Академия наук СССР
ЛЕНИНГРАДСКИЙ
ИНСТИТУТ ИНФОРМАТИКИ И АВТОМАТИЗАЦИИ

Александров В. В., Сойгин А. М.

МЕТОД ПРЯМОГО КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ

Препринт № 102

Ленинград
1989

МЕТОД ПРЯМОГО КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ

Александров В.В., Сойгин А.М.

I. Введение. Проекты вычислительных систем для моделирования сложных динамических процессов

Наблюдаемые тенденции применения компьютерной техники к исследованию научных проблем в области физических явлений указывают на принципиальную возможность построения вычислительных систем, сопоставимых по сложности с моделируемыми процессами. В ряде проектов уже реализуется идея получения больших вычислительных мощностей за счёт многопроцессорности. После рассмотрения некоторых из этих проектов мы предложим метод компьютерного моделирования, реализующий идею Лапласа об адекватном описании любого физического процесса при условии полного задания начального состояния системы.

I.I. Проект FMP

Начнём с рассмотрения работы, выполнявшейся американской фирмой *Burroughs Corp.* в конце 70-х годов. Эти исследования были одной из первых попыток общего обоснования концепций организации численного моделирования сложных физических процессов, исходя из новых возможностей компьютерного рынка. Результатом стал проект FMP (*How Model Processor*), который пока остаётся лишь на бумаге [I], но история создания которого содержит много полезного.

В I975г. *NASA* обратилась к исследователям с вопросом о возможности создания системы, способной поддерживать среднюю скорость вычислений порядка I млрд. *FLOPS* (операций с плавающей точкой в сек.) и имеющей основную память не менее 40 млн. 64-битовых слов. Именно такие параметры были необходимы для перехода от обдува макетов в аэродинамических трубах (стоимость которых росла) к моделированию на компьютерах (стоимость которых снижалась). Были заключены контракты, в том числе с фирмой *Burroughs Corp.*, на проведение предварительного исследования с последующим анализом реализуемости и системным дизайном.

Концептуальная схема предложенного фирмой проекта вычислительной системы приведена на рис.I.

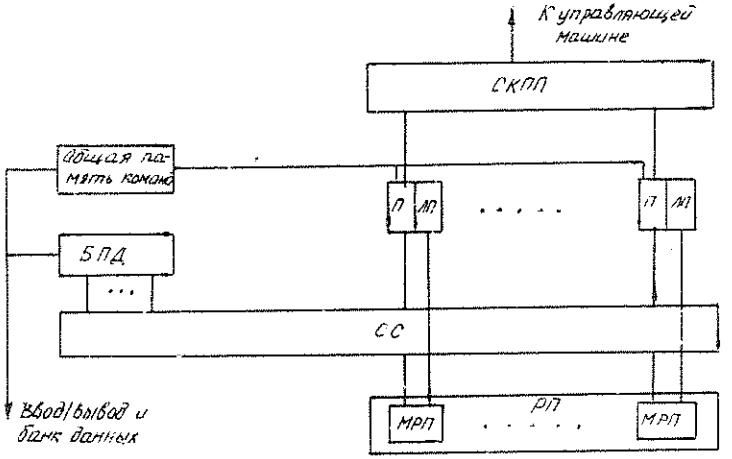

Рис. I.

Имеется много процессоров, каждый из которых может выполнять свою последовательность операций независимо от того, что делают остальные. Каждый процессор (П) имеет локальную память (ЛП) данных и команд, устройство считывания и дешифровки команд, арифметическое устройство и регистры данных. Процессоры имеют доступ к ресурсу памяти общих данных (расширенная память, РП), состоящему из модулей памяти (МРП) в количестве, равном числу процессоров. Процессоры связаны с РП через связующую сеть (СС), состоящую из двух слоёв типа омега-сети [2] (рис. 2). Входные данные и данные, предназначенные для вывода, проходят через буфер базовой памяти данных (БПД). Из БПД данные идут в большой файловый накопитель. Процессоры имеют выход на общую память команд, откуда командный код передаётся в локальную память. Они также связаны с сетью контроля и поддержки процессоров (СКПП), основная функция которой – синхронизация работы процессоров. СКПП также обеспечивает запуск и окончание работы процессоров управляющей машиной. Наличие управляющей машины позволяет также использовать её операционную систему без создания отдельной операционной системы внутри *FMP* .

Первоначально проект *FMP* предусматривал 5/2 процессоров, а в окончательном варианте их I28. Эти процессоры могут действовать параллельно по команде *DOALL* , в спецификации которой

определено множество блоков данных, над которыми выполняется описываемая после *DOALL* последовательность операций. Если число блоков данных, подлежащих преобразованию в *DOALL*, больше числа процессоров, то процессор может обращаться к РП более одного раза. Вся программа не движется дальше, пока не поступит сигнал о завершении обработки всех блоков данных. Эта барьерная синхронизация осуществляется через СКПП.

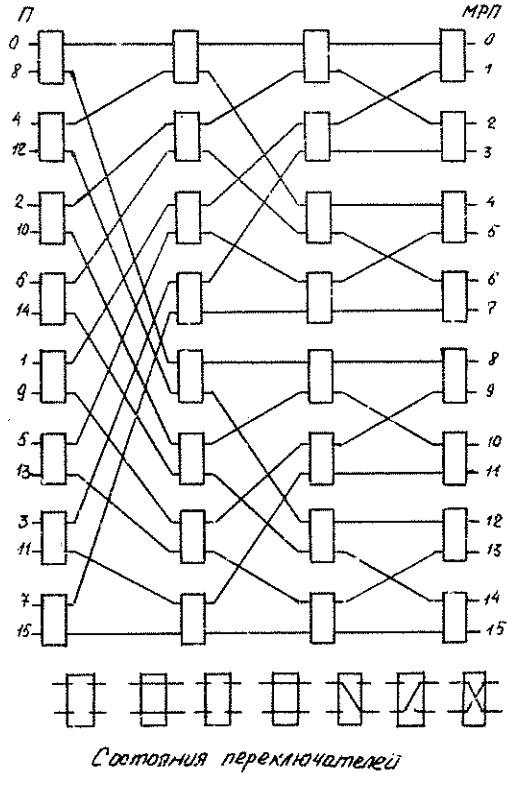

Рис. 2.

