К МОДЕЛИ УПРАВЛЕНИЯ ПАРАЛЛЕЛЬНЫМИ И РАСПРЕДЕЛЕННЫМИ ВЫЧИСЛЕНИЯМИ В БОЛЬШИХ ВЫЧИСЛИТЕЛЬНЫХ СРЕДАХ

Ю.С. Затуливетер

Институт проблем управления им. В.А. Трапезникова РАН

Введение

Опережающее развитие СБИС и массовых компьютерно-сетевых технологий привело к формированию глобальной компьютерной среды (ГКС). Она образует вычислительную среду со сверхбольшим количеством связанных сетями вычислительных узлов в виде мобильных и стационарных компьютерных устройств различных классов – от интеллектуальных датчиков, смартфонов и ПК до суперкомпьютеров. Количество связанных сетями компьютерных устройств составляет ~10^9–10^{10}.

Эта быстро растущая среда обладает неограниченно наращиваемым функциональным и вычислительным потенциалом. В настоящее время совокупные вычислительные ресурсы только ПК, входящих в ГКС в количестве около 2 млрд., представлены "суммарной" оперативной памятью более 1-2 Эбайт (10^{18} байт) и сотнями экзабайт долговременной памяти, их "пиковая" производительность – более 1-2 Эфлопс (10^{18} флопс). Быстро растут сети мобильной связи (более 4,5 млрд. абонентов). В стандартах 3G и 4G они также интегрируются в ГКС, увеличивая её вычислительный потенциал и привнося новые функциональные возможности, связанные с перемещениями узлов (навигация, универсальные платёжные "карты", разнообразные сенсоры, мониторинг и многое др.). Важнейшим компонентом ГКС становятся петакомпьютеры и, в недалёкой перспективе, экзакомпьютеры. Через 4-10 лет число вычислительных узлов (ядер) в каждом экзакомпьютере будет достигать ~10^8–10^9, что станет сравнимым с общим количеством вычислительных узлов в существующих сетях.

Сверхбыстрый рост ГКС осуществляется одновременно в двух противоположных направлениях. Первое – экспоненциальный рост количества узлов в сетях (*N*glob_net, см. рис.1) и неограниченное наращивание пространственных/потребительских ареалов их экспансии *на макроуровнях*. Второе – неограниченное наращивание количества узлов вычислительных сред *на микроуровнях* за счёт увеличения пространственной плотности размещения узлов, которое обеспечивается благодаря опережающему прогрессу полупроводниковых СБИС-технологий (экспоненциальный рост количества узлов *N*micro, см. рис.1).

Глобальная компьютерная среда де-факто стала носителем всемирного информационного пространства с метрикой глобальной сильносвязности "всё влияет на всё и сразу". Оно оказывает беспрецедентное по масштабам и глубине воздействие на мировую социосистему [1,2]. Прежде всего, функционирование и развитие мировой социосистемы и её частей в условиях глобальной сильносвязности сопровождается экспоненциальным ростом потоков и объёмов распределённой информации, однако всё большая её часть остаётся непереработанной. Циркулируя и накапливаясь в ГКС, своевременно непереработанная информация перестаёт отражать текущее состояние практически неограниченно растущего числа не учитываемых связей между компонентами социосистемы в целом и её частей.

Накопление своевременно неперерабатываемой информации в беспрецедентных условиях глобальной сильносвязности создаёт эффект неконтролируемого роста информационных шумов. Нарушение балансов между производимой и потребляемой (своевременно перерабатываемой) информацией привело глобальному кризису "перепроизводства" информации. Отсюда снижение управляемости мировой социосистемы в целом, о чем свидетельствует непрерывная череда и нарастание амплитуды кризисов существующих социально-экономических моделей развития, сформировавшихся до появления глобально сильносвязного пространства.

Одна из главных причин невозможности своевременной переработки растущих потоков и объёмов информации, накапливаемой в ГКС – её крайняя системная разрозненность из-за непрерывного воспроизводства структурной разнородности своих вычислительных и информационных ресурсов.

Для своевременной и полномасштабной переработки экспоненты роста информации необходима системная консолидация экспоненты роста ресурсов ГКС в едином алгоритмическом пространстве распределённых и параллельных вычислений. Такое пространство сделает возможным полномасштабную функциональную интеграцию практически неограниченного совокупного вычислительного потенциала ГКС в целях своевременной переработки растущих потоков и объёмов информации путём формирования в ней сколь угодно больших вычислительных систем, функционирующих в едином сильносвязном пространстве глобально распределённой информации.

Виды параллелизма ГКС

По мере экспоненциального роста количества узлов ГКС пропорционально растёт параллелизм вычислительных сред. В многом нерешённые постнеймановские проблемы параллелизма (ещё недавно

"элитарно-экзотические"), в условиях глобальной сильносвязности и массового производства компьютеров, программ и услуг становятся главной "головной болью" компьютерной индустрии ближайшего будущего.

Сети общего назначения в большей своей части составлены из универсальных компьютеров с микропроцессорными архитектурами, построенными на основе модели Дж. фон Неймана. Эта модель на логическом и инженерном уровнях аксиоматически регламентирует архитектурные принципы реализации последовательных вычислений ("команда-за-командой"). Де-факто она стала единым логическим стандартом массового производства последовательных компьютеров и программ.

