АЛГОРИТМИЧЕСКИЕ СЕТИ И ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ

В. Е. Марлей

Санкт-Петербургский институт информатики и автоматизации РАН
199178, Санкт-Петербург, 14-я линия, В.О., д.39
vmarley@mail.iias.spb.su
iv@iias.spb.su

УДК 681.3.06
В. Е. Марлей. **Алгоритмические сети и параллельные вычисления** // Труды СПИИРАН, Вып. 1, т. 2. — СПб: СПИИРАН, 2002.
Аннотация. *В работе рассматривается возможность использования формализма алгоритмических сетей для автоматизации распараллеливания программ.* — Библ. 4 назв.

UDC 681.3.06
V. E. Marley. **Algorithmic nets and concurrent calculations** // SPIIRAS Proceedings, Issue 1, v. 2. — SPb: SPIIRAS, 2002.
Abstract. *The article discussed the possibility of applying the formalism of algorithmic nets for automation of creation concurrent program from usual one.* — Bibl. 4 items.

Алгоритмический подход и алгоритмическое моделирование интенсивно развиваются в Санкт-Петербургском институте информатики и автоматизации РАН с момента его возникновения. В рамках данного направления был разработан язык представления структуры алгоритмических моделей на основе “алгоритмических сетей” [1, 2]. Под алгоритмической моделью здесь понимается формализованное описание сценария предметного специалиста для моделируемого процесса, структура которого сопоставима со структурой причинно-следственных и временных зависимостей между явлениями моделируемого процесса, вместе со всей информацией необходимой для ее программной реализации. Алгоритмические сети (АС) используются для представления структуры алгоритмических моделей и являются отображением причинно-следственных и временных связей между явлениями, представленными в сценарии моделируемого процесса, в вычислительные связи между операторами. Таким образом, алгоритмические сети задают некоторую расчетную схему, по которой проводятся вычислительные эксперименты над моделями или их функциональную спецификацию.

Если в логических функциональных схемах допустить в вершинах операторы любой природы, а не только логические, и обмениваться операторы будут не только логическими переменными, а переменными любого типа, то получится конструкция, которая называется алгоритмической сетью. Наличие элементов памяти (задержка) позволяет алгоритмическим сетям описывать динамические процессы, синтаксис соответствует синтаксису логических схем (на один вход оператора нельзя подавать одновременно два разных сигнала, недопустимы контура без элементов памяти), при этом процесс считается синхронизованным и в течение одного такта любой сигнал может измениться только один раз. С другой стороны алгоритмическую сеть можно считать графическим представлением некоторой системы рекуррентных выражений, при этом типы переменных в ней могут быть произвольными, а не только числовыми, и схемой вычислений позволяющую получить значения всех переменных вычисляемых в ее

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

вершинах.

Алгоритмические сети (АС) давно и успешно используются в СПИИРАН для автоматизации моделирования и для разработки моделей в различных предметных областях (экономика, экология, энергетика, кораблестроение, химические технологии и т. п.). На основе этого формализма было разработано несколько версий системы автоматизации моделирования КОГНИТРОН. В настоящее время разработан формальный аппарат алгоритмических сетей (алгебра алгоритмических сетей) и алгоритмических моделей на их основе разработана новая версия системы КОГНИТРОН для распределенного моделирования, работающая и под Windows и под Linux.

Обычно алгоритмические сети определяются [2] как конечный ориентированный нагруженный граф, вершинам которого сопоставлены операторы, а дугам переменные, связываемые операторами. Дуга в АС означает, что операторы в связываемых ей вершинах вычислительно связаны. Вычисления на АС производятся по шагам моделирования, происходят в каждой вершине сразу же, когда для текущего шага моделирования становятся известны значения всех переменных соответствующих входным дугам вершины. Переменные соответствующие входным висячим дугам соответствуют исходным, входным переменным, все остальные переменные — расчетные (внутренние). Операция задержки в вершинах АС служат для задания исходного состояния моделируемого процесса и для описания перехода моделируемого процесса через шаг моделирования. Уточним ограничения на структуру АС и условия ее функционирования:

- не существует контура, в котором хотя бы одна вершина не содержала оператор задержки;
- переменная может быть вычислена только в одной вершине;
- в одной АС все операторы задержки определяют рекуррентные соотношения относительно одной и той же величины и с одинаковым шагом ее изменения, и срабатывают одновременно после вычисления всех остальных вершин.

Пример АС приведен на рис.1. Для АС всегда можно построить расчетную программу, относительно множества ее входных переменных. АС отражают только структуру алгоритмических моделей, то есть только части модели.