Конструкция СС обеспечивает построение пути от любого процессора к нужному МРП через самостоятельные переключения в узлах сети, исходя из заданного адреса.

Вся концепция *FMP* строилась в предположении дискретного представления области протекания физического процесса. Отдельные элементы этой области различимы и каждый идентифицируется набором параметров, определяющих его состояние. Эти параметры изменяются во времени по некоторым законам, возможно, своим для каждого элемента. Процесс сводится к изменению значений параметров каждого элемента во времени в зависимости от параметров других элементов и внешних воздействий, при этом *FMP* работает как полная *MIMD*-системa (*MIMD-Multiple Instruction Stream, Multiple Data Stream*) с общей памятью.

Параллельно с *FMP* и во взаимодействии с группой из *Burroughs Corp.* в Нью-Йоркском университете разрабатывался другой проект.

I.2. Проект *NYU Ultracomputer* [3,4]

Первые публикации по проекту *NYU Ultracomputer* относятся к 1981-82гг. [5] - [7] . Целью исследовательской группы из института им. Куранта Нью-Йоркского университета было обоснование дизайна многопроцессорной системы с общей памятью. Общая структура системы типична для *MIMD* -машины (рис. 3).

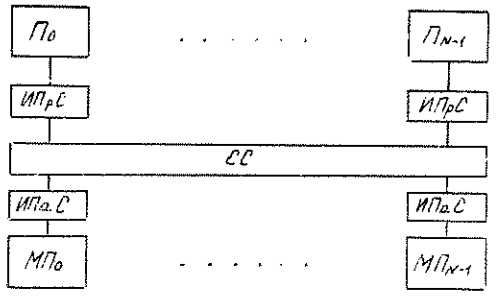

Рис. 3

Процессорные элементы $П_i$, $0 \le i \le N-1$ имеют доступ к N модулям общей памяти $МП_i$ через связующую сеть. Модули памяти и процессоры связаны с сетью через интерфейсы "процессор-сеть" и "па-

мять-сеть" (ИПаС). Связующая сеть опять имеет структуру омега-сети [2] (рис. 4). В отличие от проекта *FMP*, который с самого

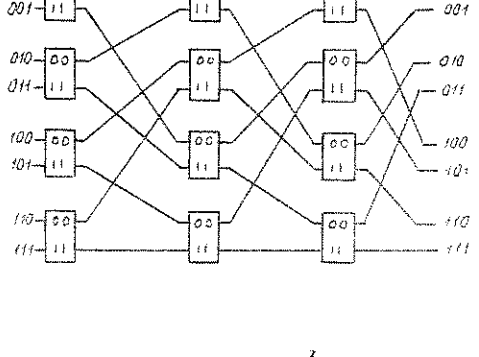

Рис. 4. $N=2^3$.

начала был ориентирован на дискретное представление процессов в сплошных средах, и это отразилось на аппаратурной реализации, дизайн *NYU Ultracomputer* более схематичен, поскольку эта система по замыслу должна обладать свойствами универсального компьютера и поэтому гораздо большее число функций возлагается на операционную систему. В проекте *FMP* использовалась только операционная система управляющей машины, дополненая встроенной в *FMP* командой *DOALL*. В *NYU Ultracomputer* операционная система пишется специально и входит составной частью в проектируемую машину.

I.3. Проект *RP3* фирмы *IBM* [8]

Проект *RP3* (*Research Parallel Processor Project*) начал разрабатываться Исследовательским отделом *IBM* в тесном сотрудничестве с группой *NYU Ultracomputer*. *RP3* – это *MIMD*-система с гибкой организацией, совмещающей как идею общей памяти, так и принцип обмена сообщениями при наличии только

локальной памяти процессоров. Дизайн RP3 включает 5/2 серийных процессоров. В полной конфигурации RP3 обеспечивает до 1.3 GIPS (млрд. команд в сек.) и 800 млн. FLOPS при 1 – 2G байт основной памяти. Скорость ввода/вывода 192 Мбайт/сек., скорость обмена между процессорами 13 байт/сек. Технология выполнения смешанная. Процессоры и основная память выполнены на полевых транзисторах с воздушным охлаждением, а переключатели связывающей сети – на биполярных транзисторах с водяным охлаждением. Конструктивно RP3 состоит из восьми Н-образных модулей (рис. 5), каждый из которых содержит 64 процессорных элемента (ПЭ) с памятью, устройством ввода/вывода и коммуникациями (рис. 6). ПЭ состоит из 32-разрядного серийного микропроцессора, 2М–4М байтовой памяти, 32К байтового кэша, устройства для операций с плавающей точкой. ПЭ имеет устройство отображения в память, динамически распределяющее память между общей и локальной.

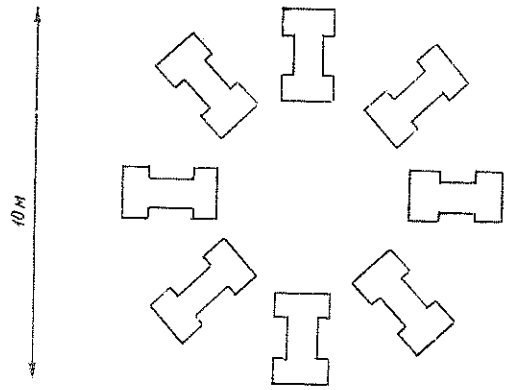

Рис. 5.