Наряду с классическими компьютерами в сетях используются компьютеры с многопроцессорными архитектурами, которые выходят за рамки полномочий классической аксиоматики, т.к. состоят из многих одновременно (параллельно) работающих процессоров и блоков памяти, связываемых коммутационными средствами. Многопроцессорные компьютеры строятся для достижения производительности, недоступной для последовательных компьютеров. Они составляют различные классы суперкомпьютеров.

Таким образом, в сетях реализуется два вида параллелизма – межкомпьютерный и внутрикомпьютерный. Они отличаются и по способам воплощения, и по назначению.

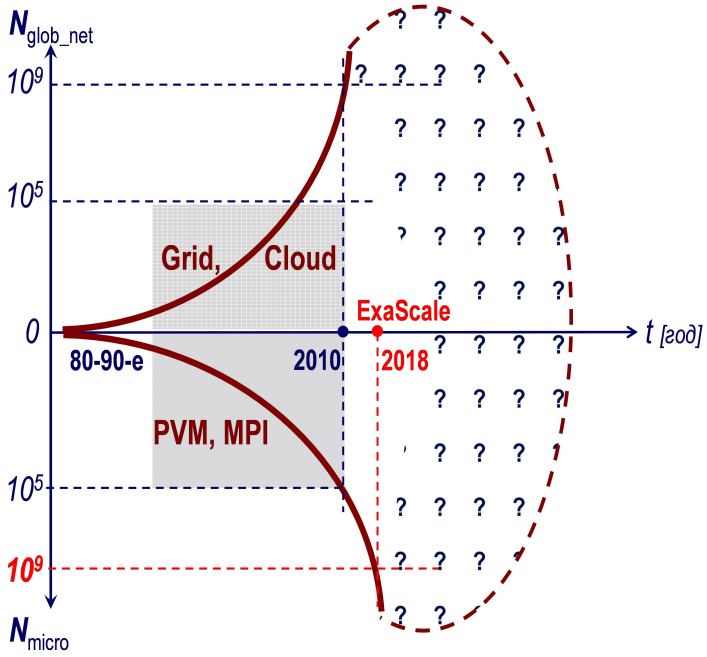


Рис.1. К единому пространству распределённых и параллельных вычислений

На рис.1 показаны тенденции экспоненциального роста межкомпьютерного параллелизма (верхний квадрант) и внутрикомпьютерного (нижний квадрант).

Межкомпьютерный параллелизм в глобальных сетях реализуется в виде пространственно распределённых вычислительных сред с неограниченно наращиваемыми множествами связанных между собой вычислительных узлов, разнесённых на большие расстояния. Взаимодействия между узлами осуществляется посредством сетевого оборудования и стека протоколов TCP/IP. Увеличение количества узлов позволяет неограниченно расширять пространственные ареалы сетей и/или увеличивать плотность расположения узлов в ограниченных ареалах. Назначение сетей – формирование среды накопления цифровой информации в узлах и передачи между ними, формирование коммуникационной среды информационных взаимодействий в субъект-объектных системах, построение управляющих и расчётных систем обработки распределённой информации. Сети обеспечивают формирование сильносвязных информационных пространств, которые служат основой для построения и интеграции распределённых сколь угодно больших субъект-объектных систем разнообразного назначения, придавая им свойство целостности. *Межкомпьютерный (сетевой) параллелизм можно рассматривать как системоформирующий. Его развитие направлено на максимально достижимое расширение системоформирующих ареалов сетей при ограниченной снизу плотности их заполнения узлами*.

Внутрикомпьютерный параллелизм реализуется в многопроцессорных системах виде максимально компактных универсально программируемых вычислительных сред. Количество узлов (процессоров, ядер) при этом наращивается путём уменьшения их размеров и расстояний между ними. Снижение размеров узлов и расстояний между ними позволяет увеличивать плотность размещения узлов в пространстве и, соответственно, скорости их взаимодействия, а также снижать энергопотребление.

С увеличением плотности растёт параллелизм и удельная производительность вычислений в расчёте на единицу пространства и энергопотребления. Благодаря опережающему прогрессу СБИС-технологий глубокого нанометрового диапазона (45-22нм и менее) компьютеры с многопроцессорными архитектурами в настоящее

время реализуются в однокристальном исполнении. Количество процессоров (ядер) в них достигает сотен и тысяч, что позволяет с одного чипа снимать производительность 2 Тфлопс и более. *Внутрикомпьютерный параллелизм можно рассматривать как вычислительный. Развитие многопроцессорных систем осуществляется в целях повышения вычислительной производительности посредством максимально достижимого увеличения плотности заполнения узлами единицы пространства.*

Параллелизм ГКС (рис.1) на межкомпьютерном уровне (сетевой) уже достиг количества узлов (компьютеров) $N_{glob\_net} \sim 10^9 - 10^{10}$, а на внутрикомпьютерном уровне число связанных узлов (ядер) N_{micro} через 4-10 лет в каждом экзакомпьютере достигнет сопоставимых значений – $10^8 - 10^9$.

Модели распределённых и параллельных вычислений выходят далеко за рамки классической аксиоматики универсальной модели последовательных вычислений. До настоящего времени используется большое количество разных моделей распределённых и параллельных вычислений. Некоторые из них нашли индустриальное воплощение и составили основу растущего рынка сетевых и параллельных вычислений.