Так как АС графически представляет некоторую вычислительную схемы, то преобразования АС это будет графическое представление преобразований над математическими выражениями. Подграф АС в этом случае становится схемой вычисляющей некоторое подмножество внутренних (расчетных) переменных АС и, чтобы он сохранял это качество, необходимо чтобы вершины подграфа АС сохраняли все связанные с ними переменные. Т.е. звездный граф соответствующий вершине АС со всеми дугами соответствующими связанным с ней переменными есть неделимая единица АС, единичная АС. Таким образом, понятие подграфа для АС получается более узким, чем в теории графов. Это побудило разработать для АС свою систему операций. Состав операций определен на основании имеющегося опыта практической работы с алгоритмическими моделями. Для АС определены отношения равенства, изоморфизма, включения и изоморфного включения. Определены бинарные операции объединения, пересечения и разности, унарные операции агрегации, дезагрегации, преобразования выражения в вершине, частичного обращения и выделения подграфа. Введенные отношения и операции определяют алгебру алгоритмических сетей. Для всех введенных операций разработаны алгоритмы реализации, дающие однозначный результат за конечное время. Аналогичные алгорит-

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

мы разработаны и для распознавания всех введенных отношений.

Для описания распределенных моделей введено понятие распределенных алгоритмических сетей. Распределенная АС может быть получена объединением отдельных АС или декомпозицией исходной. Множество отдельных АС, объединение которых было бы не пусто, называется распределенной АС (РАС), а отдельные составляющие ее АС фрагментами распределенной АС (ФАС). Распределенная АС всегда рассматривается при одинаковом шаге моделирования и одном и том же общем числе шагов для всех составляющих ее фрагментов. Пример распределенной АС представленной замещающей ее единой АС приведен на рис. 2.

Все операции и отношения для РАС реализуются так, как если бы существовала обычная АС составленная из имеющихся фрагментов $AC'=AC_1\cup...\cup AC_n\neq\varnothing$. Все операции над распределенными АС определяются через операции над подобной сетью, и отличаются только алгоритмами реализации (необходимо проводить поиск не только вершины, но и фрагмента, в котором она расположена).

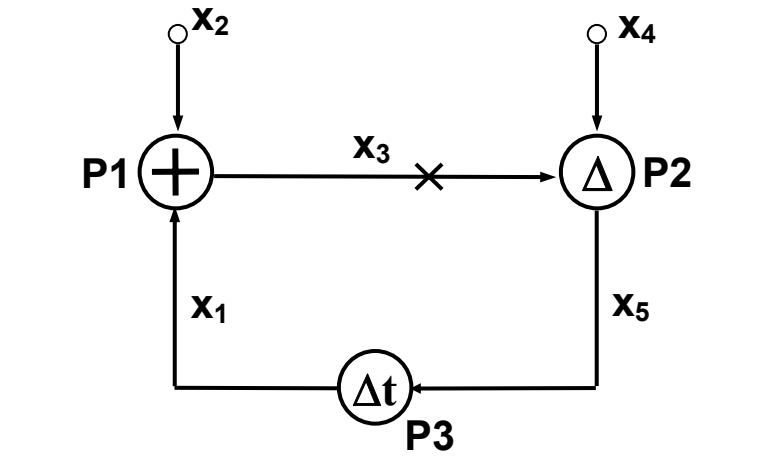
Вычисления на распределенной АС во многом аналогичны вычислениям на обычных АС, но имеют свои особенности. Поскольку исходная распределенная АС вычислима при заданном множестве входных переменных, то хоть в одном фрагменте есть хотя бы одна вершина, которая может быть вычислена только на основании этого множества, вычисленная переменная пополнит число известных. Далее возможно, что вычисление данной вершины позволит вычислить еще ряд вершин в рассматриваемом фрагменте и т.д., но это, исходя из принятых начальных условий, приведет не к полному вычислению фрагмента, а только части его сети. Выделим ту часть фрагмента АС, которая вычислялась, затем вычислимая часть выделяется как отдельный вычислимый подфрагмент. Пополненное множество известных переменных, даст вычислить хотя бы одну вершину в другом фрагменте, выделим вычислимый подфрагмент в другом фрагменте РАС. Таким образом, когда будет вычислена вся АС, ее фрагменты будут разбиты на подфрагменты. В процессе проведения вычислений по распределенной АС, после вычисления каждого такого подфрагмента происходит прерывание вычислений по фрагменту, и он ожидает прихода дополнительных исходных данных, пока их не хватит для проведения вычислений по еще одному подфрагменту и т.д. Операторы задержки срабатывают в распределенной АС также как и в обычной — все одновременно, после вычисления всех остальных вершин АС.