Прямой связи между ПЭ нет, только через переключатели связующей сети.

Программное обеспечение RP3 базируется на модифицированной операционной системе UNIX, выполненной в рамках проекта NYU Ultracomputer. Для RP3 подходят языки програм-

мирования СИ, ФОРТРАН, в дальнейшем ПАСКАЛЬ. Эти языки расширены объявлением данных общими или распределёнными. В результате *RP3* оказывается удобным устройством для исследования качеств многопроцессорных систем.

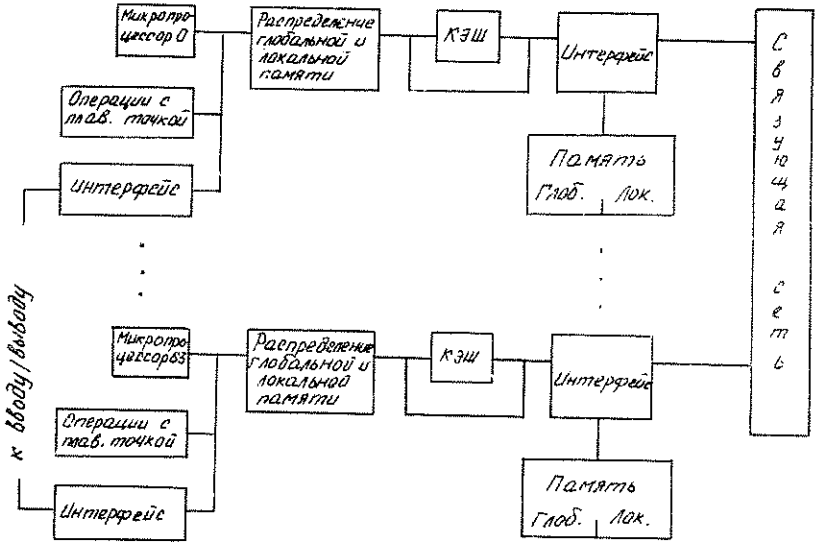

Рис. 6.

I.4. Система Cm^* университета Карнеги-Мелон [9]

Многопроцессорная система Cm^* отделения информатики университета Карнеги-Мелон, Питтсбург, Пенсильвания – это экспериментальная работающая система для научных исследований. Компонентами системы являются пары процессор-память, соединённые широкополосным каналом связи. Комбинация из процессора, локального коммутатора и памяти образует вычислительный модуль (ВМ) (рис. 7). Совокупность ВМ объединяется в кластер, а несколько кластеров образуют полную конфигурацию системы (рис. 8).

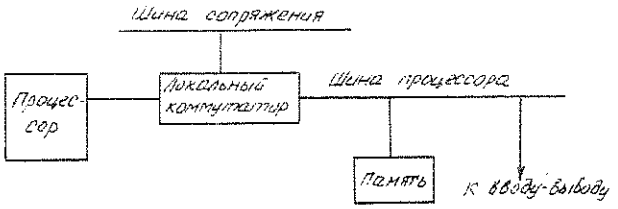
Шина сопряжения
Процес-сор
Локальный коммутатор
Шина процессора
Память
К вводу-выводу

Рис. 7.

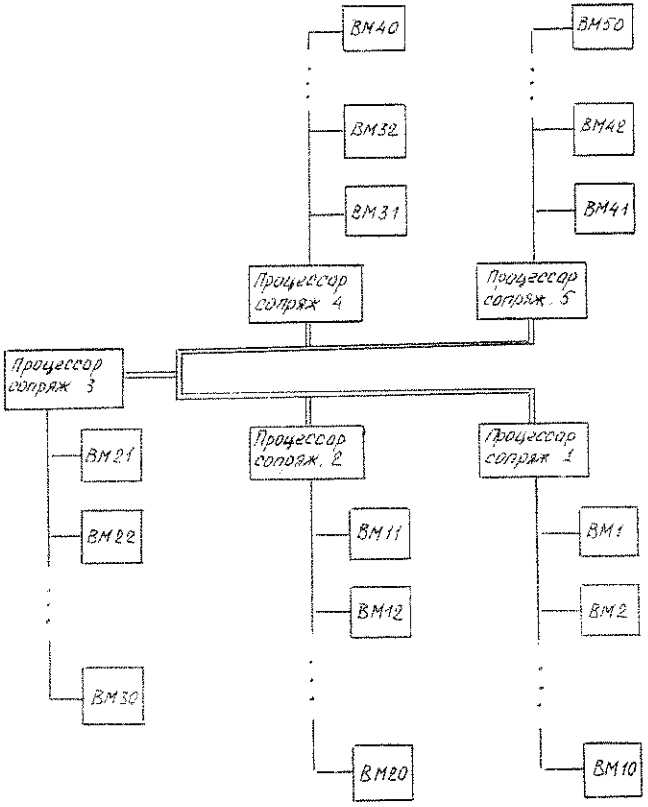
ВМ40
ВМ32
ВМ31
ВМ50
ВМ42
ВМ41
Процессор сопряж 4
Процессор сопряж. 5
Процессор сопряж 3
Процессор сопряж. 2
Процессор сопряж 1
ВМ21
ВМ22
ВМ30
ВМ11
ВМ12
ВМ20
ВМ1
ВМ2
ВМ10

Рис. 8

Каждое слово памяти имеет адрес, определяемый номером кластера, номером ВМ внутри кластера и адресом в локальной памяти ВМ. В зависимости от адреса обращения к памяти локальный коммутатор, а затем, при необходимости, процессор сопряжения организуют запрос внутри локальной памяти ВМ или внутри кластера или между кластерами. Такая организация памяти даёт преимущества локальным обращениям и уменьшает стоимость системы коммуникаций. Кроме того, нет необходимости фиксировать число кластеров и ВМ в кластере.