О распределённых системах

Со второй половины 90-х годов для реализации распределённых вычислений в ресурсах глобальных сетей разрабатывались Grid-системы, а с середины 00-х по настоящее время активно применяются Cloud-технологии (см. рис.1).

Grid-решения представляют собой, как правило, узкопрофильные системы корпоративного назначения. Их размеры составляют от десятков до тысяч вычислительных узлов (с включением больших хранилищ данных). Они позволяют на уровне заданий распараллеливать исполнение вычислительных сервисов и обеспечивать более эффективное использование сетевых ресурсов. В течение десятилетия они заняли практически все свои рыночные ниши и, в основном, достигли пределов своего развития. Одна из наиболее масштабных Grid-систем применяется в ЦЕРН для моделирования и обработки результатов экспериментов в области физики элементарных частиц на Большом адронном коллайдере.

Cloud-технологии в настоящее время находят всё более широкое применение. Они строятся в сетевой архитектуре "Клиент-Сервер", в которой серверные центры обработки данных реализуются как кластеры ("облака") из множества серверов, связанных сетью в системно интегрированную вычислительную среду. Такие решения нацелены на расширение рынка предоставления клиентам через сети отдалённых алгоритмических услуг, скомпонованных в виде тех или иных сервисов по обработке данных, на основе заранее разработанных "алгоритмических библиотек". Как правило, такие системы нацеливаются на обеспечение интенсивных потоков запросов, требующих вычислительных ресурсов высокой производительности. Серверные "облака" могут объединять большое, но априори ограниченное, количество мощных серверов. К наиболее крупным относятся "облака", объединяющие десятки тысяч серверов, которые функционируют в Google и Amazon.

Рассмотренные виды систем распределённой обработки данных индустриального уровня ориентированы на функциональную интеграцию разнородных вычислительных и информационных ресурсов сетей. В основе каждого из рассмотренных подходов лежат изначально несовместимые между собой индустриальные стандарты различных конкурирующих групп производителей. Успешно решая отдельные классы задач распределённой обработки с привлечением априори ограниченного количества вычислительных узлов, непосредственно осуществляющих обработку, такие системы принципиально непригодны для полномасштабной интеграции совокупных ресурсов сколь угодно больших сетей.

Фундаментальным барьером к дальнейшему расширению масштабов функциональной интеграции сетей путём построения распределённых систем становится разнородность вычислительных и информационных ресурсов, требующая для их интеграции решения задач многовариантных композиций, которые имеют комбинаторную сложность. С увеличением количества узлов для преодоления опережающего роста комбинаторной сложности необходимы практически неограниченные затраты средств и времени.

Перечислим некоторые принципиальные ограничения, возникающие при наращивании размеров распределённых систем в разнородных сетях:

- преодоление при создании систем комбинаторных барьеров сложности в ходе системной и функциональной интеграции вычислительных ресурсов;
- обеспечение кибербезопасности в условиях крайне высокой уязвимости многослойного программного обеспечения;
- невозможность реконфигурации систем для адаптации к большим и быстрым изменениям внешней среды.

О многопроцессорных вычислениях

Проблематика параллельных вычислений и компьютеров с многопроцессорными архитектурами, как самостоятельное направление в компьютерных науках, сформировалась в 60-х годах.

В отличие от последовательных вычислений, которые на логическом и инженерном уровнях аксиоматически регламентированы классической моделью универсальных вычислений Дж. фон Неймана, параллелизм предполагает множественность одновременного исполнения операций. При этом допустимый состав множеств исполняемых на каждом шаге действий в условиях машинной среды многовариантен. В связи с этим возникает две, характерные для параллелизма, задачи:

- определение допустимого, с точки зрения корректности вычислений, состава множеств операций, исполняемых на каждом шаге;
- из допустимых вариантов состава множеств выявление таких, которые обеспечивают как можно более высокую эффективность исполнения в той или иной вычислительной среде (по наборам таких критериев как время, объёмы памяти, надёжность, энергопотребление, стоимость и др.).

Многокритериальные проблемы параллелизма отличаются высоким уровнем сложности. Математическая особенность таких проблем в том, что они формулируются в многовариантных пространствах, т.е. и здесь для нахождения решений требуется преодоление комбинаторной сложности.

Модели параллельных вычислений и соответствующие им архитектуры начали активно разрабатываться в 60-70-е годы. За прошедшие десятилетия наработано большое количество моделей параллельных вычислений. Одни создавались как абстрактные формализмы и конструкции вне требований практической реализуемости. Другие имели жёсткую привязку к тем или иным частным структурным, архитектурным или техническим решениям, что, во-первых, предельно сужало диапазоны их использование и, во-вторых, в большинстве случаев не отвечало требованиям практической реализуемости компьютеров и программ в массовых тиражах.

До конца 80-х суперкомпьютерные многопроцессорные системы разрабатывались и применялись, в основном, как "штучные" изделия и поставлялись с индивидуальными средствами программирования. Суперкомпьютерные поколения многопроцессорных систем 90-х обретают индустриальные инструменты программирования параллельных вычислений в виде технологий PVM и MPI. Это весьма трудоёмкий инструментарий, который позволяет программистам работать с многовариантными способами представления вычислительных задач и их отображениями в универсально программируемую многопроцессорную среду. Трудоёмкость создания параллельных программ многократно возрастает из-за высокой сложности поиска эффективных решений среди обилия неприемлемых вариантов. В силу высокой стоимости суперкомпьютерных многопроцессорных систем и относительно малого их распространения существенное снижение производительности труда программистов до сих пор удавалось в приемлемых уровнях компенсировать подготовкой программистов высокой квалификации.

