ПРЕПЯТСТВИЯ К НЕОГРАНИЧЕННОМУ МАСШТАБИРОВАНИЮ ПАРАЛЛЕЛЬНЫХ СИСТЕМ

О.А. Юфрякова

Дальнейшее серьезное увеличение быстродействия однопроцессорных компьютеров только за счет совершенствования элементной базы становится принципиально невозможным. Этому препятствуют три основных физических барьера.

1. Световой барьер. Время срабатывания элементов микропроцессора оказывается сравнимым со временем прохождения сигналов по проводникам: t = l/c, где l – длина проводника, c – скорость света. Когда нарушаются неравенства d < ct или dn < c, где: d - линейный размер процессора, c - скорость света, t - длительность такта, n - тактовая частота, тогда происходит рассинхронизация работы удаленных частей микропоцессора. Поэтому миниатюризация электронных компонентов является не прихотью, а жесткой необходимостью, при конструировании более быстрых микропроцессоров. Линейные размеры микропроцессоров уменьшаются. Intel переходит на технологию 3-D Tri-Gate [6], обеспечивающую еще более плотную трехмерную упаковку транзисторов в их микрочипах.
2. Тепловой барьер. Уменьшение размеров микропроцессора при сохранении потребляемой мощности приводит к увеличению плотности потока выделяемого тепла. Следовательно, чтобы избежать перегрева, необходимо снижать энергопотребление микропроцессора и принимать дополнительные меры для отвода тепла. Это мы и наблюдаем в развитии современных массовых электронных устройств.
3. Квантовый барьер. На достигнутых нанометровых размерах элементов микропроцессоров начинают проявляться квантовые эффекты. В этих масштабах вещество не может считаться непрерывной средой, а молекулы точечными объектами. Эти соображения и туннельный квантовый эффект приводят к невозможности создания сколь угодно тонких изоляторов. Когда толщина изолятора сопоставима с длиной де Бройлевской волны электрона, с высокой вероятностью электрон проникнет сквозь изолятор. Поэтому слишком близко различные проводники электронного устройства находиться не могут во избежание их нестабильной работы. Кроме того, имеющаяся фоновая радиации (потоки частиц) также разрушает стабильную работу таких миниатюрных устройств. Экранироваться от этой радиации чрезвычайно сложно. Таким образом, наличие вышеперечисленных барьеров принципиально ограничивает наращивание мощности одного микропроцессора. Можно наблюдать, как за последние пять лет резко упал темп увеличения тактовой частоты серийных микропроцессоров от Intel и ее конкурентов. Производительность процессоров наращивается за счет увеличения количества процессорных ядер, то есть за счет распараллеливания. Значит, дальнейшее существенное увеличение производительности вычислительных систем (ВС) возможно только за счет увеличения числа элементарных процессоров и разделения вычислительной работы между ними. Причем, световой барьер требует, чтобы у каждого такого процессора был свой тактовый генератор (свои часы) и эти генераторы синхронизировались бы как синхронизируются часы в теории относительности. Таким образом, в достаточно мощной ВС время должно быть локальным для каждого процессора. Каждый процессор выполняет свою часть работы независимо от остальных процессоров до определенного момента, а затем обменивается данными с другими процессорами. Иными словами, достаточно объемные вычисления должны быть параллельными и асинхронными. Архитектура суперкомпьютера должна позволять работать в этом режиме. Кажется, что эти требования к ВС достаточны для преодоления упомянутых барьеров. В самом деле, взяв достаточно много процессоров, мы выполним работу любого объема в приемлемое время. Но, увеличивая число процессоров, мы сталкиваемся с новым барьером.
4. Системный барьер. Когда мы удваиваем число процессоров в ВС, мы ожидаем, что ее вычислительная мощность также вырастет примерно вдвое. Так и происходит в системах с малым числом процессоров и хорошей операционной системой, качественно поддерживающей распараллеливание, на параллельных алгоритмах линейной сложности. Однако даже хорошая ОС и даже на одном процессоре значительную часть его ресурсов тратит не на решение прикладной задачи, а на решение системных задач, таких как «сборка мусора», проверка правильности выполнения операций, синхронизация работы различных процессов ОС и многих других. Это обстоятельство усугубляется с ростом числа процессоров: все большая часть ресурсов ВС начинает тратиться на решение системных задач и, прежде всего, на согласование работы различных процессоров и обмен данными между ними. При неудачной архитектуре ВС и не лучшем алгоритме решения задачи с ростом числа процессоров производительность ВС на этой задаче может парадоксальным образом снижаться [4].