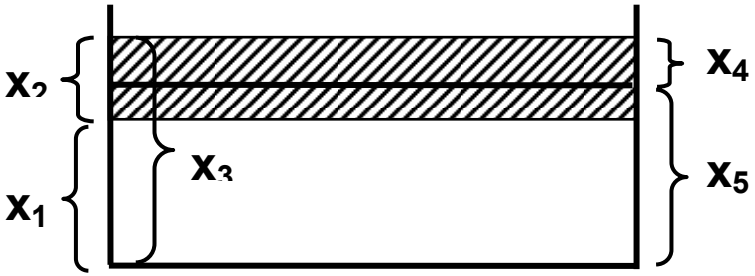
Разработанная алгебра алгоритмических сетей вполне успешно может применяться и для преобразований функциональных логических схем (исключение составит только операция частичного обращения).

Для обеспечения алгоритмической полноты алгоритмических сетей (по Дейкстре) используется возможность ссылки в вершине АС на другую алгоритмическую модель. Такой прием позволяет реализовывать ветвящиеся программы с вложенными циклами, синтезируя каждый раз только линейные программы.

Алгоритмические сети унаследовали от функциональных логических схем важное свойство — они представляют процесс вычислений в максимально распараллеленном виде, поскольку вычисления в каждой вершине производятся сразу же, как становятся известны значения входных переменных. Как условие синхронизации процесса вычислений выступает требование одновременного

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru





P_1: $x_3(t)=x_1(t)+x_2(t)$

P_2: $x_5(t)=x_3(t)-x_4(t)$

P_3: $x_1(t+1)=x_5(t)$

$x_1(t+1)=x_1(t)+x_2(t)-x_4(t)$

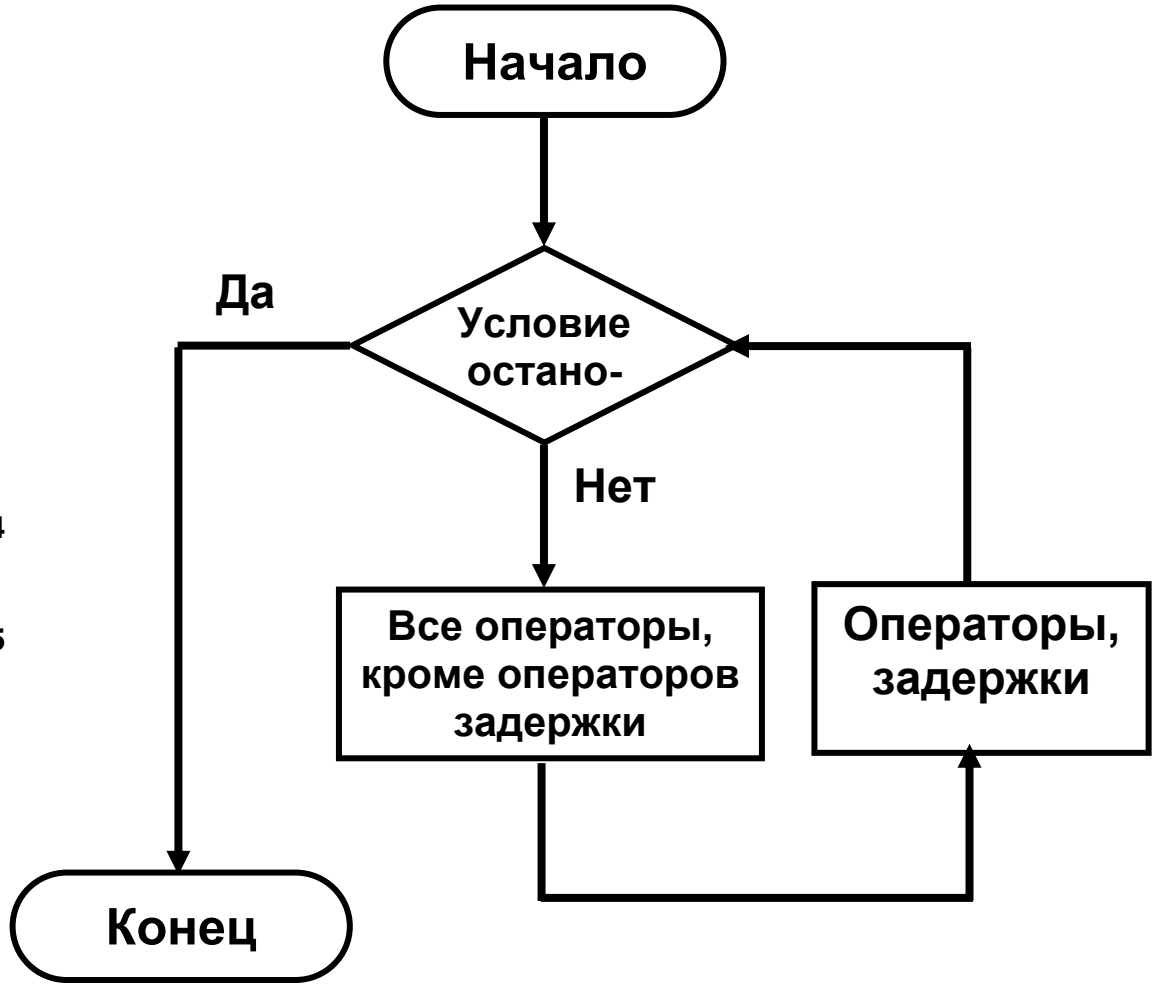
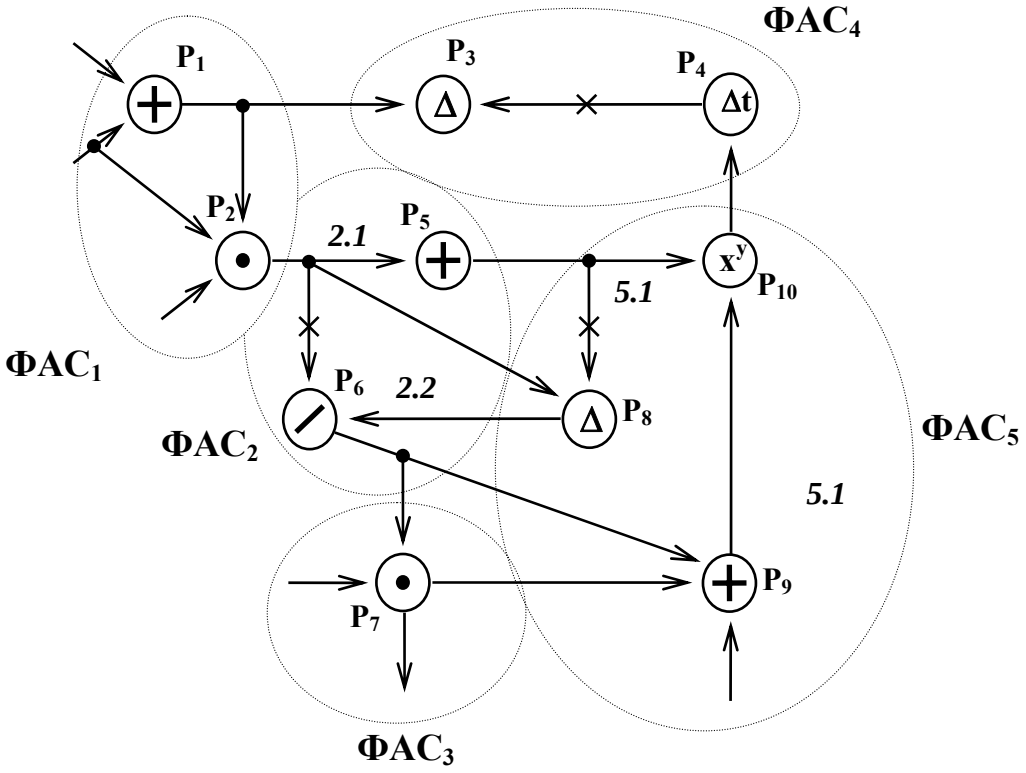