Технологии PVM и MPI и их клоны до сих пор остаются (рис.1) основным стандартом программирования многопроцессорных систем. Однако в ходе гонки суперкомпьютеров за производительность за счёт увеличения параллелизма (числа процессоров, ядер) проблемы производства эффективных программ обостряются. Разрабатываемые в рамках тематики ExaScale экзакомпьютеры будут представлять собой сверхбольшие универсально программируемые вычислительные среды, с числом ядер 10^8 – 10^9. Программирование таких сред с применением существующих технологий, в которых многовариантное управление параллелизмом вычислений возложено на программистов, становится нереальным.

Новые факторы и проблемы

До сих пор разработки и исследования моделей распределённых и параллельных вычислений (включая многопроцессорные архитектуры) развивались как отдельные направления [3,4]. В отсутствие общей модели распределённых и параллельных вычислений в каждом из них сосуществуют многие взаимоисключающие подходы в виде профильных решений. Во многом это объясняется различиями существующих многопроцессорных и сетевых сред, а также принципов организации вычислений в них.

Новые требования к массовости параллелизма вычислительных сред актуализируют поиски общих методов и универсальных моделей, объединяющих способы рассмотрения и решения задач организации параллельных и распределённых вычислений. Наряду с теоретическими аспектами они должны включать в себя эффективные архитектуры [5] и принципы их программирования, отвечающие требованиям индустриального производства программ для высокопараллельных систем [6].

Одна из главных проблем формирования и развития сверхбольших сетевых и вычислительных сред – структурная разнородность аппаратных платформ и форм представления данных, программ процессов и систем. Разнородность, являясь причиной комбинаторная сложности проблем интеграции, при больших размерах становится фундаментальным барьером, преодоление которого технологическими методами невозможно. Из этого следует, что решение возникающих проблем качественного развития ГКС в рамках существующих моделей распределённых и параллельных вычислений в условиях разнородности становится стратегически бесперспективным. Игнорируя фактор комбинаторной сложности разнородных сред, известные модели параллельных и распределённых вычислений не учитывают тенденции сверхбыстрого роста их размеров.

Глобальные масштабы сетей и задач глубокой переработки распределённой информации в ГКС требуют новых подходов к решению проблем построения распределённых и параллельных систем обработки данных в сколь угодно больших средах. Свойство глобальной сильносвязности требует формирования в ГКС бесшовно программируемого алгоритмического пространства распределённых и параллельных вычислений с единым полем распределённой информации [7,8]. Оно позволит скрыть разнородность машинных сред и, тем самым, устранить барьеры комбинаторной сложности, что снимет верхние ограничения на размеры вычислительных сред.

К обобщению классической модели универсальных вычислений

Первоочередные проблемы на пути к единому алгоритмическому пространству – выявление и устранение первопричин непрерывного воспроизводства разнородности и построение обобщённой модели распределённых и параллельных вычислений, которая регламентирует структурные формы представления данных и программ и распространит свойство универсальной программируемости с внутренних ресурсов компьютеров на ресурсы сколь угодно больших сетей. В перспективе такая модель должна стать универсальной основой для реализации бесшовного программирования вычислений в распределённых (сетевых) и высокопараллельных (многопроцессорных) средах.

Сети общего назначения в большей своей части составлены из универсальных компьютеров с микропроцессорными архитектурами, построенными на основе модели Дж. фон Неймана, которая представляет собой классическую аксиоматику универсальных машинных вычислений и несёт в себе два принципиальных ограничения [1,9,10]:

- свойство универсальной программируемости реализуется только во внутренних ресурсах изолированных компьютеров, что не позволяет регулярным образом распространять свойство "бесшовной" программируемости на совокупные ресурсы компьютеров связанных сетями;
- отсутствует универсальная регламентация способов инженерного воплощения аппаратных средств и структурных форм представления информации в памяти, что является причиной непрерывного воспроизводства разнородности компьютерных архитектур и форм представления данных, программ, процессов и систем. Отсюда произрастают фундаментальные причины чрезвычайной разнородности ГКС и комбинаторной сложности функциональной интеграции её ресурсов.

В [1,9,10] представлено обобщение модели фон Неймана на основе компьютерного исчисления древовидных структур, которое позволило:

- ввести математически замкнутые формы представления структурно-сложной информации и, тем самым, устранить избыточные степени свободы управления вычислениями, скрытые в классических постулатах управления вычислениями посредством манипуляций машинными ресурсами, которые открывают неконтролируемые пути непрерывного воспроизводства разнородных форм представления данных и программ;
- распространить свойство универсальной программируемости с внутрикомпьютерных ресурсов на ресурсы сколь угодно больших сетей.

К модели управления параллелизмом в вычислительных задачах

Далее излагаются принципы построения автоматной модели, которая позволяет с единых позиций рассматривать управление распределёнными и параллельными вычислениями в режиме динамического распараллеливания на основе принципов Data Flow [11]. Особенность модели – независимость вычислительной сложности алгоритма управления от размерности задач, определяемой общим количеством вычислительных операций.

