая завершения одного события до начала следующего. Чтобы получить данные о погоде, дорожном движении, финансах, промышленности, сельском хозяйстве, океанах, ледяных шапках и здравоохранении, нам нужны параллельные компьютеры.

2. Экономят время и деньги

За счет экономии времени параллельные вычисления удешевляют работу. Более эффективное использование ресурсов может показаться незначительным в малом масштабе. Но когда мы расширяем систему до миллиардов операций - например, банковского программного обеспечения, - мы видим огромную экономию средств.

3. Решает более сложные или большие проблемы.

Вычислительная техника становится все более зрелой. Благодаря ИИ и большим данным одно веб-приложение может обрабатывать миллионы транзакций каждую секунду. Кроме того, «грандиозные задачи», такие, как обеспечение безопасности киберпространства или обеспечение доступности солнечной энергии, потребуют петафлопс вычислительных ресурсов. Мы добьемся этого быстрее с помощью параллельных вычислений.

4. Использует удаленные ресурсы

Люди создают 2,5 квинтиллиона байтов информации в день. Это число 25 с 29 нулями. Мы не можем вычислить эти цифры. Или можем? При параллельной обработке несколько компьютеров с несколькими ядрами каждый могут обрабатывать в режиме реального времени во много раз больше данных, чем последовательные компьютеры, работающие по отдельности.

Проще говоря, параллельные вычисления являются частью многоядерных процессоров в наших телефонах и ноутбуках, которые обеспечивают их эффективную работу. В самом сложном случае это потрясающие 200000 ядер суперкомпьютера American Summit, которые помогают нам решать проблемы в генетике, раке, окружающей среде и даже моделировать работу вселенной. Идея заключается в том, что компьютер может разбить проблему на части и одновременно работать над ними. По мере роста объемов данных в нашем мире параллельные вычисления будут идти в ногу, чтобы помочь нам разобраться в этом. В этом и заключается необходимость их изучения.

Библиография:

1. *FLOPS* [online] Доступен по адресу: https://ru.wikipedia.org/wiki/FLOPS [Дата обращения 28.03.2021].
2. *Рейтинг суперкомпьютеров мира Top500* [online] Доступен по адресу: https://www.tadviser.ru/index.php/Статья:Рейтинг_суперкомпьютеров_мира_Top500#2020:_.D0.92_.D1.82.D0.BE.D0.BF-500_.D1.81.D1.83.D0.BF.D0.B5.D1.80.D0.BA.D0.BE.D0.BC.D0.BF.D1.8C.D1.8E.D1.82.D0.B5.D1.80.D0.BE.D0.B2_.D0.BE.D1.81.D1.82.D0.B0.D0.BB.D0.B8.D1.81.D1.8C_2_.D1.81.D0.B8.D1.81.D1.82.D0.B5.D0.BC.D1.8B_.D0.B8.D0.B7_.D0.A0.D0.BE.D1.81.D1.81.D0.B8.D0.B8 [Дата обращения 28.03.2021].
3. ВОЕВОДИН, В.В., *Суперкомпьютеры: вчера, сегодня, завтра*. // Сборник научно-популярных статей Российская наука на заре нового века. Под редакцией академика В.П. Скулачева. М.: научный мир, 2001. С. 475-483.
4. КОСТЕНКО, В.А., *К вопросу об оценке оптимальной степени параллелизма.* // Программирование. 1995, 4, с. 24-28.
5. ИВАНОВ, К. К., *Основы параллельной работы программ* / К. К. Иванов, С. А. Раздобудько, Р. И. Ковалев. – Текст: непосредственный // Молодой

ученый. – 2017. – № 7 (141). – С. 13-15. – URL: https://moluch.ru/archive/141/39880/ [Дата обращения: 28.03.2021].
6. АНТОНОВ, А.С., ВОЕВОДИН Вл.В., *Эффективная адаптация последовательных программ для современных векторно-конвейерных и массивно-параллельных супер-ЭВМ*. // Программирование. 1996, 4, с. 37-51.