Рис. 1. Пример алгоритмической сети и реализуемого ей алгоритма

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru



План вычислений

$$P_1\ P_2\ P_3\ P_5\ P_8\ \begin{matrix} P_6 \\ P_7 \end{matrix}\ P_9\ P_{10}\ P_4$$

Рис. 2. Замещающая АС распределенной сети с выделенными фрагментами и вычислительными подфрагментами

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

срабатывания всех элементов задержки и переход к следующему такту вычислений только после окончания всех вычислений во всех вершинах сети на данном такте вычислений (аналогия с тактовым импульсом в конечных автоматах). Процедура планирования вычислений на алгоритмических сетях позволяет планировать вычисления для заданного числа используемых процессоров. Это позволяет использовать алгоритмические сети для автоматизации распараллеливания программ, что является весьма актуальным в настоящее время. В частности одним из тормозов широкого использования суперкомпьютерных кластеров является необходимость для пользователей самостоятельно распараллеливать свои программы. Алгоритмические же сети могут быть использованы для автоматизации данного процесса.

Рассмотрим алгоритм, реализуемый при расчете по алгоритмической сети. Сначала производится планирование вычислений, которое состоит из создания упорядоченной последовательности вершин, при которой каждая вершина встречается в ней только тогда, когда известны значения всех ее входных переменных. Затем, в соответствии с этим планом синтезируется последовательность математических выражений соответствующих вершинам, которая и выступает как расчетная часть программы, при этом выражения для всех вершин с операторами задержки помещаются в самом конце программы после всех выражений соответствующих вершинам с операторами других типов. Пример плана вычислений для АС приведен на рис.3. Алгоритм планирования вычислений приведен на рис.4. Полученная линейная расчетная программа добавляется к некоторой инвариантной для всех программ на основе АС части, в которую заносится условие останова расчета (например, число шагов моделирования, на которое нужно просчитать модель). Таким образом, получаемая расчетная программа представляет собой единичный цикл с линейной частью соответствующей АС как это показано на рис.1. Однако подобная программа будет, только если она составляется при условии использования только одного процессора, в силу причин изложенных в предыдущем абзаце для представления алгоритма, реализуемого АС удобнее использовать параллельные граф-схемы алгоритмов (ПГСА). В [3] показано, что АС соответствуют максимально распараллеленные ПГСА, пример ПГСА соответствующей АС на рис.3. приведен на рис.5. Из этой ПГСА видно, что в принципе мы имеем возможность в отдельные моменты вычислять операторы для двух вершин, т.е. использовать два процессора. На рис.3 приведен также план вычислений для данного случая. То есть для АС можно строить планы вычислений и расчетные программы для заданного числа процессоров и при этом всегда можно узнать какое максимальное число процессоров целесообразно использовать для расчета по данной АС.