Модель имеет два уровня. Первый – представление вычислительных задач в виде информационных графов. Второй – автоматное представление структуры управления вычислениями с динамическим распараллеливанием.

Представление задач в виде информационных графов

Предполагается, что исходная задача задаётся в виде двудольного информационного графа, в котором направленные дуги связывают вершины двух разных типов – операторные и объектные. Операторные вершины представляют вычислительные действия, объектные – переменные, которые являются аргументами или значениями операторных. Подобная форма представление задач относится к непроцедурным и применяется в моделях Data Flow [11,12].

Информационный граф – двудольный ориентированный граф $\mathbf{G}=<\mathbf{A}\cup\mathbf{U}, \mathbf{S}>$, где $\mathbf{A}=\{a_1, a_2, \dots\}$ – множество операторных вершин, $\mathbf{U}=\{u_1,u_2,\dots\}$ – множество объектных вершин, $\mathbf{S}= \mathbf{S}(\mathbf{A, U})\cup\mathbf{S}(\mathbf{U, A})$ – множество ориентированных дуг: $\mathbf{S}(\mathbf{A, U})$ из операторных вершин к объектным, $\mathbf{S}(\mathbf{U, A})$ – из объектных к операторным.

Операторным вершинам $\forall a\in\mathbf{A}$ и объектным $\forall u\in\ \mathbf{U}$ ставится в соответствие типы вершин: $b(a)\in\mathbf{B}$ и $d(u)\in\mathbf{D}$, соответственно, где $\mathbf{B}=\{b_1, b_2, \dots, b_m\}$– множество базисных операторов (операций), $\mathbf{D}=\{d_1, d_2, \dots, d_n\}$ – множество базисных типов данных. Дуги $\mathbf{S}(\mathbf{U, A})$ определяют вхождение переменных в качестве аргументов, дуги $\mathbf{S}(\mathbf{A, U})$ – передачу вычисленных значений (выходных данных) каждого оператора.

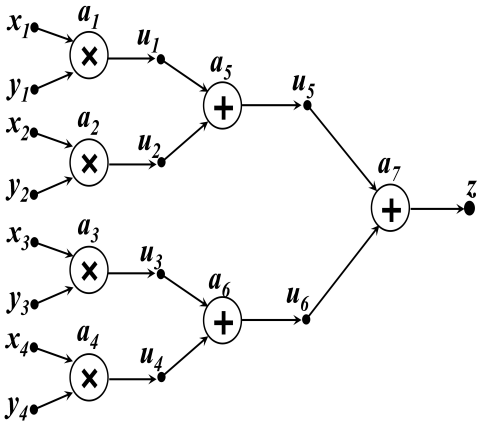


Рис.2. Информационный граф скалярного произведения векторов размером 4

Информационные графы взаимно однозначно соответствуют системе формульных выражений в явном виде, в которых объектным вершинам ставятся в соответствие уникальные имена переменных, а также типы данных, а операторным – уникальные идентификаторы операторов, а также типы операций.

На рис. 2 приведён пример информационного графа для скалярного произведения векторов $\boldsymbol{x}=(x_1, x_2, \ldots)$ и $\boldsymbol{y}=(y_1, y_2, \ldots)$: $z=\sum_i x_i * y_i$

Информационный граф рис.2 строится по исходной системе формульных выражений в явном виде:

$$(1)\quad \begin{cases} z=u_5+u_6; \\ u_5=u_1+u_2; \\ u_6=u_3+u_4; \\ u_1=x_1*y_1; \\ u_2=x_2*y_2; \\ u_3=x_3*y_3; \\ u_4=x_4*y_4; \end{cases}$$

Управляющие автоматы [12]

Порядок выполнения действий в информационном графе определяется правилом: каждая операторная вершина может выполняться, если введены или вычислены значения всех её аргументов (входных переменных), а также для вычисленных значений имеются свободные ячейки памяти. Автоматная модель управления реализует это правило.

Введём дополнительные обозначения.