Кроме того, согласно законам Амдала, изменения в проценте занятости процессора влекут значительные потери в производительности по сравнению с максимально возможной. Что это действительно так, говорит опыт применения мультипроцессорных систем для решения практических задач: даже небольшие отклонения от оптимального распараллеливания влекут за собой потери в теоретически достижимой производительности, особенно, если число процессоров велико. Это самая серьезная проблема в области

параллельного программирования.

Чтобы преодолеть системный барьер, то есть создать ВС произвольно большой мощности, точнее неограниченно масштабируемую ВС, мы должны наложить дополнительные ограничения на архитектуру ВС, на алгоритмы управления ВС (на уровне ОС) и на алгоритмы решения прикладной вычислительной задачи.

Под архитектурой ВС здесь понимается набор ее основных функциональных элементов и оснащенный граф каналов передачи данных и управления, соединяющих функциональные элементы.

Анализ оптимальных по разным критериям архитектур параллельных ВС проведен в основном в Новосибирской научной школе, созданной Э. В. Евреиновым. Им в 60-тых годах прошлого века была предложена принципиально новая архитектура ВС – известная ныне как архитектура однородной вычислительной системы (ОВС) [1]. Она базируется на следующих 3 принципах: параллельность операций, переменность логической структуры, конструктивная однородность элементов и связей между ними.

Первый принцип предполагает, что всякая трудоемкая распараллеливаемая задача может быть представлена в виде связанных между собой однотипных простых подзадач и для такой задачи может быть предложен параллельный алгоритм, допускающий ее эффективное решение.

Второй принцип предполагает, что процесс решения сложной задачи может быть представлен некоторой структурной моделью, включающей в себя подзадачи и связи между ними. Это означает, что для каждой сложной задачи можно предложить оптимальную структуру из обрабатывающих элементов, связанных между собой соответствующим образом.

Третий принцип предполагает, что все простые подзадачи получаются примерно одинаковыми по объему вычислений и связаны между собой одинаковыми схемами обмена данных. Это означает, что система для решения сложной задачи может быть построена из одинаковых обрабатывающих элементов, связанных между собой одинаковым образом, то есть коммутационный граф ВС должен быть однородным. Наиболее известными примерами однородных графов являются полный граф, гиперкуб и тор.

Но для обеспечения неограниченной масштабируемости и преодоления системного барьера однородности ВС недостаточно. Дело в том, что при отображении логической структуры (виртуальной) ВС оптимальной для решения прикладной задачи на постоянную физическую структуру (реальной) ВС соседние логические процессоры могут отобразиться на далекие физические процессоры, что с учетом асинхронности процесса вычислений приведет к возникновению существенных задержек при передаче данных. Поэтому при программировании придется учитывать не только задержки в передаче данных между узлами логической структуры, но и эти, которые учесть относительно просто в виду однородности ВС, реальные задержки, которые возникли при таком отображении. А если иметь в виду, что это отображение осуществляется операционной системой и результат его заранее не известен, то, очевидно, что эффективное программирование в таких ситуациях невозможно.