Система Cm^* является средством для изучения параллельных алгоритмов, и выполняемые на ней эксперименты могут указать пути эффективного использования систем из сотен процессоров, таких как система моделирования аэродинамических процессов, предложенная фирмой *Burroughs Corp.* .

I.5. Многопроцессорная система института прикладной физики Мюнстерского университета (ФРГ) [I0]

Данная система рассчитана на численное решение вполне определённого класса задач, а именно, на выполнение итерационных процессов вида:

$$u^{k+1} := N(u^k, u^{k-1}, \ldots, u^{k-m+1}),$$

где каждое очередное значение u^i представляет собой вектор большой размерности $n = 10^3 - 10^8$. Преобразование N, вообще говоря, нелинейное. Основная идея состоит в возможности одновременного вычисления n компонент вектора очередной итерации:

$$u^k, \ldots \longrightarrow N_1(u^k, \ldots) = u_1^{k+1}$$

$$\cdots\cdots\cdots\cdots\cdots\cdots$$

$$u^k, \ldots \longrightarrow N_n(u^k, \ldots) = u_n^{k+1}$$

Эта фаза независимых (автономных) вычислений, выполняемых отдельными автономными процессорами (АП). Несколько АП (сейчас в действующей системе из десять), имеющих собственную память, связаны с общей памятью. Каждый АП вычисляет в данный промежуток времени одну из компонент u_i^{k+1}. После вычисления всех компонент наступает фаза коммуникации, когда в общей памяти собирается весь вектор u^{k+1}. Структура системы изображена на рис. 9.

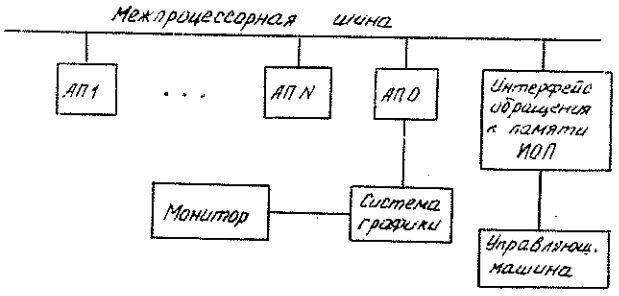

Рис. 9.

Интерфейс обращения к общей памяти (ИОП) снабжён контроллером, обеспечивающим распределение данных между АП. Управляющая машина служит для запуска системы и её контроля со стороны пользователя. Она выполнена на микропроцессоре Z80 с операционной системой CP/M, дополненной языком JCL (job control language), который нужен для управления одновременной работой АП. Каждый АП, как и в других рассмотренных системах, состоит из микропроцессора (тоже Z80), устройства арифметики с плавающей точкой и памяти. ИОП содержит общую память, локальную память для работы программы управления и микропроцессор Z80. ИОП задает АП через межпроцессорную шину адреса обращения к данным. Такой принцип согласования работы всех АП одним контролирующим устройством первоначально предусматривался и в проекте FMP, но затем функции формирования адресов обращения были переданы самим процессорам. Это было связано с трудностями настройки контролирующего устройства на каждый новый тип задач. В данном же случае, когда система предназначена для определённого типа многопроцессорных вычислений, такой подход оправдан и не требует сложной сети коммуникаций. Кроме прочего, ИОП осуществляет барьерную синхронизацию, определяя готовность всех АП к переходу от одной фазы к другой.

Результаты вычислений выводятся на графику, для чего служит отдельный процессор АП0.

Стоимость всей аппаратуры системы составляет 20 тыс. марок, а по производительности она сопоставима с компьютером Perkin - El -

I.6. Работа по созданию системы SUPRENUM (ФРГ) [II]

Данная работа интересна не в последнюю очередь потому, что позволяет увидеть, какие силы направляются в развитых странах на создание микропроцессорных систем для научных исследований.

Исходя из того, что естественные и технические науки нуждаются в сверхсложных вычислениях и основным направлением здесь является переход к многопроцессорным системам для параллельных алгоритмов, ряд научных и промышленных организаций ФРГ объединили свои усилия для работы над проектом SUPRENUM. В проекте принимают участие:

по прикладному математическому обеспечению
- Германский аэрокосмический исследовательский институт;
- фирма "Дорнье";
- центр математических исследований и обработки информации (ЦМИ) в Сант-Августине;
- ядерный исследовательский институт в Юлихе;
- ядерный исследовательский институт в Карлсруэ;
- Дюссельдорфский университет;

по языку программирования
- ЦМИ;
- Дармштадтский технический университет;
- Боннский университет;

по аппаратурной части и операционной системе
- ЦМИ;
- фирма "Крупп Атлас Электроник";
- фирма "Штольман";
- Брауншвейгский технический университет;
- Нюрнбергский университет.

В течение 1984-85г.г. разрабатывались общие идеи, концепции, подбирались партнёры. С точки зрения целей речь шла об увеличении скорости решения задач механики жидкости, метеорологии, физики плазмы, ядерной физики, геологии и т.д. на несколько порядков. Все эти задачи близки в том смысле, что исследуемые процессы определяются локальными связями, т.е. могут описываться уравнениями в частных производных. Планируемая система ориентирована на чис-

ленные методы, возникающие при дискретизации таких уравнений. Была выбрана концепция *MIMD* -системы с локальной передачей сообщений. Процессоры будут объединяться в кластеры по 16 штук, а из кластеров будет формироваться двумерный массив. Предполагается к концу 1988 г. запустить модельную систему из 16 кластеров. Операционная система будет основана на системе *UNIX*, а языком программирования является многопроцессорное расширение языка *Modula* 2.