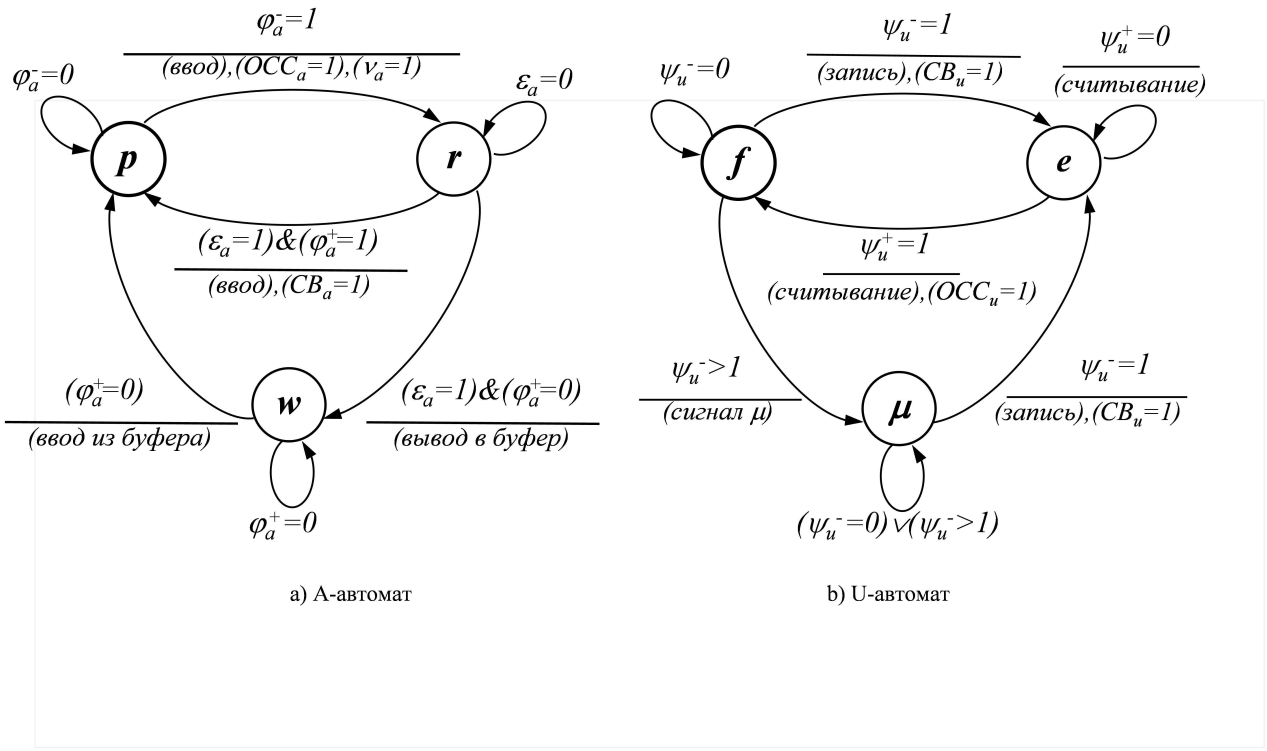
Для $\forall a \in \mathbf{A}$ ставится в соответствие два кортежа объектных вершин (переменных):

$\mathbf{U}^-(a)=\{u \in \mathbf{U}: \exists \mathbf{S}(u, a)\}$ – входной кортеж и $\mathbf{U}^+(a)=\{u \in \mathbf{U}: \exists \mathbf{S}(a, u)\}$ – выходной.

Для $\forall u \in \mathbf{U}$ ставится в соответствие два кортежа операторных вершин (операций):

$\mathbf{A}^-(u)=\{a \in \mathbf{A}: \exists \mathbf{S}(a, u)\}$ – входной кортеж и $\mathbf{A}^+(a)=\{a \in \mathbf{A}: \exists \mathbf{S}(u, a)\}$ – выходной.

Каждой операторной вершине и объектной вершине поставим в соответствие А-автомат (рис.3 a) и U-автомат (рис.3 b), соответственно.



a) A-автомат b) U-автомат

Рис. 3.

Управляющие автоматы определяют поведение операторных и объектных вершин в ходе исполнения вычислительного процесса.

Автоматы соединены между собой. Каждой дуге информационного графа соответствуют две встречно направленные связи между автоматами. Оба типа автомата в простейшем случае имеют по три состояния.

A-автоматы реализуют управление потоками данных: оператор может быть выполнен, если определены значения его входных переменных. Алфавит состояний A-автомата: $\{p,r,w\}$, где p – оператор пассивен (не все входные значения получены), r – исполняется, w – ждёт разгрузки результата.

U-автоматы реализуют принцип защиты значения объекта: новые данные не могут быть записаны в объект до момента полного использования предыдущих значений. Алфавит состояний U-автомата: $\{f,e,\mu\}$, где f – объектная вершина свободна, e – занята (хранит значение), μ – состояние неопределённости из-за конфликта по записи от двух или более операторных вершин.

Переходы автоматов из одного состояния в другое значениями индикаторных функций $\varphi^-(\,)$, $\varphi^+(\,)$, $\psi^-(\,)$, $\psi^+(\,)$. В моменты переходов автоматы вырабатывают внутренние управляющие сигналы, которые принимают значения из {0,1}. Они передаются встречно между соседними автоматами по направлению дуг информационного графа. Прямые сигналы (по направлению совпадают с ориентацией дуг) называются сигналами возбуждения (СВ). Обратные – называются сигналами обратного согласования (ОСС). Каждый автомат имеет накопитель СВ и ОСС. На них определяются индикаторные функции. Внутренние сигналы синхронизируют передачу потоков данных в информационном процессе.

Каждый A-автомат помимо внутренних сигналов вырабатывает сигнал запуска своего оператор на выполнение (ν=1) и принимает от него обратный сигнал завершения работы (ε=1), см. рис. 3а. Это внешние относительно автоматов сигналы связи с информационным графом. Они связывают информационный процесс в графе с управляющим в автоматной сети.

В ходе исполнения вычислительного процесса автоматная сеть обеспечивает автоматическое асинхронное распараллеливание в соответствии с правилом "операторная вершина готова к обработке, как только все её аргументы получают значение". Такое правило лежит в основе моделей Data Flow [11] и, как известно, обеспечивает выявление максимально возможного параллелизма вычислительных операций в информационном графе.

Важно отметить, что в пространстве сильносвязных алгоритмических взаимодействий СВ определяют прямое направление влияния в соответствии с формулой "всё влияет на всё и сразу". Сигналы ОСС определяют встречное направления в соответствии с формулой "всё зависит от всего и сразу".