Рассмотренные свойства АС позволяют ее эффективно использовать для автоматизации распараллеливания программ. Однако класс обычных и даже распределенных АС описывает достаточно узкий класс алгоритмов, тем более, что имеющиеся в АС логические условные операторы обеспечивают только управление данными, а не управление вычислениями. То есть алгоритмы только с одним циклом и линейной (но возможно распараллеленной) программой внутри него. Для того чтобы расширить класс представимых в АС алгоритмов, введен новый вид АС — АС со ссылками в вершинах на другие модели. При этом модели на которые ссылаются могут считаться со своим шагом и условием останова и рекурентность в них может быть по другим переменным, отличным от исходной АС (например, исходная АС считается по годам, а та на которую ссылаются считается по пройденным километрам). Из исходной АС передаются в вызываемую АС значения переменных, которые должны быть использованы при расчете в вызываемой АС, которая, после окончания расчета, передает значения своих переменных для выходных дуг вершины исходной АС со ссылкой. Условия останова вызываемой модели, имя вызываемой модели, массив данных с которым она будет считаться, также могут быть сформированы внутри вызывающей АС. Таким обра-

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

зом, появляется возможность описания алгоритмов с вложенными циклами, то есть реализация одной из конструкций Дейкстры [4]. Использование конструкции Дейкстры обусловливает, что в вызывающую АС будут передаваться только последние из вычисленных значений переменных или последнее состояние числового массива В случае не скалярных переменных.

Использование ссылок на другие модели позволяет и описать управление вычислениями, соответствующее конструкции ветвления используемого Дейкстрой. Пример такой конструкции приведен на рис. 6. В соответствии с результатом проверки условия вызывается либо одна, либо другая модель и соответственно вычисления передаваемой исходную АС переменной будут происходить разными способами. Правда описывается только случай, когда ветвление описывает разные способы вычисления одной переменной. В случае если требуется описать вычисление по разным веткам различных переменных, то рассматриваемая конструкция усложняется, так как в исходной АС потребуется ввести распознавание какой переменной приравнять полученное значение. Таким образом, АС со ссылками в вершинах на другие модели оказываются способны описывать структурные (по Дейкстре) алгоритмы, класс которых, как показано в [4] равномощен классу всех алгоритмов и можно говорить об алгоритмической полноте АС как средства представления алгоритмов.

Данный подход, конечно, имеет недостатки. Необходимо исходную программу представить в виде структурной, затем по сути дела приходится каждый вложенный цикл или ветвление выделять в отдельную подпрограмму. Но построение структурной программы из исходной и выделение в ней подпрограмм может быть полностью автоматизировано. Преимуществами является то, что в каждой из выделенных подпрограмм внутри цикла буде только линейные программы, для которых легко автоматически установить информационные связи, что позволит рассматривать их информационный граф как АС и использовать алгоритм планирования вычислений на АС для автоматического распараллеливания программ для заданного числа процессоров.

Конечно, это даст распараллеленную форму только для выделенных подпрограмм и встанет вопрос об их совместном распараллеленном выполнении. Здесь явно потребуется какая-то оптимизация процесса выполнения программы, но основа для формирования хотя бы одного допустимого варианта уже есть. Например, когда при выполнении программы встречается ссылка на подпрограмму, начинать выполнение подпрограммы дополняя, в случае если она не использует всех имеющихся процессоров, это выполнением операторов исходной, а если свободных процессоров нет останавливать выполнение исходной до окончания выполнения подпрограммы. Решение данных проблем является темой дальнейшего исследования.

Таким образом, процесс автоматизации распараллеливания программ для их использования в суперкомпьютерных вычислительных кластерах может быть представлен в виде цепочки «программа — структурная программа — алгоритмическая сеть — план вычислений для заданного числа процессоров — распараллеленная программа».