Чтобы исправить ситуацию, дополним требования, предъявленные к ВС, еще одним требованием – принципом локальности, сформулированным Воробьевым В.А.:

«Выполнение очередного шага вычислений должно зависеть только от состояний тех компонент системы, которые находятся от процессора, выполняющей этот шаг, на расстоянии не более, чем заданное l, и задерживают вычисления на время не более, чем заданное t. Эти параметры не должны зависеть от n – числа процессоров в системе. Таким образом, архитектура ВС, ОС ВС и алгоритм решения задачи должны быть локальными, то есть одинаково зависящими только от состояния небольшой пространственно-временной окрестности данного процессора выполняющего данный массовый шаг алгоритма решения задачи» [2].

Воробьев В.А. также проанализировал однородные графы, удовлетворяющие принципу локальности и локально вкладываемые в планарный граф ([2], [3]). В частности, гиперкуб и полный граф не локальны в смысле Воробьева.

Алгоритм вычислений можно представить в виде графа, вершины которого соответствуют отдельным шагам вычислений, а дуги – информационным связям между шагами. Согласно принципу локальности, этот граф должен быть таким, чтобы при вложении его в граф логической ВС, в котором вершина – виртуальный процессор, а ребро – линия связи между такими процессорами, соблюдалось следующее правило: соседи в алгоритмическом графе должны оказаться в графе виртуальной ВС на расстоянии не большем, чем некоторое заранее заданное для системы число l. Аналогичное правило должно выполнятся при отображении графа виртуальной ВС на коммутационный граф физической ВС средствами операционной системы.

Наличие единого тактового генератора ВС противоречит принципу локальности, который требует, чтобы порядок срабатывания процессоров ВС определялся вычисляемыми логическими условиями, только исходя из информации от соседних процессоров.

Аналогично, принципу локальности противоречит использование общей памяти при массовых параллельных вычислениях.

Еще одним аспектом принципа локальности является требование отсутствия централизованного управления массовыми вычислениями. Ни один процессор не имеет полной информации о состоянии системы в режиме массовых вычислений. Процессоры принимают решения на основе только локальной информации лишь из окрестности ограниченного радиуса l по пространству и радиуса t по времени.

Последовательное применение принципа локальности требует разработки новых архитектурных подходов в построении вычислительных систем, последовательной разработки локально-параллельной алгоритмики, включая модификацию теории оптимальных алгоритмов и теории сложности, алгоритмов для локально-параллельных систем, разработку операционных систем и языков программирования. Поясним сказанное.

Современная теория вычислений (от теории алгоритмов до схемотехники) базируется на тех же предположениях о природе пространства и времени, что и классическая физика:

Вычислительное событие, состоящее в присваивании значения некоторой переменной, мгновенно меняет состояние всей системы.

Порядок вычислительных событий не зависит от наблюдателя.

Сигналы распространяются мгновенно, поэтому задержки при пересылке данных не учитываются.

На этом основании рассматриваемую теорию вычислений можно назвать классической, в том же смысле, как и всю классическую физику.

Как и механика, классическая теория вычислений неадекватна при сверхвысоких тактовых частотах вычислительных устройств: оценки вычислительной сложности алгоритмов неверны, а многие задачи (в частности, требующие глобальной синхронизации событий) теряют ясный физический смысл. По аналогии с физикой следует построить релятивистскую теорию вычислений, учитывающую конечность скорости передачи сигнала и время доставки информации от одного процессора к другому по коммутационному графу ВС.

Что касается ОС для неограниченно масштабируемых ВС, то и в них должен быть внедрен принцип локальности. Все операции ОС следует разделить на локальные, удовлетворяющие принципу локальности, и не локальные, которые либо требуют знания информации о состоянии всей системы или значительной ее части, либо изменяют состояние такой части. Не локальные операции должны быть единичными такие, например, как запуск ВС, создание виртуальной ВС и ее инициализация, завершение работы. Массовые вычислительные операции и операции передачи данных должны быть локальными. Кроме того, отображение виртуальной логической структуры ВС на физическую структуру ВС также должно быть локальным. Иными словами, при таком отображении, процессоры близкие логически должны отображаться в реальные процессоры близкие физически. Таким образом, ОС должна «знать» схему коммутации своих элементов и время доставки данных от одних элементов к другим. На практике это означает, что понятие локальности операций нужно внедрить в стандарты MPI и PVM. Это возможно в виду открытости этих стандартов. Сложнее осуществить требование локальности при отображении виртуальных ВС на реальные, так как оно реализуется разработчиками суперкомпьютерных ОС и требует исчерпывающих знаний интерфейсов аппаратуры физической части ВС.

С этой точки зрения, существующие архитектуры ВС и вышеупомянутые теории алгоритмов и вычислений можно назвать классическими. Классические ВС не способны преодолеть световой и системный барьеры.

Так, первое место из списка суперкомпьютеров TOP500 2010 занимает Tianhe (Китай) мощностью 2566 Tf/s и с 186368 процессорами, второе – Jaguar (США) мощностью 1759 Tf/s с 224162 процессорами, третье – Nebulae (Китай) мощностью 1271 Tf/s с 120640 процессорами. В этом списке последние годы наблюдаются любопытные изменения: если раньше рекордсмены, как правило, имели и рекордное число процессоров, то теперь рекордсмен по числу процессоров (294912 процессоров PowerPC производительностью 3.4 GFlops) компьютер JUGENE занимает лишь девятое место по производительности на тестах LINPACK. А чемпион по производительности – китайская машина Tianhe имеет 186368 процессоров Intel Xeon производительностью 11.72 GFlops. Мы видим, что число процессоров более не растет, что свидетельствует о том, что рекордные вычислительные системы приблизились к описанным выше барьерам. Рост производительности суперкомпьютеров достигается за счет увеличения локальности архитектуры и переноса все большего количества процессорных элементов (ядер) на один чип [5].

Релятивистская ВС – это однородная, асинхронная ВС с распределенными памятью и временем, аппаратное и программное обеспечение которой удовлетворяет принципу локальности. В таких ВС применяются протоколы обменов, характерные для сетей ЭВМ, но аппаратно реализованные как в транспьютерных сетях, поскольку элементарный обмен данными между процессорами и будет самой дорогой массовой операцией.

Из классических ВС к релятивистским ближе асинхронные массовые MIMD архитектуры с распределенной памятью, поддерживающие стандарт MPI. С другой стороны к ним приближаются однородные суперкластеры из ПК. В таких кластерах межпроцессорный обмен данными очень медленный, поэтому они автоматически обладают многими чертами релятивистской ВС. Но именно медленность коммуникаций и не позволит достичь им наивысшей производительности.

Именно релятивистские ВС являются теоретически неограниченно масштабируемыми и способными преодолеть оба барьера.

ЛИТЕРАТУРА:

1. Евреинов Э.В., Хорошевский В.Г. Однородные вычислительные системы. // Новосибирск. Наука,

1978. – 320 с.
2. В.А.Воробьев. Теоретические основы построения однородных вычислительных систем на неразрезных процессорных матрицах. Дисс. … докт. техн. наук /ИПУ РАН/.– Москва, 1999. – 350 с.
3. В.А.Воробьев. Теория однородных вычислительных систем: однородные структуры. – Архангельск: Поморский госуниверсистет, 2001. – 96 с.
4. Юфрякова О.А. Оценки оптимального числа процессоров ОВС для наискорейшего исполнения параллельных алгоритмов. – Высокопризводительные параллельные вычисления на кластерных системах. Материалы шестого Международного научно-практического семинара. Том 2. / Под ред. Проф. Р.Г.Стронгина. Санкт-Петербург, 2007. – 255 с.
5. Рейтинговый список 500 самых производительных суперкомпьютеров – http://www.top500.org/list/2010/11/
6. Официальный сайт корпорации Intel. – http://www.intel.com/