Балансное уравнение вычислительного процесса

С помощью автоматной модели носителя управляющего процесса построим аналитическое выражение механизма управления потоками данных.

Множества операторных $\mathbf{A}$ и объектных $\mathbf{U}$ вершин на каждом шаге j с учётом текущего состояния соответствующих автоматов могут быть тождественно представлены в виде разбиений на непересекающиеся подмножества вершин с одинаковым состоянием:

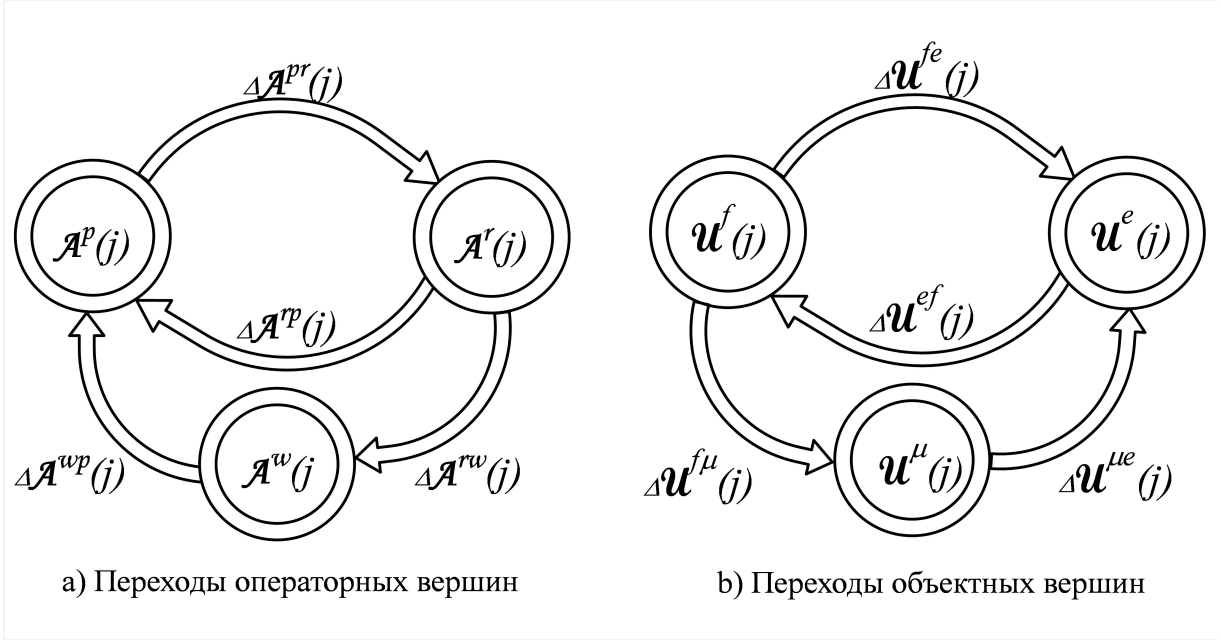
$$(2)\quad \begin{cases} \mathbf{U} \equiv \mathbf{U}^{f}(j) \cup \mathbf{U}^{e}(j) \cup \mathbf{U}^{\mu}(j), \\ \mathbf{A} \equiv \mathbf{A}^{p}(j) \cup \mathbf{A}^{r}(j) \cup \mathbf{A}^{w}(j), \\ j=0,1,2,\ldots \end{cases}$$

Отметим, что множество $\mathbf{A}^{r}(j)$ на каждом шаге j определяет множество параллельно исполняемых операторных вершин.

С учётом взаимодействия вершин информационного графа новый состав каждого из множеств (2) с одинаковыми состояниями на каждом шаге определяется следующей системой рекуррентных выражений:

$$(3)\quad \begin{cases} \mathbf{A}^{p}(j+1) = (\mathbf{A}^{p}(j) \cup \Delta\mathbf{A}^{wp}(j) \cup \Delta\mathbf{A}^{rp}(j)) \setminus \Delta\mathbf{A}^{pr}(j); \\ \mathbf{A}^{r}(j+1) = (\mathbf{A}^{r}(j) \cup \Delta\mathbf{A}^{wp}(j)) \setminus (\Delta\mathbf{A}^{rp}(j) \cup \Delta\mathbf{A}^{rw}(j)); \\ \mathbf{A}^{w}(j+1) = (\mathbf{A}^{w}(j) \cup \Delta\mathbf{A}^{rw}(j)) \setminus \Delta\mathbf{A}^{wp}(j); \\ \mathbf{U}^{f}(j+1) = (\mathbf{U}^{f}(j) \cup \Delta\mathbf{U}^{ef}(j)) \setminus (\Delta\mathbf{U}^{fe}(j) \cup \Delta\mathbf{U}^{f\mu}(j)); \\ \mathbf{U}^{e}(j+1) = (\mathbf{U}^{e}(j) \cup \Delta\mathbf{U}^{fe}(j) \cup \Delta\mathbf{U}^{\mu e}(j)) \setminus (\Delta\mathbf{U}^{ef}(j)); \\ \mathbf{U}^{\mu}(j+1) = (\mathbf{U}^{\mu}(j) \cup \Delta\mathbf{U}^{f\mu}(j)) \setminus \Delta\mathbf{U}^{\mu e}(j); \\ j=0,1,2,\ldots \end{cases}$$

Состав множеств в левой части (3) определяется как баланс переходных потоков множеств операторов и объектов правой части, которые обозначены $\Delta\mathbf{A}^{qs}$ и $\Delta\mathbf{U}^{qs}$ соответственно. Их автоматы на j такте переходят из состояния q в s, где $q,s \in \{p,r,w\}$ либо $q,s \in \{f,e,\mu\}$.



a) Переходы операторных вершин

b) Переходы объектных вершин

Рис.4.