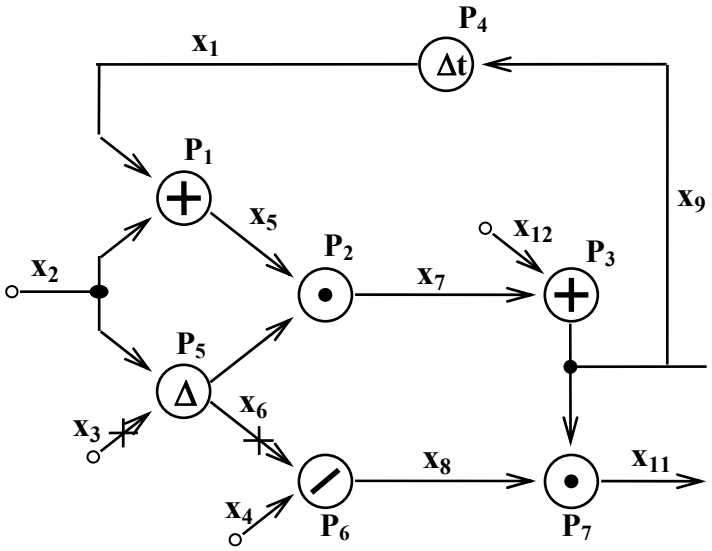
Литература

[1] *Иванищев В.В*. Алгоритмический базис для описания механизмов экономики // Алгоритмические модели и автоматизация научных исследований. М.: Наука, 1980, с.37–42.

[2] *Иванищев В.В, Марлей В.Е.* Основы теории алгоритмических сетей. СПб: СПбГТУ, 2000. — 180 с.

[3] *Марлей В.Е.* алгоритмические сети и параллельные граф-схемы алгоритмов // Проблемы информационной технологии и интегральной автоматизации производства. Л.: Наука, 1989. — с. 130–136.

[4] *Дал У., Дейкстра Э., Хоар К.* Структурное программирование. М.: Мир, 1976. — 90 с.



P1 P5 P2 P6 P3 P7 P4

P1: x5(t)=x1+x2
P5: x6(t)=x3–x2
P2: x7=x5•x6
P6: x8=x6/x4
P3: x9=x7+x12
P7: x11=x8•x7
P4: x1(t+1)=x9(t)

а)

P1 P2
P3 P7 P4
P5 P6

б)

Рис. 3. План вычислений и соответствующая расчетная программа для случая одного процессора (а), план вычислений для случая двух процессоров (б)