Все расмотренные многопроцессорные системы разрабатываются для решения близких по своему происхождению задач: это моделирование процессов в реальных физических средах. Отсюда и общность основных чёрт архитектуры: все они являются *MIMD* -системами с общей памятью или с обменом сообщениями.

Далее будет изложен один общий подход к моделированию физических процессов - метод прямого компьютерного моделирования. Это численный метод, но он существенно отличается по своему смыслу от численного решения уравнений. Метод основывается на элементарных преобразованиях и обменах малыми блоками данных, которые представляют малые элементы реальной физической среды. В этом смысле метод является частным случаем синтезирования динамического дискретного пространства [12]. Как будет видно из дальнейшего изложения, основные проблемы реализации метода - это проблемы такой организации систем данных, когда их структура соответствует моделируемому явлению. Поэтому можно использовать предложенные в [12] пути разрешения противоречий между логическими связями в системе данных и их обработкой с одной стороны и фактическим размещением данных на физических носителях с другой.

2. Метод прямого компьютерного моделирования физических процессов

Описание физических процессов через математические уравнения применяется не одну сотню лет. Смысл этого описания в том, что физическому процессу сопоставляется некоторая функция, для которой известны соотношения между её значениями на каких-то подмножествах. Из этих соотношений выводятся уравнения, которые рассматри-

ваются как модель физического явления. Отыскание решений этих уравнений и их анализ составляет основную задачу теории. Переход от функциональных соотношений к уравнениям, как правило, возможен только при упрощающих предположениях. При этом могут быть утрачены важные подробности представления физического процесса.

Если теория даёт вычисляемые решения уравнений, то проблема исчерпана. Вопрос только в хорошей обусловленности формул [13]. Если же получить окончательный ответ из теории нельзя, то необходима численная модель.

Численная модель заимствует у теории машинное представление законов, описывающих физическую ситуацию. Поэтому, если теория даёт только уравнение, то его численное решение будет "моделированием моделирования".

Сложность уравнения, представляющего процесс, ещё не означает, что процесс не может быть смоделирован при помощи простой вычислимой модели [13]. Как убедиться в том, что реальная задача может быть смоделирована простым способом? Надо построить простую модель.

Предлагаемый авторами подход состоит в отказе от моделирования, как численного решения уравнений, поскольку они уже получены при определённых упрощениях. Кроме того, в излагаемом методе будет получаться очень простая численная модель.

Областью определения функции, представляющей процесс, является моделируемая физическая среда. Надо обратиться к этапу разбиения этой среды на малые элементы и записать функциональные соотношения между параметрами состояния этих элементов.

Метод прямого компьютерного моделирования (ПКМ) состоит в воспроизведении последовательных изменений параметров состояния малых элементов физической среды согласно функциональным соотношениям, записанным как операторы присваивания (рис. 10).

На каждом шаге ПКМ происходит параллельное изменение значений параметров элементов разбиения, поэтому переприсваивание этих значений может быть выполнено в принципе одновременно на множестве процессоров. В каждый момент этой фазы один процессор обслуживает один элемент разбиения среды. Изменение значений параметров состояния вызывает изменение параметров преобразования состояний на следующем шаге, поэтому до его начала требуется обмен информацией между элементами о новых значениях параметров.

Пусть исследуемая физическая среда разбита на N малых

элементов и состояние любого n-го элемента, $1 \le n \le N$ определяется конечным набором параметров $\{PRST_i(n,\tau)\}$, $1 \le i \le S$, тогда динамика среды в методе ПКМ – это изменение вместе с τ значений параметров. Предположим, что имеется многопроцессорная система с общей памятью, у которой число процессоров P, вообще говоря, не связано с числом N элементов разбиения физической расчётной среды (не исключено что P = I). По числу элементов разбиения в общей памяти формируется список, который будет содержать значения параметров состояний всех элементов расчётной среды. Каждый элемент списка содержит все определяющие параметры соответствующего элемента разбиения. С началом очередного цикла все процессоры начинают действовать самостоятельно, считывая значения $PRST$ из элементов списка состояния, ещё не обслуженных в этом цикле. Получив все нужные значения параметров, процессор начинает переприсваивания их значений по алгоритму:

$$PRST_i(n,\tau+\Delta\tau) := F_n(\{PRST_{\{i_F\}}(\{n\},\tau)\}, f(n,\tau)), \quad 1 \le i \le S, \quad 1 \le n \le N, \qquad (EPS)$$

где $\{i_F\}$ – подмножество параметров, определяющих изменение i-го параметра в зависимости от конкретного физического смысла закона преобразования; $\{n\}$ – множество элементов разбиения, влияющих на изменение состояния n-го элемента; $f(n,\tau)$ -заданные внешние воздействия. Множество $\{n\}$ определяет совокупность адресов обращения процессора к общей памяти. Очередной цикл присваивания значений параметров состояния не заканчивается, пока процессорами не будут обслужены все элементы списка параметров состояния.

Метод ПКМ даёт возможность так организовать программную систему, чтобы иметь пути варьирования алгоритмов присваивания (EPS). Различные алгоритмы будут приводить в определённых случаях к тем же результатам, что и традиционные уравнения. Метод ПКМ является численным методом моделирования, в котором воспроизводится непосредственно течение процесса, а не ищется приближённо решение уравнения, сопоставляемого процессу. Метод универсален в том смысле, что может применяться в любой физической ситуации, когда малые элементы расчётной среды можно идентифицировать значениями параметров состояния, а изменения этих параметров представить в

терминах функционально задаваемого алгоритма. Метод не нуждается в тех ограничивающих предположениях о ходе процесса, которые обычно необходимы при выводе традиционных математических уравнений, сопоставляемых физическому процессу, поэтому, в частности, нет никаких различий в моделировании линейных и нелинейных процессов, и вообще это разграничение теряет смысл.

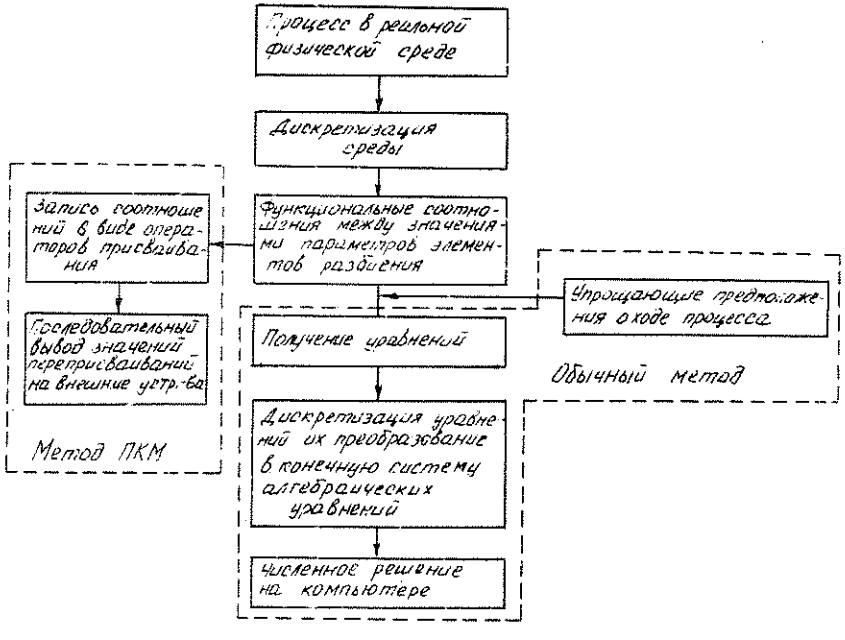

Рис. 10.

3. Пример применения метода ПКМ: колебания струны

Начальной точкой применения метода ПКМ является разбиение моделируемой физической среды (расчётной области) на малые элементы и запись конечных функциональных соотношений между значениями параметров элементов разбиения. В предлагаемом примере струна разбивается на малые участки и в качестве представляющих элементов

берутся точечные массы, движущиеся по вертикалям, расположенным на равном расстоянии друг от друга.

Параметрами фиксированного состояния произвольного элемента с номером n будут смещение $AMPL(n, t)$ относительно равновесной прямой и скорость $VELO(n, t)$ вдоль вертикали, где в качестве параметра, упорядочивающего последовательность состояний, взято время t, хотя фактически роль такого параметра выполняет счётчик циклов алгоритма (EPS). Значения $AMPL$ и $VELO$ в исходном состоянии должны быть заданы. За один цикл пересчёта оператор присваивания нового значения параметру $AMPL$ будет:

$$AMPL(n,t+\Delta t):=AMPL(n,t)+VELO(n,t)\Delta t \qquad (1)$$

для всех n, кроме может быть конечных точек струны, о чём будет сказано ниже. Здесь Δt – коэффициент пропорциональности, связывающий число циклов алгоритма ( ) с протекающим лабораторным временем.

Пересчёт параметра $VELO$ следует из второго закона Ньютона. В предельном случае, когда между элементами отсутствуют взаимные связи и они движутся только под действием внешней силы, оператор присваивания будет:

$$VELO(n,t+\Delta t):=VELO(n,t)+(1/(d\cdot\Delta x))f(n,t)\Delta t, \qquad (2)$$

где d – линейная плотность, Δx – расстояние между вертикалями, по которым движутся точечные массы. В более реальной ситуации когда струна упругая и, кроме внешней силы, действует сила натяжения, вид оператора присваивания будет другим:

$$VELO(n,t+\Delta t):=VELO(n,t)+(K/(d\cdot\Delta x))(AMPL(n+1,t)-$$
$$-2\cdot AMPL(n,t)+AMPL(n-1,t)+(1/K)f(n,t))\Delta t, \qquad (3)$$

где K – коэффициент упругости. Если струна колеблется в вязкой среде, то:

$$VELO(n,t+\Delta t):=VELO(n,t)+(1/(d\cdot\Delta x))(K(AMPL(n+1,t)-$$
$$-2\,AMPL(n,t)+AMPL(n-1,t)+f(n,t)-m|VELO(n,t)|)\Delta t, \qquad (4)$$

где m – коэффициент вязкости.

Сравнивая записанные алгоритмы с общим алгоритмом метода ПКМ, мы видим, что здесь множество $PRST\{i_p\}$ – это оба определяющих параметра состояния: $PRST_1 = AMPL$ и $PRST_2 = VELO$, а множество элементов $\{n\}$, влияющих на алгоритм пересчёта состояния n-го элемента, состоит, кроме самого n-го элемента, из двух соседних. Алгоритмы пересчёта могут выполняться одновременно для всех элементов.

Алгоритмы преобразования параметров крайних точек струны будут другими и зависят от заданных граничных условий. Если элементы, соответствующие крайним точкам, имеют номера 0 и N-I, то при закреплённых концах:

$$AMPL(0,t) = AMPL(N-1,t) = 0;$$
$$VELO(0,t) = VELO(N-1,t) = 0. \qquad (5)$$

Если концы движутся вдоль своих вертикалей по заданным законам $LE(t)$ и $RE(t)$ изменения смещения, то:

$$AMPL(0,t) := LE(t);$$
$$AMPL(N-1,t) := RE(t) \qquad (6)$$

В случае незакреплённых концов и при наличии только сил натяжения:

$$VELO(0,t+\Delta t) := VELO(0,t) + \frac{2K}{d\cdot\Delta x}[AMPL(1,t) - AMPL(0,t)]\Delta t;$$
$$VELO(N-1,t+\Delta t) := VELO(N-1,t) - \frac{2K}{d\cdot\Delta x}[AMPL(N-1,t) - AMPL(N-2,t)]\Delta t. \qquad (7)$$

Наличие внешних сил и сил вязкости учитывается их добавлением к выражению в квадратных скобках.

В изложенном примере не возникло ни одно из ограничений, необходимых для перехода к известному дифференциальному уравнению колебаний струны. Можно записать и другие алгоритмы преобразований, в зависимости от представления элементов разбиения.

Чтобы сравнить метод ПКМ с обычными методами описания и оправдать утверждение из п.3 о возможности значительных упрощений, приведём схему решения одномерного волнового уравнения в случае закреплённых концов струны и отсутствия вязких сил.

Переход от функционального соотношения между параметрами состояния представляющих элементов, которое мы записали в виде операторов переприсваивания, к уравнению в частных производных возможен только в предположении о"малости"колебаний:

$$\Delta x \approx \sqrt{(\Delta x)^2 + (AMPL(n+1,t) - AMPL(n,t))^2} \qquad (8)$$

для всех n и t. В противном случае возникнет нелинейное уравнение, аналитическое решение которого вообще невозможно. Тем самым мы выбираем самый лёгкий вариант обычного численного решения.

Пусть начальная форма струны с закреплёнными концами является графиком функции $u(x)$, $0 \le x \le L$, начальная скорость – $v(x)$, $0 \le x \le L$, а внешняя действующая сила будет $F(x,t)$, $0 \le x \le L$, $t \ge 0$. Тогда форма струны определяется функцией:

$$u(x,t) = \sum_{n=1}^{\infty} b_n(t) \sin \frac{n\pi x}{L} + \frac{1}{2}\left[u(x-at) + u(x+at) + \frac{1}{a}\int_{x-at}^{x+at} v(s)\,ds\right] \qquad (9)$$

Здесь:

$$b_n(t) = \frac{L}{n\pi a}\int_0^t B_n(\tau) \sin \frac{n\pi a}{L}(t-\tau)\,d\tau \qquad (10)$$

$$B_n(\tau) = \frac{2}{L}\int_0^L F(\xi,\tau) \sin \frac{n\pi \xi}{L}\,d\xi, \qquad (11)$$

$$a = \sqrt{\frac{T}{d}},$$

где T – постоянная сила натяжения вдоль струны, что опять является необходимым предположением для вывода уравнения колебаний, но совершенно не нужно в методе ПКМ, тем более, что это явное огрубление, исключающее неоднородность линейных деформаций вдоль струны.

Для использования формулы (9) необходимо выполнить нечётное периодическое продолжение функций $u(x)$ и $v(x)$, первоначально заданных при $0 \le x \le L$, на всю ось с периодом $2L$. Вопрос выбора числа членов конечной суммы ряда $\sum_{n=1}^{\infty} b_n \sin \frac{n\pi x}{L}$ также тре-

бует отдельного рассмотрения. После вычисления интегралов (I0) и (II) тогда остаётся вычисление значений искомого решения $u(x,t)$ во всех нужных точках, что эквивалентно выполнению операторов присваивания, составляющих весь алгоритм ПКМ. Сравнительная схема метода ПКМ и численного решения уравнения колебаний струны приведена на рис. II.

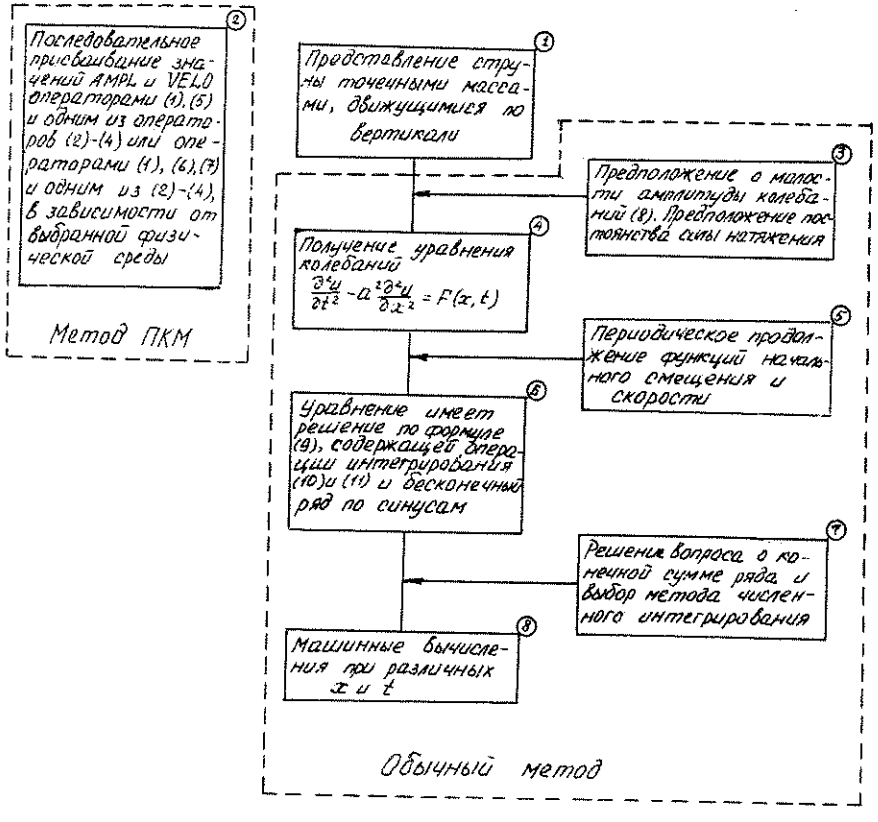

Рис. II. Блоки ② и ⑧ сравнимы по числу выполняемых операций присваивания.

4. Возможность отображения метода ПКМ на многопроцессорные системы

Проанализируем кратко совместимость метода ПКМ с рассмотренными в п.I проектами многопроцессорных систем.

Концепция, положенная в основу проекта *FMP*, близка к общей идее метода ПКМ, но в аппаратурной реализации *FMP* есть один существенный момент, затрудняющий выполнение алгоритма ПКМ. В нашем методе текущее состояние моделируемой среды представляется списком состояний элементов, число которых никак не связано с числом процессоров вычислительной системы. Этот список состояний, применительно к *FMP*, может находиться только во внешнем файловом накопителе. То, что в *FMP* модули расширенной памяти доступны любому из процессоров, для ПКМ не играет никакой роли. Нужен доступ процессоров к элементам списка состояния. Сказанное в равной степени относится к проектам *NYU Ultracomputer* и *RP3*, т.к. в них под общей памятью также понимается набор модулей памяти в том же количестве, что и число процессоров. Это равенство принципиально для функционирования самопереключающейся связующей омега-сети. Проблема структуры общей памяти и системы коммуникаций между процессорами и списком состояний в методе ПКМ не может быть решена только за счёт аппаратурной реализации.

Система *Cm** вообще плохо совместима с методом ПКМ, т.к. в ней отсутствует накопитель данных для размещения списка состояний.

Очень привлекательна с точки зрения возможного внедрения метода ПКМ система из Мюнстерского университета (п.I5). Её структура соответствует общему алгоритму метода ПКМ. Кроме того, система выполнена на процессорах, используемых в персональных ЭВМ средней мощности. Сомнение вызывает только возможность увеличения числа процессоров при той же системе управления передачей данных.

Система *SUPRENUM*, несмотря на солидный подход к проектированию и близость планируемых для решения задач к возможному применению метода ПКМ, не вызывает интереса, т.к. предназначена для численных методов решения уравнений в частных производных и в ней реализован принцип локального обмена сообщениями.

Как уже отмечалось, организация системы коммуникаций между процессорами и общей памятью остаётся серьёзной проблемой для отображения алгоритма ПКМ на вычислительную структуру, а системы с локальным распределением памяти и локальным обменом сообщениями

вообще могут быть не адекватны исследуемому физическому процессу. Возможным выходом является обращение к динамическим автоматным сетям ([I4] , [I5]), которые представляют собой один из вариантов синтезирования динамического дискретного пространства [I2] . Динамическая архитектура таких систем обеспечивает в принципе произвольные отношения между элементами, что даст возможность выполнения операций присваивания при любом множестве $\{n\}$ в алгоритме (EPS).

5. Заключение

Сегодня нет сомнений, что будущее численных методов исследования физических процессов связано с многопроцессорными вычислительными системами. Вопрос в том, что будет пониматься под численными методами. Авторы попытались изложить свою точку зрения и объяснить, что традиционная математическая физика по своей сути плохо согласуется с отображением реальных физических процессов в структуру функционирования компьютерных систем. Предлагаемый метод прямого компьютерного моделирования является, возможно, разумной альтернативой.

Ссылки:

[1] Lundstrom S.F. "Applications Considerations in the System Design of Highly Concurrent Multiprocessors". IEEE Trans. Comput., v. 36, № 11 (1987) p. 1292-1309

[2] Lawrie D. "Access and Alligment of Data in an Array Processor". IEEE Trans. Comput., v. 24, № 12 (1975) p. 1145-55.

[3] Gottlieb A. et al. "The NYU Ultracomputer-Designing an MIMD Shared Memory Parallel Computer". IEEE Trans. Comput., v. 32, № 2 (1983) p. 175-89.

[4] Edler A. et al., "Issues Related to MIMD Shared-Memory Computers: the NYU Ultracomputer Approach". 12th Annu. Intern. Symp. Computer Architecture (1985) p. 126-35.

[5] Rushfield N. "Atmospheric computations on highly parallel MIMD computers". Courant. Inst., NYU, Ultracomputer Note 22 (1981)

[6] Kalos M. et al. "Molecular Simulations of Equilibrium Properties". Courant Inst., NYU, Ultracomputer Note 27 (1981).

[7] Gottlieb A., Schwartz J.T. "Networks and Algorithms for Veri Large Scale Parallel Computations". Computer, v. 15, № 1 (1982) p. 27-36.

[8] Pfister G.F. et al. "The IBM Research Parallel Processor Prototype (RP3): Introduction and Architecture". Proc. Intern. Conf. Parallel Processing, (1985) p. 764-71.

[9] Остланд Н.С. и др. О применении сильносвязанных многопроцессорных систем для научных исследований // Параллельные вычисления .-М.: Наука, (1986).-Стр.295-341.

[10] Radehaus Ch et al. "Special Purpose Computer for Monlinear Differential Equations". Comput. Phys. Commun., v. 38, № 4 (1985) p. 345-50.

[11] Trottenberg U. "SUPRENUM-an MIMD-multiprocessor System for Multi-level Scientific Computing". Lect. Notes Comput. Sci., v. 237 (1986) p. 18-23.

[12] Александров В.В., Арсентьева А.В. Информация и развивающиеся структуры .-ЛИИАН, Л. (1984).

[13] Жаблон К., Симон Ж.-К. Применение ЭВМ для численного моделирования в физике .-М.: Наука, (1983).

[14] Торгашев В.А. Управление вычислительными процессами в машинах с динамической архитектурой"/ Препринт ЛИИАН № 33 (1982).

[15] Ponomarev V.M. et al. "Distributed Computing and Machines with Dynamic Architecture". Препринт ЛИИАН № 55 (1983)