На рис. 4 показана диаграмма балансного уравнения. Важное свойство данной модели – состояние процесса на каждом шаге полностью определяется составом множеств вершин с одинаковыми состояниями. Количество таких множеств не связано с количеством вершин в информационном графе задачи и зависит только от сложности автоматов, определяющих состояние вершин в ходе выполнения процесса. Представление параллельных и распределённых вычислительных процессов посредством балансного уравнения позволяет существенно уменьшить размерность пространства состояний процесса по сравнению с числом состояний собственно автоматной сети.

Сложность каждого типа автоматов определяется количеством его различимых состояний. В данном рассмотрении – два типа автоматов, алфавит состояний в каждом из них (рис.3) состоит из трёх символов, что даёт диаграмму балансного уравнения с 6 вершинами (по три для каждого из двух типов автоматов).

Предложенный метод очевидным образом обобщается на случаи более сложного поведения вершин графов, что обеспечивается введением автоматов с большим числом состояний.

Метод формализации процессов управления на основе балансных уравнений управления вычислительными процессами благодаря независимости своей сложности от размеров графов задач и вычислительных сред может использоваться для решения задач моделирования и управления параллельными/распределёнными процессами в больших и сверхбольших вычислительных средах.

ЛИТЕРАТУРА:

1. Затуливетер Ю.С. Проблемы глобализации парадигмы управления в математически однородном поле компьютерной информации // Проблемы управления. 2005. -Ч.I: №1, С.1-12, URL: http://zvt.hotbox.ru/Zatuliveter_comp_glob_contr_1.htm. -Ч.II: №2, С.13-23, URL: http://zvt.hotbox.ru/Zatuliveter_comp_glob_contr_2.htm.
2. Затуливетер Ю.С. Информационная природа социальных перемен. Серия "Информация и Социум", -М.: СИНТЕГ, 2001. –132с. URL: http://www.ipu.ru/sites/default/files/publications/17713/3221-17713.pdf.
3. Таненбаум Э., М. ван Стеен. Распределенные системы. Принципы и парадигмы. Издательство: СПб.: Питер, 2003. -877с.
4. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. . – 608 с.
5. Ю.С. Затуливетер, Е. А. Фищенко, С. Е. Артамонов, В. А. Козлов. Элементы стратегии опережения и архитектурные предпосылки к созданию однокристального ускорителя массовых вычислений общего назначения на базе архитектуры отечественного многопроцессорного компьютера ПС-2000 // Программные системы: теория и приложения. Электронный научный журнал Института программных систем им. А.К. Айламазяна РАН, 2014. Т. 5, № 1(19), с. 37–74. URL: http://psta.psiras.ru/read/psta2014_1_37-74.pdf.
6. Затуливетер Ю.С. Введение в проблему параметризованного синтеза программ для параллельных компьютеров / (Препринт/Институт проблем управления). -М., 1993, 88с.
7. Затуливетер Ю.С. ExaScale: на пути к единому пространству распределенных и параллельных вычислений // Научный сервис в сети Интернет: Экзафлопсное будущее: Труды Международной суперкомпьютерной конференции (20-25 сентября 2010г., г. Новороссийск). – М.: Изд-во МГУ, 2011. С.10-14. –URL: http://agora.guru.ru/abrau2011/pdf/10.pdf.
8. Затуливетер Ю.С., Фищенко Е.А. Принципы формирования универсального бесшовно программируемого и кибербезопасного алгоритмического пространства // Программные системы: теория и приложения. Электронный научный журнал Института программных систем им. А.К. Айламазяна РАН, 2014. Т. 5, № 1(19), с. 153–173. URL: http://psta.psiras.ru/read/psta2014_1_153-173.pdf.
9. Затуливетер Ю.С. На пути к глобальному программированию // Открытые системы. 2003. № 3. – С. 46-47. -URL: http://www.osp.ru/os/2003/03/182704/.
10. Затуливетер Ю.С. Компьютерный базис сетецентрического управления // Российская конференция с международным участием "Технические и программные средства в системе управления, контроля и измерения" (УКИ'10). Труды конференции. Москва, 18-20 октября 2010 г. Учреждение Российской Академии наук Институт проблем управления им. В.А. Трапезникова РАН. С.17-37. -URL: http://cmm.ipu.ru/proc/Затуливетер%20Ю.С.%20.pdf.
11. Dennis J. Data Flow Supercomputers // Computer. - Vol.13. - No.11. Nov, 1980. P.48-56.
12. Затуливетер Ю.С. Алгоритм согласованных асинхронных вычислений. В кн.: Однородные вычислительные структуры и малые ЭВМ: Тез. докл. на Всесоюзном семинаре, г. Звенигород, 1979, с.112-116 (Институт проблем управления, Москва).