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

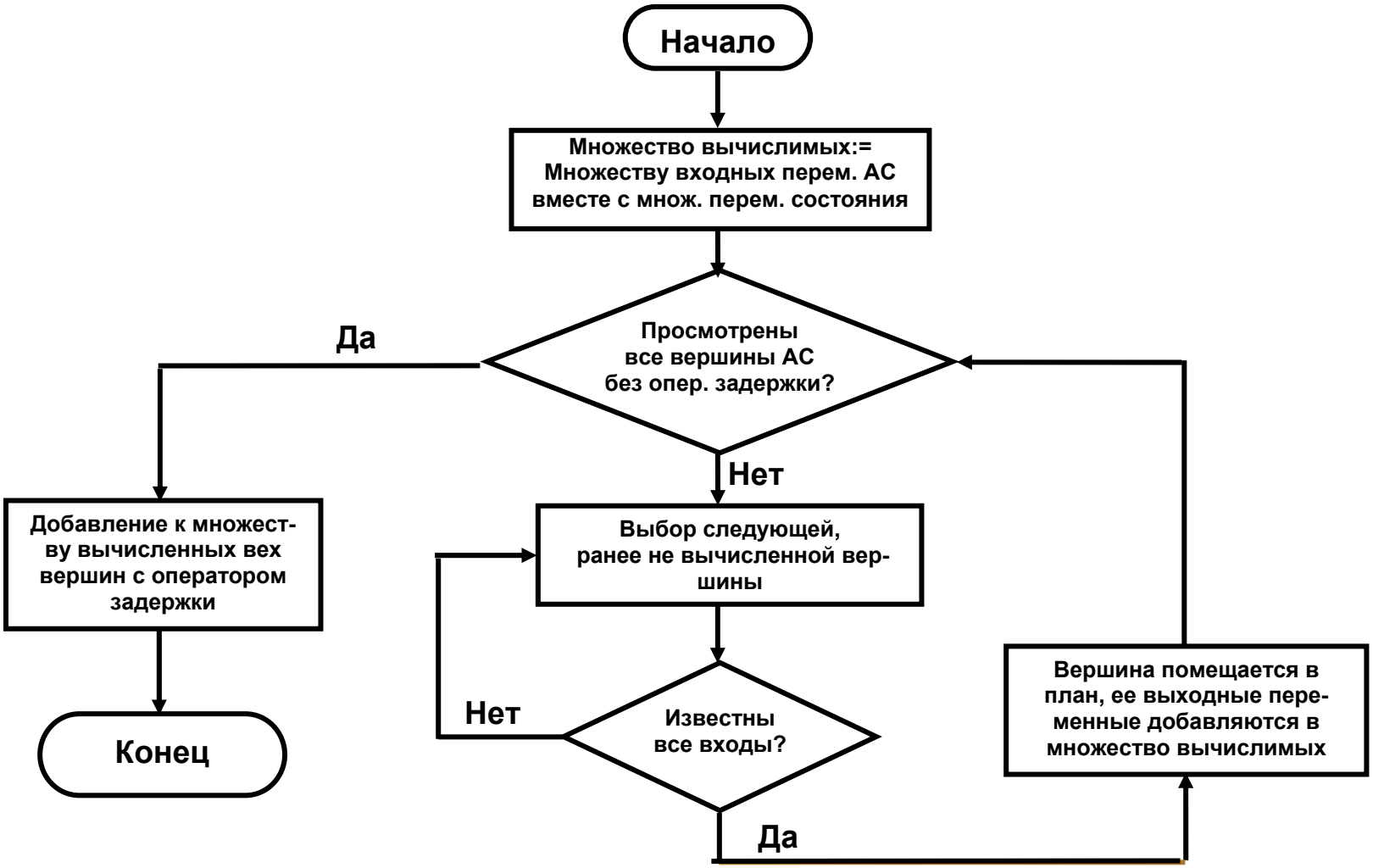


Рис. 4. Алгоритм планирования вычислений на АС

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

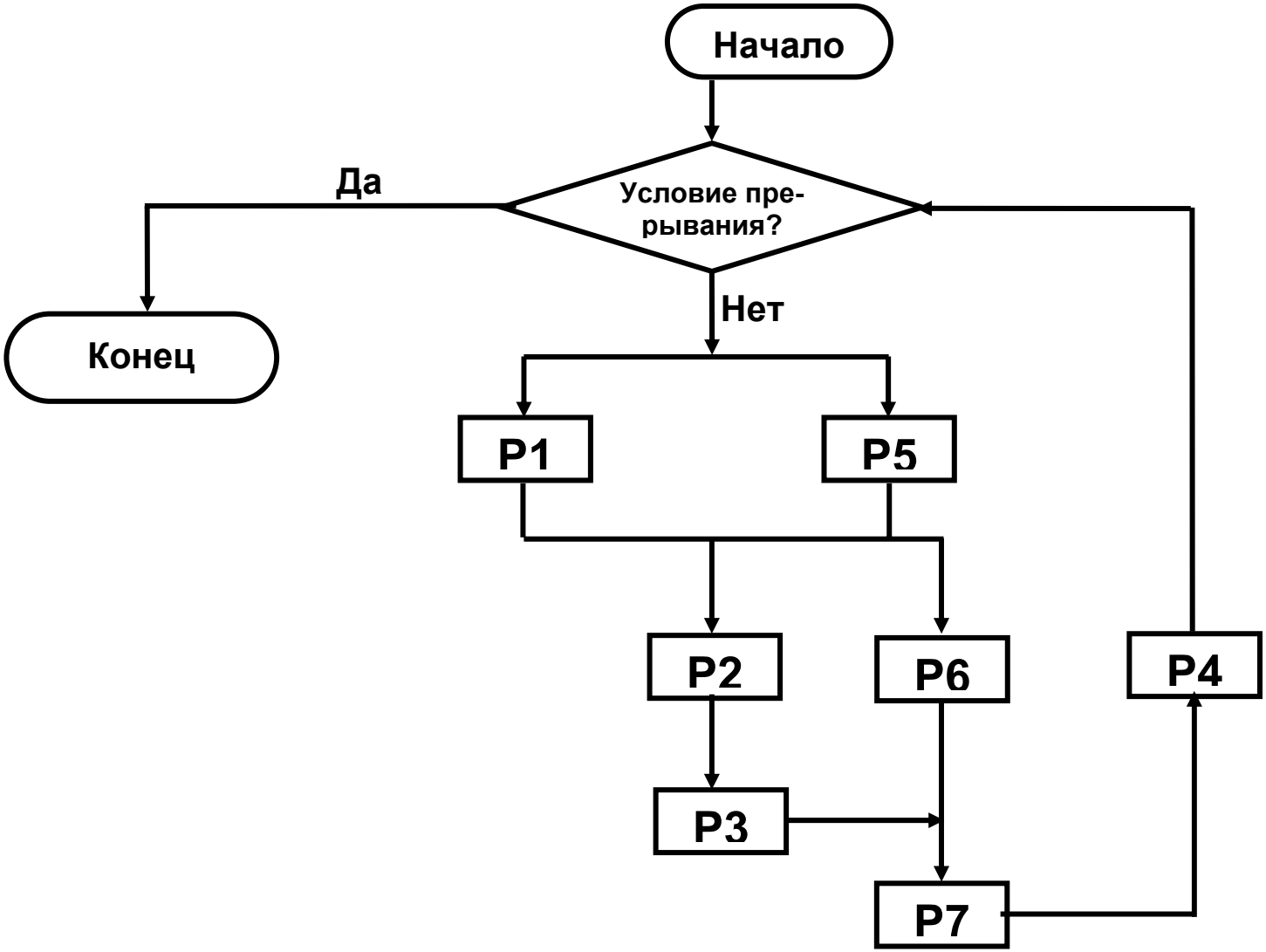


Рис. 5. Пример ПГСА соответствующей АС на рис. 2

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

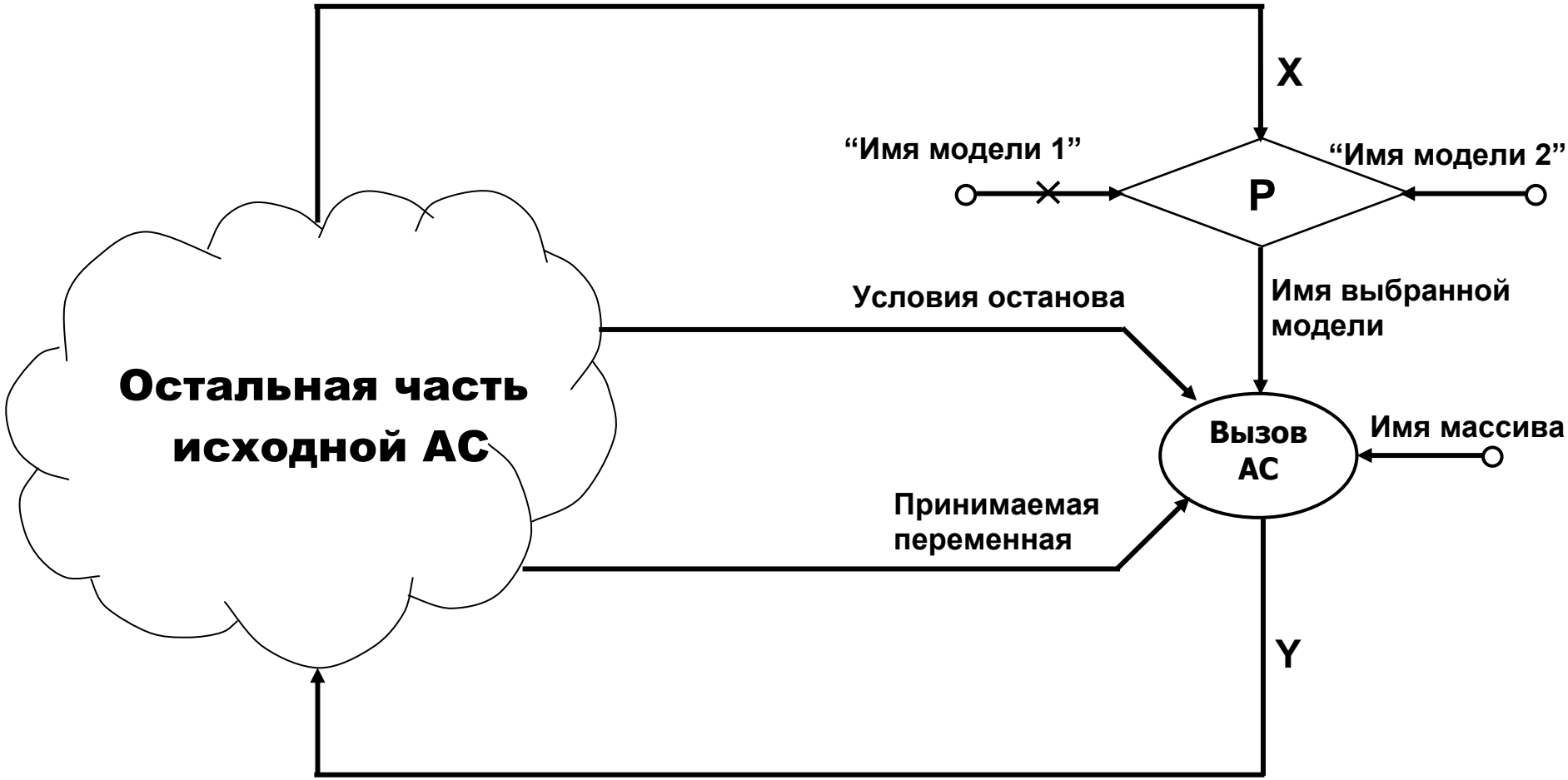


Рис. 6. Пример реализации ветвления для АС со ссылками в вершинах на другие модели

Труды СПИИРАН. 2002. Вып. 1. Т. 2. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2002. Issue 1. V. 2. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru