УДК 004.3:.4

Распределённые вычислительные системы с программируемой структурой*)

В. Г. Хорошевский

В работе изложены концептуальные основы построения распределённых вычислительных систем (ВС) с программируемой структурой. Описаны модели функционирования ВС и континуальный метод расчёта показателей их эффективности. Приведены результаты по анализу и синтезу структур распределённых ВС. Предложены стратегии обеспечения стохастически оптимального функционирования распределённых ВС в режимах параллельного мультипрограммирования. Рассмотрены функциональные структуры реализованных ВС с программируемой структурой: “Минск-222”, МИНИМАКС, СУММА, МИКРОС и пространственно-распределённой мультикластерной ВС. Показано, что рассматриваемая концепция позволяет создавать технико-экономически эффективные и живучие суперВС.

Ключевые слова: вычислительная система, анализ эффективности, структура, параллельное мультипрограммирование.

Оглавление

*) Работа выполнена при поддержке Совета по грантам Президента РФ (ведущая научная школа НШ-5176.2010.9) и Российского фонда фундаментальных исследований (гранты 09-07-12016, 09-07-13534, 09-07-90403, 10-07-00157).

1. Введение

Десять лет, прошедшие с момента появления первой ЭВМ, позволили выявить пределы в развитии средств обработки информации, базирующихся на концептуальной машине Дж. фон Неймана, на модели вычислителя [1 – 14]. Функционирование таких средств основано на имитации процесса решения задачи специалистом-вычислителем и на принципах последовательного выполнения операций, фиксированности логической структуры и конструктивной неоднородности [15 – 17]. Исследователи и проектировщики средств обработки информации уже в начале 60-х годов XX столетия пришли к ясному пониманию необходимости технической реализации новых принципов обработки информации.

Исследования по созданию вычислительных средств, базирующихся на модели коллектива вычислителей, были начаты в Институте математики (ИМ) Сибирского отделения АН СССР в начале 1960-х годов по инициативе математика и механика С.Л. Соболева (1908 – 89; академик АН СССР с 1939 г.). Непосредственным руководителем работ стал специалист в области вычислительной техники Э.В. Евреинов (1928; доктор технических наук с 1968 г., профессор с 1972 г.). Успех в исследованиях ИМ СО АН СССР был бы немыслим, если бы они не опирались на достижения советских и западных научно-технических школ и прежде всего школы основоположника отечественной вычислительной техники С.А. Лебедева (1902 – 74; академик АН СССР с 1953 г.). Особенно плодотворное влияние и поддержку в становлении и развитии направления “Вычислительные системы с программируемой структурой” оказали академики С.Л. Соболев, Н.Н. Яненко (1921 – 84; академик АН СССР с 1970 г.), А.В. Ржанов (1920 – 2000; академик АН СССР с 1984 г.), А.С. Алексеев (1928 – 2007; академик АН СССР с 1984 г.).

Первая работа сотрудников ИМ СО АН СССР [18] о возможности построения вычислительных систем (ВС) высокой производительности опередила американские публикации в данной области примерно на 6 месяцев. В середине 1960-х годов выходит в свет монография [19], обобщающая первые результаты работ ИМ СО АН СССР по функциональным структурам вычислительных систем и параллельному программированию; под руководством Э.В. Евреинова создаются первые ВС [15]: “Минск-222” (1965 – 1966 гг.) и управляющая ВС [15] для автоматизации научных исследований (1964 – 1967 гг.). К началу 1970-х годов завершается формирование концепции вычислительных систем с программируемой структурой как средств обработки информации, основанных на модели коллектива вычислителей. Уместно заметить, что первоначальное название рассматриваемых средств – “Однородные вычислительные системы” [19], в конце 1970-х годов закрепляется название “ВС с программируемой структурой” (см. [15], стр. 26), так как оно точнее отражает архитектурные воз-

можности систем – коллективов вычислителей. “Однородные ВС” и “ВС с программируемой структурой” следует рассматривать как синонимические термины.

Начиная с 70-х годов XX столетия теоретические и проектные работы в Сибирском отделении АН СССР (ныне СО РАН) по вычислительным системам с программируемой структурой ведутся под руководством одного из разработчиков первой в мире ВС с программируемой структурой “Минск-222” В.Г. Хорошевского (1940; д.т.н. с 1974 г., чл.-корр. РАН с 2000 г.). Главными направлениями работ становятся:

- архитектура вычислительных систем;
- теория структур ВС: анализ и синтез структур коммуникационных сетей ВС;
- теория функционирования ВС: организация оптимального (субоптимального, стохастически оптимального) функционирования ВС в моно- и мультипрограммных режимах (обработки наборов и обслуживании потоков параллельных задач);
- надёжность и живучесть (потенциальная и структурная) ВС;
- самодиагностика и самоконтроль ВС;
- отказоустойчивые параллельные вычислительные технологии;
- проектирование и конфигурирование вычислительных систем;
- распределённые операционные системы;
- системы параллельного программирования;
- параллельные алгоритмы и программы для решения прикладных задач.

Работы по вычислительным системам из академической сферы распространяются в промышленность, под руководством автора данной статьи создаётся ряд систем: МИНИМАКС (1975 г.), СУММА (1976 г.), МИКРОС-1 (1986 г.), МИКРОС-2 (1992 г.), МИКРОС-Т (1996 г.). Выходит в свет большое число публикаций как сотрудников СО АН СССР (СО РАН), так и других организаций. В 1978 г. издательством “Наука” публикуется монография [15]. Академик С.Л. Соболев дал (23.12.1977 г.) следующий отзыв:

“Книга – фундаментальный труд по теории и практике высокопроизводительных систем, основанных на новых принципах обработки информации. Созданное и развитое авторами научное направление однородных вычислительных систем является стержнем книги. Концепция однородных вычислительных систем позволяет в максимальной степени исчерпать современные достижения технологии микропроцессоров.

Книга представляет большой интерес для широкого круга учёных и специалистов промышленности”.

Значимость разделов этой книги, посвящённых теории функционирования вычислительных систем, была оценена с достаточной полнотой только через 20 – 25 лет после её опубликования. Это объясняется тем, что лишь в конце XX столетия появились реальные условия для создания вычислительных систем действительно с массовым параллелизмом.

2. Понятие о вычислительных системах с программируемой структурой

Вычислительные системы (ВС) с программируемой структурой – это распределённые средства обработки информации, основанные на модели коллектива вычислителей [15]. В таких ВС нет единого функционально и конструктивно реализованного устройства: все компоненты (устройство управления, процессор и память) являются распределёнными. Тип архитектуры ВС – MIMD; в системах заложена возможность программной перенастройки архитектуры MIMD в архитектуры MISD или SIMD.

Основная функционально-структурная единица вычислительных ресурсов в системах рассматриваемого класса – это *элементарная машина* (ЭМ). В качестве ЭМ могут быть использованы ЭВМ, вычислительные ядра, многоядерные микропроцессоры, вычислительные узлы (композиции микропроцессоров), оснащённые средствами межмашинной коммутации. Допускается конфигурирование ВС с произвольным числом ЭМ. Следовательно, ВС с программируемой структурой относятся к масштабируемым средствам обработки информации

и допускают формирование конфигураций с массовым параллелизмом (Scalable Massively Parallel Architecture Computer Systems).

В системах с программируемой структурой диалектически сочетаются архитектурные свойства универсальных и специализированных средств обработки информации. Рассматриваемые ВС – это универсальные параллельные компьютеры, которые способны программно настраиваться под структуру и параметры решаемых задач. Отмеченные архитектурные свойства ВС с программируемой структурой оценены математиком и механиком Н.Н. Яненко: *"Чем шире класс задач, охватываемой специализированной машиной, тем сложнее её структура, и как наиболее совершенную форму ЭВМ следует рассматривать ЭВМ с перестраиваемой архитектурой"* [20].

При построении ВС с программируемой структурой доминирующими являются следующие три принципа:

1) массовый параллелизм (параллельность выполнения большого числа операций);

2) программируемость (автоматическая перестраиваемость или реконфигурируемость) структуры;

3) конструктивная *однородность.*

Следует подчеркнуть, что принцип программируемости структуры ВС является таким же важным, каким в своё время были предложения С.А. Лебедева и Дж. фон Неймана относительно организации в ЭВМ автоматической модификации программ. *Принцип программируемости структуры требует, чтобы в ВС была реализована возможность "хранения" программного описания функциональной структуры и программной её модификации (перенастройки) с целью достижения адекватности структурам и параметрам решаемых задач.*

Выделяют подкласс пространственно-распределённых ВС. В него включаются макросистемы – системы сложной конфигурации, в которых в качестве функциональных элементов выступают пространственно-рассредоточенные вычислительные средства, основанные на моделях вычислителя и коллектива вычислителей, и сети связи, обеспечивающие взаимный теледоступ между средствами обработки информации. *Пространственно-распределённая ВС – это объединение географически удалённых друг от друга сосредоточенных ВС, основанное на принципах:*

1) *параллельности* функционирования вычислительных ресурсов (т.е. способности нескольких или всех сосредоточенных ВС совместно и одновременно решать одну сложную задачу, представленную параллельной программой);

2) превалирующего использования массовых аппаратурно-программных средств и существующих компьютерных сетей, включая Internet;

3) совместимости (информационной и программной) сосредоточенных ВС.

Пространственно-распределённые ВС в общем случае предназначаются для реализации параллельных программ решения задач произвольной сложности (с произвольным объёмом вычислений) в *монопрограммном* и *мультипрограммных режимах* (на рассредоточенных в пространстве ресурсах). Они должны быть приспособленными и для выполнения функций, присущих вычислительным сетям.

Первая пространственно-распределённая ВС с программируемой структурой АСТРА [15] была создана ИМ СО АН СССР и Новосибирским электротехническим институтом МВ и ССО РСФСР; работы по проектированию ВС были начаты в 1970 г., а первая модель была сдана в эксплуатацию в 1972 г. Было построено семейство моделей АСТРА, среди которых были городские и междугородные конфигурации (Новосибирск, Москва). Модели формировались из ЭВМ "Минск-32" и использовали телефонные каналы связи. Были выполнены проекты распределённых ВС и на базе машин третьего поколения семейства ЕС ЭВМ [15].

Целесообразно подчеркнуть, что современное информационное пространство – это локальные и распределённые корпоративные вычислительные сети и глобальная сеть Internet. Дальнейшим шагом в развитии архитектуры сетей должны были стать распределённые вы-

числительные системы, способные реализовать параллельные алгоритмы решения сложной задачи на географически рассредоточенных ЭВМ и ВС. В самом деле, в конце прошлого столетия получают распространение распределённые информационные технологии: метакомпьютинг (Metacomputing), масштабируемый (Scalable), глобальный (Global), Интернет (Internet) компьютинг и P2P-компьютинг (Peer-to-peer Computing).

В 1997 году Национальный научный фонд США инициировал программу развития информационных технологий PACI (Partnerships for Advanced Computational Infrastructure). В ходе работ по программе PACI возникла концепция GRID (Global Resource Information Distribution), изначально рассматриваемая как аналогия распределённой электротехнической инфраструктуры. Аксиоматически ясно, что GRID-системы должны предоставлять распределённым пользователям разнообразные услуги по обработке и хранению данных и, главное, они должны быть способны реализовать параллельные алгоритмы решения суперсложных задач на своих рассредоточенных ресурсах.

Следует особо подчеркнуть, что для любого этапа развития индустрии обработки информации характерно создание суперкомпьютеров. К последним относятся вычислительные средства, обладающие рекордной эффективностью (производительностью, надёжностью, живучестью и технико-экономической эффективностью) для фиксированного этапа. Суперкомпьютеры могут быть основаны на любой из архитектурных парадигм, однако в них всегда присутствует параллелизм. Если придерживаться терминологической строгости, то их следует называть *суперВС*. Архитектура современных высокопроизводительных ВС, как правило, отличается от своих изначальных канонов. Архитектура одних и тех же систем в зависимости от уровня рассмотрения их функциональных структур может выглядеть и как MISD, и как SIMD, и как MIMD. Таким образом, можно констатировать, что *мультиархитектура стала парадигмой при конструировании высокопроизводительных ВС.*

Обобщая опыт развития индустрии обработки информации, можно заключить, что независимо от изначальной архитектурной парадигмы фирмы-создатели суперкомпьютеров к началу XXI столетия перешли на платформу ВС с программируемой структурой.

Ниже будет представлено несколько фрагментов теории функционирования большемасштабных распределённых ВС; в последнем разделе будут описаны функциональные структуры советских и российских систем с программируемой структурой.

3. Анализ эффективности функционирования большемасштабных распределённых ВС

Одним из важнейших разделов теории функционирования распределённых ВС является анализ их эффективности (производительности, надёжности, живучести и экономичности) и осуществимости параллельного решения задач. Сложность и специфичность такого анализа обусловливается рядом факторов.

Масштабируемость и большемасштабность ВС. Современные суперкомпьютеры – это ВС с массовым параллелизмом, их архитектура допускает варьирование количества ЭМ (вычислительных ядер) в пределах $10^2 - 10^6$. Например, системы Jaguar Cray XT5-HE и IBM RoadRunner обладают быстродействием 2331 и 1375,8 TeraFLOPS и имеют в своём составе 224162 и 122400 вычислительных ядер соответственно. Конфигурация ВС IBM BlueGen/P с производительностью 1002.7 TeraFLOPS состоит из 294912 ядер. В будущих системах с производительностью порядка ExaFLOPS количество ЭМ может достичь значения 10^8.

Программируемость структуры ВС. Суперкомпьютеры уже обладают способностью самоадаптации, т.е. в них допускается программная организация множества подсистем и настройка виртуальных конфигураций, адекватных структурам и параметрам поступающих задач.

“Стохастичность” ВС. По своей природе ВС (из-за неабсолютной надёжности ресурсов) – стохастический объект, который предназначается в общем случае для обслуживания

вероятностных потоков заданий, представленных параллельными программами со случайными параметрами (числом ветвей, временем решения и т.п.).

Успех анализа большемасштабных распределённых ВС безусловно определяется моделями, описывающими их функционирование. Адекватность моделей изучаемым объектам – распределённым ВС – должна гарантировать простоту и удовлетворительную для практики точность расчётов показателей эффективности.

3.1. Виртуализация архитектуры распределённых ВС

Архитектура любого средства обработки информации раскрывается через совокупность его свойств и характеристик. Постоянное совершенствование архитектуры ВС, а ныне и острая потребность в системах, обладающих производительностью порядка $10^{12}-10^{18}$ опер./с, стимулируют развитие методов их анализа и организации функционирования. При этом комплексная проблема живучести распределённых ВС выдвинулась на передний план. Живучесть вычислительных систем является более ёмким понятием, чем надёжность ВС. Под *живучестью* понимается способность ВС (достигаемая программной организацией структуры и функционального взаимодействия между её компонентами) в любой момент функционирования использовать суммарную производительность всех исправных ресурсов для решения задач.

Вычислительное ядро любой высокопроизводительной ВС компонуется из однородных ЭМ. Будем говорить, что ВС находится в состоянии $k \in E_0^N = \{0,\ 1\ ,...,\ N\}$, если в ней имеется k работоспособных ЭМ.

Под *живучей ВС* понимается (виртуальная) конфигурация из N элементарных машин, в которой:

– указано минимально допустимое число n работоспособных ЭМ, гарантирующее производительность системы не менее требуемой;

– обеспечена возможность решения сложных (с большим числом операций) задач, представленных адаптирующимися параллельными программами;

– отказы любых ЭМ (вплоть до числа $N-n$) и восстановления отказавших машин приводят только к увеличению или уменьшению времени реализации параллельной программы;

– при изменении состояния $k=1,2,...,N$ производительность подчиняется следующему закону: $\Omega(k) = A_k \cdot \Delta(k-n) \cdot \varphi(k,\omega)$,

где A_k – коэффициент;

$$\Delta(k-n) = \begin{cases} 1\ , \text{ если} & k \geq n, \\ 0\ , \text{ если} & k < n, \end{cases}$$

ω – показатель производительности элементарной машины; $\varphi(k,\omega)$ – неубывающая функция от k и ω (как правило, $\varphi(k,\omega) = k\omega$). К *адаптирующимся* относятся программы, при реализации которых автоматически устанавливается число параллельных ветвей, равное числу (работоспособных) ЭМ в текущий момент времени (в последнее время такие программы иногда называют масштабируемыми).

Следует обратить внимание на то, что в живучей ВС число избыточных ЭМ – переменное и заключено между 0 и $N-n$; в ней нет резервирования, нет простоев исправных машин. Все исправные ЭМ такой ВС включаются в *вычислительное ядро* и участвуют в реализации параллельных процессов, что приводит к уменьшению времени решения задач.

Современные ВС, как правило, восстанавливаемые (или даже самовосстанавливающиеся). Будем считать, что ремонтные работы в ВС осуществляются некоторой *восстанавливающей системой*, состоящей из m устройств (ВУ), $1 \leq m \leq N$. Каждое ВУ в любой момент времени может производить ремонт только одной ЭМ. Будем полагать так-

же, что для формирования в системе виртуальных конфигураций имеются специальные (аппаратно-программные) средства, составляющие *реконфигуратор*. Он предназначается для выполнения следующих функций:

– исключения из вычислительного ядра отказавших машин и включения в него машин после их восстановления;

– формирования вычислительного ядра из оставшихся работоспособных ЭМ и вновь отремонтированных машин;

– преобразования адаптирующейся параллельной программы с целью достижения соответствия между числом её ветвей и числом машин вычислительного ядра;

– вложения преобразованной программы в ядро с новой структурой и организации её прохождения.

Вычислительные системы с программируемой структурой обладают гибкими структурными возможностями [15 – 19] для такой виртуализации. При этом живучесть ВС рассматривается в двух аспектах: потенциальном и структурном. При изучении потенциальных возможностей ВС особенности структуры или сети межмашинных связей в явном виде не учитываются и считается, что в системе обеспечиваются возможности для достижения необходимой связности k исправных ЭМ, $k \in \{n, n+1, ..., N\}$.

3.2. Континуальная модель функционирования ВС

Математическое ожидание $N(i,\ t)$ числа работоспособных машин определяет в момент времени $t \geq 0$ и производительность ВС, и ёмкость ёе распределённой памяти; i – начальное состояние ВС, очевидно, что $N(i,\ 0) = i,\ i \in E_0^N$. Именно $N(i,\ t)$ машин составляют в среднем в момент времени $t \geq 0$ вычислительное ядро ВС, а функция

$$\mathcal{N}(i,\ t) = N(i,\ t) / N,\ \ i \in E_0^N$$

характеризует потенциальную живучесть системы.

Функция $\mathcal{N}(i,\ t)$, безусловно, может быть рассчитана классически с использованием аппарата теории массового обслуживания. Такой расчёт трудоёмок, т.к. он связан с приближёнными вычислениями переходных вероятностей дискретных состояний ВС [16]. Указанных трудностей можно избежать, если учесть то, что высокопроизводительные ВС–большемасштабные (математически: $N \to \infty$). В самом деле, при исследовании потенциальной живучести вычислительной системы вместо рассмотрения всего её пространства состояний $E_0^N = \{0,\ 1,\ ...,\ N\}$, т.е. учёта функционирования каждой ЭМ, можно изучать поведение ВС в целом как ансамбля большого числа идентичных машин. Такой подход основывается на допущении, что в любой момент времени производительности вычислительной и восстанавливающей систем пропорциональны соответственно не случайному числу исправных ЭМ и не случайному числу занятых ВУ, а математическим ожиданиям соответствующих чисел. Допущение естественно для большемасштабных ВС (для систем с большим числом ЭМ и ВУ), в которых случайности, связанные с выходом ЭМ из строя или их восстановлением, либо с включением ВУ в ремонт ЭМ или освобождением ВУ, мало сказываются на значениях суммарной производительности систем. Эти значения для каждого момента времени оказываются близкими к средним значениям производительности систем. Случайности, связанные с выходом машин из строя и освобождением ВУ, сказываются, если число работоспособных ЭМ в ВС становится небольшим или если число занятых устройств в восстанавливающей системе становится близким к m. Однако вероятности таких событий при существующем уровне надёжности микропроцессоров (интенсивности отказов $\lambda \leq 10^{-8}\ 1/ч$) чрезвычайно малы.

На основании вышесказанного при изучении потенциальной живучести ВС за основу берётся стохастическая модель функционирования [21], представленная на рис. 1. Производи-

тельность ВС в любой момент времени $t \geq 0$ определяют $N(i,\ t)$ машин вычислительного ядра, т.е. те работоспособные ЭМ, которые непосредственно используются для реализации адаптирующейся параллельной программы. В случае отказа ЭМ “покидает” вычислительное ядро и берётся на учёт реконфигуратором ВС; λ–интенсивность отказов машины. Пусть $L'(i,\ t)$ – среднее число отказавших ЭМ, учитываемых реконфигуратором ВС в момент $t \geq 0,\ i \in E_0^N$. Реконфигуратор исключает из вычислительного ядра отказавшие машины, образует из оставшихся работоспособных ЭМ связную подсистему, сокращает число параллельных ветвей в адаптирующейся программе и организует её прохождение на вычислительном ядре с новой структурой. В результате выполнения таких функций реконфигуратор с интенсивностью ν' “переключает” отказавшие ЭМ из ядра в число машин, подлежащих восстановлению. Пусть $K(i,\ t)$ – математическое ожидание числа отказавших машин, учитываемых восстанавливающей системой.

Пусть $M(i,\ t)$ – среднее число устройств, занятых восстановлением отказавших ЭМ; μ – интенсивность восстановления отказавших ЭМ одним ВУ. После восстановления элементарные машины берутся на учёт реконфигуратором ВС. Пусть $L''(i,\ t)$ – среднее число восстановленных ЭМ, взятых на учёт реконфигуратором ВС. Включение восстановленных ЭМ в состав ядра ВС осуществляется с интенсивностью ν''. Среднее время $1/\nu''$ такого включения зависит от времени образования связного подмножества машин из существовавшего ядра и восстановленных ЭМ, времени перенастройки параллельной программы на большее число ветвей и времени запуска этой программы на вновь созданном ядре.

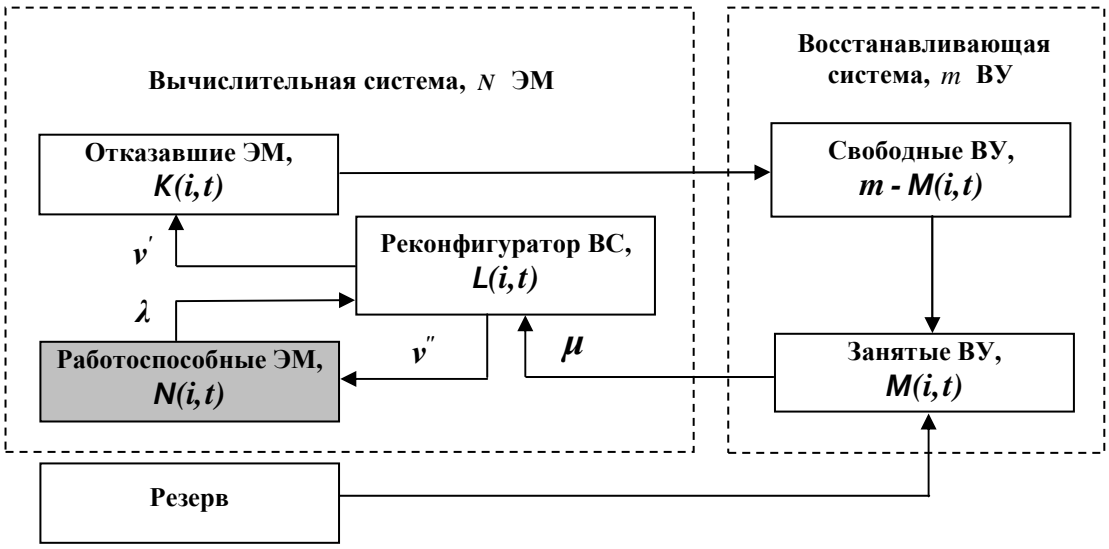


Рис. 1. Модель функционирования живучей ВС

При изучении большемасштабных реконфигурируемых ВС, в общем случае, используются виртуальные восстанавливающие устройства и “ремонт” обнаруженных отказавших ЭМ сводится к их замене на машины из резерва. В этих случаях интенсивность μ воспринимается как среднее число машин резерва, включаемых в единицу времени одним виртуальным ВУ в состав вычислительной системы.

Очевидно следующее равенство

$$L(i,\ t) + K(i,\ t) + N(i,\ t) = N,$$

где $L(i,\ t) = L'(i,\ t) + L''(i,\ t)$ – среднее число ЭМ, с которыми работает реконфигуратор ВС.

3.3. Анализ потенциальной живучести ВС

Произведем расчёт функции $\mathcal{N}(i,\ t)$ потенциальной живучести ВС. Ясно, что при использовании континуальной модели функционирования ВС естественно пойти по пути вывода разностных и дифференциальных уравнений непосредственно для $N(i,\ t)$ и $M(i,\ t)$. В качестве иллюстрации здесь рассмотрим наиболее вероятную для практики ситуацию:

$$K(i,\ t) = M(i,\ t) \le m,$$

$$i \in \{N-m,\ N-m+1\ ,...,\ N\},$$

которая, как легко показать, имеет место при выполнении неравенства [16]

$$N\lambda \le m(\lambda+\mu+\lambda\mu/\nu)$$

Тогда

$$N(i,\ t) = \frac{N\mu\nu}{\alpha_1\alpha_2} + \left[\frac{N\mu\nu}{\alpha_1} + (i+j)\nu + i(\alpha_1+\mu)\right]E_1 + \left[\frac{N\mu\nu}{\alpha_1} + (i+j)\nu + i(\alpha_2+\mu)\right]E_2,$$

где $\nu = \nu' + \nu''$,

$$\alpha_{1,2} = -\frac{1}{2}(\lambda+\mu+\nu) \mp \frac{1}{2}\sqrt{\lambda^2+\mu^2+\nu^2-2(\lambda\mu+\lambda\nu+\mu\nu)},$$

$$j = L(i,0),\ \ E_1 = \frac{e^{\alpha_1 t}}{\alpha_1-\alpha_2},\ E_2\frac{e^{\alpha_2 t}}{\alpha_2-\alpha_1}.$$

Полученные формулы позволяют оценить поведение ВС в переходном режиме. Для высокопроизводительных систем естественен стационарный режим эксплуатации:

$$N = \lim_{t\to\infty} N(i,t) = \frac{N\mu\nu}{\lambda\mu+\lambda\nu+\mu\nu},$$

$$M = \lim_{t\to\infty} M(i,t) = \frac{N\lambda\nu}{\lambda\mu+\lambda\nu+\mu\nu},$$

$$L = \lim_{t\to\infty} L(i,t) = N-N-M,$$

$$K = \lim_{t\to\infty} K(i,t) = M.$$

Неравенство $\lambda < \mu << \nu$, как известно, имеет место всегда. Если учесть последнее, то получим предельно простые формулы:

$$N = N\mu(\lambda+\mu)^{-1},\ M = N\lambda(\lambda+\mu)^{-1}.$$

Описанный континуальный подход одинаково применим и для анализа живучести и технико-экономической эффективности ВС, и для анализа осуществимости решения сложных задач, представленных параллельными программами [16, 17]. Простота расчётных формул приводит к нетрудоёмкой технологии экспресс-анализа эффективности большемасштабных распределённых ВС, систем с произвольным числом элементарных машин. Экспресс-анализ возможен и для стационарного, и для переходного режимов функционирования ВС.

4. Структуры вычислительных систем

Теория структур ВС – раздел, в котором рассматриваются формальные описания, осуществляется анализ и синтез сетей связи между элементарными машинами (вычислительными узлами).

Под структурой (Structure, Topology) вычислительной системы понимается граф G, вершинам которого сопоставлены элементарные машины, а рёбрам – линии связи между ними. Проблема анализа и выбора (синтеза) структур ВС не является тривиальной. В самом деле, универсальное решение – структура в виде полного графа (Complete graph), однако такая структура практически реализуема при небольшом числе ЭМ. Для достижения произво-

дительности ВС в диапазоне $10^{12}...10^{18}$ опер./с при существующих интегральных технологиях необходимо $10^{2}...10^{8}$ ЭМ. Следовательно, полносвязные структуры суперВС практически не реализуемы (хотя бы из-за ограничений существующих технологий БИС). Нужны такие структуры ВС, которые были бы существенно проще полных графов и которые бы позволяли достичь "эффективного" решения задач (с учётом ненадёжности компонентов ВС).

4.1. Требования, предъявляемые к структурам ВС

Остановимся на основных требованиях, которые предъявляются к структурам современных вычислительных систем.

1. *Простота вложения параллельных алгоритмов решения сложных задач в структуру ВС.* Структура ВС должна быть адекватна достаточно широкому классу решаемых задач; настройка проблемно-ориентированных виртуальных конфигураций и реализация основных схем обмена информацией между ЭМ [17] не должны быть связаны со значительными накладными расходами.

2. *Удобство адресации элементарных машин и "переноса" подсистем в пределах ВС.* Вычислительная система должна предоставлять возможность пользователям создавать параллельные программы с виртуальными адресами ЭМ. Следовательно, структура ВС должна позволять реализовать простейший механизм преобразования виртуальных адресов ЭМ в реальные (физические) адреса. Необходимость организации одновременного решения нескольких задач на ВС (т. е. необходимость разделения "пространства" элементарных машин между задачами) обосновывает требование простоты перемещения подсистем в пределах системы (при сохранении их топологических свойств). При выполнении данных требований будет достигнута эффективность ВС при работе как в моно-, так и мультипрограммных режимах. Кроме того, следует отметить, что данные требования являются необходимыми условиями для создания отказоустойчивых параллельных программ.

3. *Осуществимость принципа близкодействия и минимума задержек при межмашинных передачах информации в ВС.* Принцип близкодействия предопределяет реализацию обменов информацией между "удалёнными" друг от друга ЭМ через промежуточные машины системы. Следовательно, в условиях ограниченности числа связей у каждой ЭМ структура должна обеспечивать минимум задержек (латентности) при транзитных передачах информации.

4. *Масштабируемость и большемасштабность структуры ВС.* Для формирования конфигураций ВС с заданной эффективностью необходимо, чтобы структура обладала способностью к наращиванию и сокращению числа вершин (машин). Изменение числа ЭМ в ВС не должно приводить к коренным перекоммутациям между машинами и (или) к необходимости изменения числа связей для любых ЭМ.

5. *Коммутируемость структуры ВС.* Система должна быть приспособлена к реализации коллективных межмашинных обменов информацией, а также к формированию в ней подсистем (для мультипрограммирования). Следовательно, структура ВС должна обладать способностью осуществлять заданное число одновременных непересекающихся взаимодействий между элементарными машинами.

6. *Живучесть структуры ВС.* Важным требованием к ВС в целом является обеспечение работоспособности при отказе её компонентов или даже подсистем. Основой функциональной целостности ВС как коллектива ЭМ является живучесть структуры. Под последним понимается способность структуры ВС обеспечить связность требуемого числа работоспособных ЭМ в системе при ненадёжных линиях межмашинных связей.

7. *Технологичность структур ВС.* Структура сети межмашинных связей ВС не должна предъявлять особых требований к элементной базе, к технологии изготовления микропроцессорных БИС. Системы должны быть восприимчивы к массовой технологии, их "вычислительное ядро" должно формироваться из однородных микропроцессорных БИС. Последнее позволит достичь приемлемых значений технико-экономических показателей вычислительных систем.

Анализ путей удовлетворения перечисленным требованиям приводит к безальтернативному выбору *однородных* (или регулярных, т. е. описываемых однородными графами) *структур* для формирования ВС. В современных условиях при формировании суперВС используют разнообразные структуры; в системе, как правило, несколько межмашинных сетей, но среди них превалируют однородные.

4.2. Структурные характеристики ВС

Структурные задержки при передачах информации между машинами ВС определяются расстоянием между вершинами структуры, сопоставленными взаимодействующим ЭМ. Для оценки структурных задержек в ВС используются диаметр d и средний диаметр $\bar{d}$ структуры. *Диаметр* структуры ВС – максимальное расстояние, определённое на множестве кратчайших путей между вершинами всевозможных их пар:

$$d = \max_{i,j}\{d_{ij}\}, \tag{1}$$

средний диаметр

$$\bar{d} = (N-1)^{-1}\sum_{l=1}^{d} l n_l, \tag{2}$$

где d_{ij} – расстояние, т. е. минимальное число рёбер, образующих путь из вершины i в вершину j; $i, j \in \{0, 1, ..., N-1\}$; n_l – число вершин, находящихся на расстоянии l от любой выделенной вершины графа G.

Показателем, оценивающим *структурную коммутируемость* ВС, является вектор-функция

$$\mathcal{K}(G, s, s') = \{\mathcal{K}_h(G, s, s')\}, \quad h \in \{1, 2, ..., [N/2]\}, \tag{3}$$

в которой координата $\mathcal{K}_h(G, s, s')$ есть вероятность реализации в системе при заданных структуре G и коэффициентах готовности s и s' соответственно одной ЭМ и линии связи h одновременных непересекающихся межмашинных взаимодействий (обменов информацией между ЭМ); $[N/2]$ – целая часть числа $N/2$.

Структурная живучесть ВС оценивается вектор-функцией

$$\mathcal{L}(G, s, s') = \{\mathcal{L}_r(G, s, s')\}, \quad r \in E_2^N = \{2, 3, ..., N\}. \tag{4}$$

Здесь $\mathcal{L}_r(G, s, s')$ – вероятность существования подсистемы ранга r (т. е. подмножества из r работоспособных ЭМ, связность которых устанавливается через работоспособные линии) при заданных структуре G, коэффициентах готовности s и s' ЭМ и линии связи соответственно.

Введённые показатели позволяют осуществить с достаточной полнотой анализ структурных возможностей ВС и анализ структурной живучести ВС, в частности. Отметим прикладное значение введённых показателей.

Диаметр d и средний диаметр $\bar{d}$ – это структурные характеристики, связанные с производительностью ВС. Диаметр структуры ВС определяет максимально необходимое число транзитных вершин при межмашинных обменах информации, следовательно, он является количественной характеристикой для максимальных структурных задержек. Средний диаметр структуры ВС можно использовать в качестве показателя, оценивающего средние задержки при выполнении межмашинных взаимодействий.

По значениям координат *вектор-функции структурной коммутируемости* $\mathcal{K}(G, s, s')$ можно судить относительно возможностей ВС по реализации обменов информацией между её машинами. Данная характеристика важна для анализа структур ВС, работающих в мультипрограммных режимах. В этих режимах система программным способом разбивается

на подсистемы. Максимальное число подсистем в ВС определяется, очевидно, величиной $[N/2]$. В случае мультипрограммирования обмены совершаются в пределах каждой из подсистем. Тогда, например, координата $\mathcal{K}_h(G, s, s')$ будет информировать о приспособленности структуры к генерации в пределах ВС h подсистем или к одновременному решению на ВС h задач, $1 \le h \le [N/2]$.

Координаты *вектор-функции структурной живучести* ВС характеризуют приспособленность системы в условиях отказов ЭМ и линий связи к порождению подсистем тех или иных рангов, следовательно, её приспособленность к решению задач заданной сложности. В частности, координата $\mathcal{L}_r(G, s, s')$ определяет возможности структуры по реализации на ВС задач ранга r, т. е. сложных задач, представленных параллельными программами с числом ветвей, равным $r \in \{2, 3, ..., N\}$.

Подмножество координат

$$\{\mathcal{L}_{r^{\circ}}(G, s, s'),\ \mathcal{L}_{r^{\circ}+1}(G, s, s'), ..., \mathcal{L}_{r^{*}}(G, s, s')\}$$

характеризует приспособленность ВС к выполнению на ней *адаптирующихся параллельных программ*, допускающих автоматическое изменение своих рангов от r° до r^{*}, где $1 < r^{\circ}, r^{*} \le N$. При $r^{\circ} = n$ и $r^{*} = N$ вектор-функция структурной живучести ВС будет характеризовать её способность к организации в ней виртуальных образований, обладающих живучестью.

4.3. Простейшие структуры ВС

В суперВС предшествующих десятилетий использовались нульмерные, одномерные и двумерные простейшие структуры. В первом случае структура сети связей “вырождена”, взаимодействие между машинами ВС осуществляется через общую шину (Common bus, Unibus). В случае одномерных структур (“линейки” – Linear graph или “кольца” – Ring) обеспечивается связь каждой ЭМ с двумя другими (соседними) машинами. В нульмерных структурах имеется общий ресурс – шина, в одномерных же структурах этот ресурс трансформируется в распределённый, т. е. в локальные связи между вычислителями. Следовательно, характеристики последних структур существенно лучше, чем у нульмерных.

Увеличение размерности структуры повышает структурную живучесть ВС. В самом деле, двумерные структуры предоставляют каждой ЭМ непосредственную связь с четырьмя соседними. В качестве примеров двумерных структур может служить 2D-решётка (Two-dimensional Grid) и 2D-тор (Two-dimensional torus). Следовательно, в системах с двумерной структурой при отказах некоторых ЭМ и (или) связей между ними сохраняется возможность организации связных подмножеств исправных машин.

В n-мерных структурах каждая ЭМ связана с $2n$ соседними машинами. Существуют технико-экономические и технологические ограничения в наращивании размерности структуры ВС.

4.4. Гиперкубические и тороидальные структуры ВС

Гиперкубы, или структуры в виде булевых n-мерных кубов, а также торы нашли широкое применение при построении современных высокопроизводительных ВС с массовым параллелизмом. *Гиперкуб* (Hypercube) по определению – это однородный граф, для которого справедливо

$$n = \log_2 N,$$

где N – количество вершин; n – число рёбер, выходящих из каждой вершины; n называют также *размерностью* гиперкуба. Каждая ЭМ в гиперкубической ВС имеет связь ровно с n другими ЭМ. Гиперкуб размерности n называют также nD-кубом.

Если вершины гиперкуба пронумеровать от 0 до $N-1$ в двоичной системе счисления так, что каждый разряд соответствует одному из n направлений, то получим булев n-мерный куб.

Тор – это многомерная решётка, в которой имеют место отождествления связей граничных вершин в каждом из направлений. Простейший вариант – $2D$-тор – образуется из решётки путём отождествления связей граничных вершин в каждой "строке" и в каждом "столбце". Четырёхмерный гиперкуб ($N=16,\quad n=4$) является также $2D$-тором.

Возможности распространённых структур ВС отражены в табл. 1.

Таблица 1

Показатель	Тип структуры ВС					
	Полный граф	Линейка	Кольцо	2D-решётка	2D-тор	Гиперкуб
Диаметр	1	$N-1$	$\lfloor N/2 \rfloor^{*}$	$2(\sqrt{N}-1)$	$2(\sqrt{N}-2)$	$\log_2 N$
Количество рёбер	$N(N-1)/2$	$N-1$	N	$2(N-\sqrt{N})$	$2N$	$(N\log_2 N)/2$

* $\lfloor N \rfloor$ – целая часть числа N.

Из табл. 1. следует, что структуры с меньшим диаметром имеют большее количество рёбер; удвоение числа вершин в гиперкубе увеличивает его диаметр только на единицу.

4.5. Перспективные структуры ВС

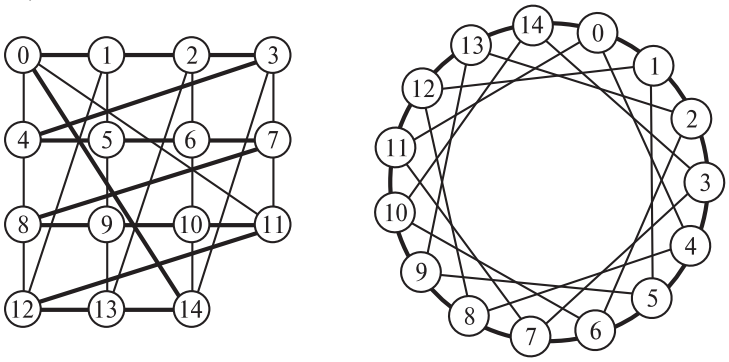
Рассмотрим структуры, удовлетворяющие требованиям, изложенным выше, т. е. перспективные для формирования масштабируемых и большемассштабных вычислительных систем (в частности, ВС с программируемой структурой).

В компьютерной индустрии получили распространение n-мерные структуры ВС, известные сейчас как циркулянтные (Circulant Structures). Впервые они были определены и исследованы в Отделе вычислительных систем ИМ СО АН СССР в начале 1970-х годов и первоначально назывались *D_n-графами* [17]. По определению, *D_n-граф*, или *циркулянтная структура*, есть граф G вида: $\{N; \omega_1, \omega_2, ..., \omega_n\}$, в котором:

- N – количество вершин или порядок графа;
- вершины помечены целыми числами i по модулю N, следовательно, $i \in \{0, 1, ..., N-1\}$;
- вершина i соединена ребром (или является смежной) с вершинами $i \pm \omega_1, i \pm \omega_2, ..., i \pm \omega_n \pmod N$;
- $\{\omega_1, \omega_2, ..., \omega_n\}$ – множество целых чисел, называемых образующими, таких, что $0 < \omega_1 < \omega_2 < ... < \omega_n < (N+1)/2$, а для чисел $N; \omega_1, \omega_2, ..., \omega_n$ наибольшим общим делителем является 1;
- n – размерность графа;
- $2n$ – степень вершины в графе.

Графы G вида $\{N; 1, \omega_2, ..., \omega_n\}$, т. е. D_n-графы или циркулянты с единичной образующей (Loop Networks – петлевые структуры), интенсивно изучаются в последнее время. Циркулянтные структуры $\{N; 1, \omega_2\}$ широко внедрены в практику ВС, и читатели хорошо знают, что циркулянт $\{64; 1, 8\}$ отражает структуру квадранта ВС ILLIAC-IV [15, 17]. Равновероятно

используют изображения циркулянт с единичной образующей в виде и двумерных “матриц”, и хордовых колец (рис.2).



а *б*

Рис. 2. D_2-граф: {15; 1, 4}:
а – двумерная матрица; *б* – хордовое кольцо

Целые числа $i \in \{0, 1, 2, ..., N-1\}$, отмечающие вершины D_n-графа, называют *адресами. Адресация* вершин в таких структурах *называется диофантовой* (в честь древнегреческого математика из Александрии Диофанта, Diophantos, III в.). *В циркулянтных структурах при полном переносе какой-либо подструктуры* (всех вершин подструктуры на одно и то же расстояние в одном из направлений) *сохраняются все её свойства и адресация вершин*. Таким образом, при диофантовой адресации элементарных машин ВС можно простыми средствами реконфигурации осуществить виртуальную адресацию вершин-машин и, следовательно:

1) создавать отказоустойчивые параллельные программы, не ориентированные на физические номера машин;

2) реализовывать мультипрограммные режимы обработки информации;

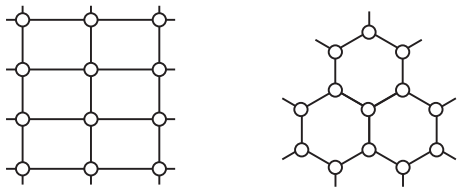
3) исключать отказавшие вершины-машины из подсистем, а значит, обеспечить живучесть ВС.

Пусть параллельная программа реализуется на некоторой подсистеме ВС (т. е. на подграфе в пределах D_n-графа). Далее, пусть i – физический номер элементарной машины (вершины D_n-графа), исключаемой из подсистемы (подграфа), а j – номер ЭМ, включаемой в неё, $i, j \in \{0, 1, ..., N-1\}$. Тогда, очевидно, алгоритм преобразования виртуального адреса α машины, используемого в параллельной программе, сводится к его изменению по формуле

$$\alpha := [\alpha + (j - i)] \mod N,$$

где $\alpha, i, j \in \{0, 1, ..., N-1\}$.

В качестве структур ВС, допускающих масштабирование (изменение числа машин) без коренной перекоммутации уже имеющихся межмашинных связей, используются *Л*(*N*, *v*, *g*)-*графы*. В такие графы [22] вкладываются D_n-графы; *Л*(*N*, *v*, *g*)-граф – это неориентированный однородный граф с числом и степенями вершин соответственно *N* и *v* и значением обхвата *g* (рис. 3). В *Л*(*N*, *v*, *g*)-графах каждая вершина при $v \geq 3$ входит в не менее *v* кратчайших простых циклов длиной *g* (длина кратчайшего цикла в графе называется *обхватом*). При $v = 2$ *Л*(*N*, *v*, *g*)-граф является простым циклом с *N* вершинами.



а *б*

Рис. 3. Фрагменты *Л*(*N*, *v*, *g*)-графов:
а – $v = 4$, $g = 4$; *б* – $v = 3$, $g = 6$

4.6. Анализ и синтез структур ВС

Введённые показатели (1) – (4) в равной степени пригодны и для анализа, и для синтеза структур ВС. Как в том, так и в другом случае требуется осуществлять расчёт значений структурных показателей ВС. Получение аналитических выражений для координат вектор-функций структурной коммутируемости ВС (3) и структурной живучести (4) является задачей, разрешимой лишь для частных случаев. Для расчёта этих показателей используют метод статистического моделирования (Монте-Карло).

Проблема синтеза структур заключается в поиске таких графов G^*, которые реальные (физические) конфигурации ВС делали бы максимально приспособленными для программирования виртуальных конфигураций. Если при проектировании ВС преследуется цель её адаптации под какую-либо область применения, под класс решаемых задач, то физическая структура G^* должна быть максимально приспособлена для программной настройки проблемно-ориентированных виртуальных конфигураций. Если же требуется достичь живучести ВС, то G^* должна быть адаптирована под программирование живучих виртуальных конфигураций. Или же, если при создании ВС требуется максимизировать эффективность использования ЭМ, то определяется структура G^*, которая минимизирует задержки (Latency – латентность) при транзитных передачах информации между ЭМ.

На этапе выбора структур ВС заметную роль играют интуитивные соображения. Так, например, было очевидно, что кольцевые и тороидальные структуры ВС обладают большей живучестью, чем “линейка” и “решётка” соответственно.

Проблема синтеза структур ВС может быть сформулирована с ориентацией на любой из структурных показателей (1) – (4). Здесь мы дадим следующую постановку: найти структуру G^*, которая обеспечивала бы максимум координаты вектор-функции структурной живучести (4), т. е.

$$\max_{G} \mathcal{L}_r(G, s, s') = \mathcal{L}_r(G^*) \qquad (5)$$

при заданных значениях N, r, v, s, s'. Структура G^*, для которой выполняется (5), называется *оптимальной*. В упрощённой постановке можно ограничиться поиском G^* в некотором классе структур, например, в классе D_n- или $Л(N, v, g)$-графов.

К сложным относится проблема синтеза оптимальных структур большемасштабных ВС, она практически решается при помощи статистического моделирования и, следовательно, с использованием мощных вычислительных средств. Трудоёмкость поиска G^* можно заметно снизить, если воспользоваться двумя нижеприведёнными гипотезами.

***Гипотеза* 1**. *Структура G^*, при которой достигается $\mathcal{L}_N(G^*)$ – максимум живучести ВС, обеспечивает и $\mathcal{L}_r(G^*)$ – максимум живучести подсистем ранга $r < N$.*

***Гипотеза* 2**. *Структура с минимальным (средним) диаметром относится к G^*, т. е. обладает максимальной структурной живучестью.*

Справедливость гипотез, высказанных автором в 1970-х годах, подтверждена результатами статистического моделирования структур ВС.

4.7. Оптимальные структуры ВС

Основываясь на гипотезах, *оптимальными* будем считать структуры G^*, имеющие при заданных порядке N и степени ν вершин минимальный диаметр. Существуют алгоритмы синтеза оптимальных структур ВС для конкретных классов графов. Для практических целей созданы и пополняются каталоги [22] оптимальных структур ВС (D_n- и $Л(N, v, g)$-графов).

Пользование каталогами так же просто, как таблицами элементарных функций. Фрагмент каталога оптимальных D_n-графов отражён в табл. 2.

Таблица 2

D_n-граф	Характеристика					
	N	ω_1	ω_2	ω_3	ω_4	ω_5
D_2-граф	16	1	6			
	32	1	7			
	64	1	14			
	128	1	15			
	256	1	92			
D_3-граф	16	1	2	6		
	32	1	4	10		
	2048	37	116	202		
		48	407	615		
		349	390	686		
D_4-граф	16	1	2	3	4	
	32	1	2	8	13	
	64	1	4	10	17	
D_5-граф	16	1	2	3	4	5
	32	1	2	3	4	12
	1024	22	189	253	294	431
		30	133	230	253	485
		6	317	403	425	475

Как уже отмечалось (см. п. 4.4), гиперкубы весьма популярны при формировании сетей межмашинных связей большемасштабных ВС. Представляет интерес сравнить структурные показатели гиперкубов, оптимальных D_n- и $Л(N, v, g)$-графов. В табл. 3 отражены результаты сравнения названных структур по степени v, диаметру d (1) и среднему диаметру $\bar{d}$ (2) для одинаковых чисел $N = 2^v$ вершин графов. При этом следует заметить, что степень вершины D_n-графа (циркулянта) равна степени гиперкуба или (в случае нечётной степени) меньше на единицу.

Таблица 3

$N = 2^v$	Гиперкубы		Циркулянты			$Л(N, v, g)$-графы			
	$v = d$	$\bar{d}$	v	d	$\bar{d}$	v	g	d	$\bar{d}$
64	6	3.0	6	4	2.5	6	6	3	2.29
256	8	4.0	8	4	3.3	8	6	3	2.7
512	9	4.5	8	5	4.02	9	6	3	2.81
1024	10	5.0	10	5	4.04	10	6	4	3.01
2048	11	5.5	10	6	4.70	11	6	4	3.47
4096	12	6.0	12	6	4.68	12	6	4	3.57
8192	13	6.5	12	6	5.34	13	6	4	3.78
16384	14	7.0	14	6	5.38	14	6	4	3.83
32768	15	7.5	14	7	6.09	15	6	4	3.89
65536	16	8.0	16	7	6.12	16	6	5	4.06
131 072	17	8.5	16	8	6.73	17	6	5	4.39
262 144	18	9.0	18	8	6.75	18	6	5	4.62
1 048 576	20	10.0	20	8	7.41	20	6	5	4.85
16 777 216	24	12.0	24	10	8.76	24	6	6	5.56
268 435 456	28	14.0	28	11	10.15	28	6	6	5.94

Заметим, что диаметры D_n- и $Л(N, v, g)$-графов значительно меньше диаметров гиперкубов даже при одинаковых (или меньших на единицу) степенях их вершин. Более того, D_n- и $Л(N, v, g)$-графы обладают и меньшими средними диаметрами по сравнению с гиперкубами. Рассматриваемые показатели для $Л(N, v, g)$-графов при $g > 4$ являются самыми лучшими: так, диаметры для оптимальных $Л(N, v, g)$-графов оцениваются величиной $0.21\mathbf{log}_2 N$, в то время как в гиперкубах – $\log_2 N$. Из сказанного следует, что в ВС, структура которых является D_n- и $Л(N, v, g)$-графами, время межмашинных обменов информацией значительно меньше по сравнению с временем гиперкубических ВС.

Таким образом, D_n- и $Л(N, v, g)$-графы более перспективны для формирования сетей межмашинных (межпроцессорных) связей в ВС, чем гиперкубы.

Численный анализ показал, что большемасштабные ВС (с массовым параллелизмом) при существующей физико-технологической базе могут обладать высокой структурной живучестью. Учёт топологии межмашинных связей и надёжности линий связи не приводит к существенной разнице между реальной и потенциально возможной живучестью ВС.

5. Организация оптимального функционирования ВС и параллельное мультипрограммирование

В зависимости от сложности (трудоёмкости) решаемых задач и характера их поступления принято выделять три режима функционирования ВС [23]:

– решение сложной задачи;

– обработка наборов задач;

– обслуживание потоков задач.

Первый режим является монопрограммным, в нём все ресурсы ВС используются для решения одной сложной задачи. При разработке параллельных программ применяется методика крупноблочного распараллеливания [15, 18], обеспечивающая линейный рост ускорения ВС при увеличении количества входящих в неё ЭМ. Эффективность использования ресурсов ВС определяется и алгоритмами вложения в них параллельных программ. СуперВС 1970 – 90 годов были ориентированы именно на этот монопрограммный режим функционирования. Однородность структуры таких суперВС позволяла простыми средствами решить проблему вложения в них параллельных программ. Современные суперВС, характеризующиеся иерархической структурой и различными пропускными способностями каналов связи между ресурсами (вычислительными узлами, ЭМ, процессорами и их ядрами, оперативной памятью, кеш и т. п.), потребовали разработок новых алгоритмов вложения параллельных программ [24, 25].

Как уже отмечалось, вычислительные системы XXI века рассчитываются на производительность 1 TeraFLOPS – 1 ExaFLOPS, они большемасштабные и масштабируемые. Обеспечить эффективную загрузку таких суперВС представляется возможным только при помощи средств параллельного мультипрограммирования.

Под параллельным мультипрограммированием следует понимать теоретическую и практическую деятельность по созданию методов, алгоритмов и программ, оптимизирующих использование ресурсов ВС при одновременной реализации множества параллельных программ.

Рассмотрим мультипрограммные режимы функционирования ВС: обработку наборов задач и обслуживание потоков задач. В этих режимах ресурсы ВС распределяются между одновременно решаемыми задачами, представленными параллельными программами с различными количествами ветвей.

Режим обработки на ВС набора задач организуется тогда, когда они известны априори, и для каждой из них заданы такие параметры, как ранг (число параллельных ветвей в программе), время решения и/или штраф за задержку решения на единицу времени. Режим обслуживания потока задач является наиболее общим. В нём ВС используется для обслуживания вероятностных потоков заданий, представленных параллельными программами

со случайными параметрами. Математические методы, алгоритмы и программы, организующие работу ВС в последних двух режимах, составляют *инструментарий параллельного мультипрограммирования*. Этот инструментарий должен быть практически эффективным. Он не должен приводить к временны́м затратам ВС, существенно уменьшающим долю системного времени, приходящуюся на реализацию пользовательских параллельных программ. Эффективность [17] инструментария должна оцениваться такими же количественными характеристиками, как и параллельные алгоритмы: показателями сложности, коэффициентами ускорения и накладных расходов.

Современная суперВС (в силу своей природы) относится к сложным вероятностным системам. Следовательно, эффективный инструментарий параллельного мультипрограммирования должен, безусловно, базироваться на эвристике [15, 26] и стохастике [27 – 29]. Применяются два подхода к стохастической оптимизации функционирования большемасштабных и масштабируемых распределённых ВС. Оба подхода ориентированы на организацию работы ВС при обслуживании потоков параллельных задач (заданий) со случайными параметрами. Первый подход – теоретико-игровой [21, 27, 28] – применим к потокам с очередями заданий, а второй основан на методах стохастического программирования [29].

5.1. Теоретико-игровой подход к организации функционирования ВС

При рассмотрении игрового подхода к оптимизации работы ВС снимем ограничение об “абсолютной” надёжности элементарных машин, что имело место в работе [27]. Итак, пусть имеется вычислительный центр (ВЦ), который эксплуатирует ВС, состоящую из N не абсолютно надёжных ЭМ. Реализация параллельных программ осуществляется на основной подсистеме из n ЭМ; $N-n$ машин составляют структурную избыточность. В случае отказа любой машины основной подсистемы в ВС реализуется процедура восстановления, которая заключается в поиске в пределах основной подсистемы отказавшей ЭМ и реконфигурации структуры ВС. Поиск отказавшей ЭМ осуществляется средствами (само)диагностики ВС. Реконфигурация системы заключается в программной настройке новой n-машинной конфигурации из всех исправных ЭМ существовавшей основной подсистемы и части ЭМ структурной избыточности. Последняя процедура осуществляется в ВС реконфигуратором.

Средства (само)диагностики и реконфигуратор ВС являются компонентами операционной системы (ОС). Эта композиция, по сути, является виртуальным восстанавливающим устройством (ВУ); здесь, следуя традиции, её будем называть просто ВУ. В соответствии с принципом децентрализованного управления можно считать, что в распределённой ВС количество ВУ равно количеству ЭМ, входящих в её состав. Тогда для распределения $\{p_i\}$, $i \in E_0^N$, вероятностей состояний системы в стационарном режиме работы справедлива [16] формула

$$p_i = \frac{N!}{(N-i)!\,i!}\,\frac{\mu^i \lambda^{N-i}}{(\lambda+\mu)^N}\,s \qquad (6)$$

Далее, допустим, что на ВС поступает случайный поток параллельных задач (заданий) такой интенсивности, что имеет место их конечная очередь $\Im$. Пусть ожидающая решения задача $I_j^r \in \Im$ представлена парой чисел $\left\langle t_j^r, r \right\rangle$, где t_j^r – время решения задачи, r – её ранг.

Пусть $\Im^r$ – подмножество задач I_j^r ранга r, $\bigcup\limits_j I_j^r = \Im^r$, $r \in E^*$. Очевидно, что $\Im^r \subset \Im$, $\bigcup\limits_r \Im^r = \Im$, $\Im^r \cap \Im^k = \varnothing$ для $r \neq k$, $r, k \in E^*$. Считается, что число элементов в подмножестве $\Im^r \subset \Im$ достаточно большое и допускает разбиение $\Im^r$ на такие группы G^r,

для которых выполняется равенство $\left|\sum_j t_j^r - \Theta\right| = o(\Theta)$, где $\sum_j t_j^r$ есть суммарное время решения всех задач $I_j^r \in G^r$, а Θ – константа времени, выбранная заранее. Будем в дальнейшем считать, что Θ есть единица времени, и оперировать с каждой группой G^r как с задачей, имеющей ранг $r \in E^*$ и единичное время решения.

Допускаем также, что $\Im^0 \neq \varnothing$, $\Im^o \subset \Im$. Любая задача, принадлежащая $\Im^o \subset \Im$, требует для "своего решения" точно единицу времени и ноль машин ВС. Следует заметить, что если некоторое $\Im^r = \varnothing$, то подмножество ранга r должно быть сформировано из групп G^k, $k < r$, k, $r \in E^*$.

Далее, имеется также диспетчер (компонент ОС), который в дискретные моменты времени $t = 0, 1, 2, \ldots$ назначает параллельные задачи (точнее: группы G^r, $G^r \subset \Im^r \subset \Im$) для решения на ВС. Затраты на решение той или иной задачи G^r, $r \in E^*$, будем интерпретировать как *"платёж" диспетчера вычислительному центру*. Требуется создать такой алгоритм работы комплекса "ВЦ – диспетчер", чтобы в условиях не абсолютной надёжности ресурсов ВС затраты на решение задач в единицу времени были минимальны.

5.2. Игровые модели функционирования ВС

Имеется ВЦ, который в дискретные моменты времени t выделяет ресурсы ВС для решения задач. Пусть ресурсы ВЦ составляют n элементарных машин ВС. Известно подмножество $\mathcal{P} = \{p_i\}$, $i \in E^*$, значений вероятностей состояний ВС. Пусть также имеется диспетчер, который в дискретные моменты времени t назначает для решения на ВС задачи G^r различных рангов $r \in E^*$, но с единичным временем решения.

Следуя игровой терминологии, объекты "ВЦ" и "диспетчер" будем называть *игроками*. Фактически эти игроки являются виртуальными, они представлены соответствующими компонентами ОС. Далее, считаем, что ВЦ и диспетчер используют *чистые стратегии* i и r, i, $r \in E^*$, соответственно, если первый игрок для решения задач отводит i машин, а второй игрок назначает задачу ранга r. Если ВЦ выбирает стратегию с номером i, а диспетчер – стратегию с номером r, то диспетчер "платит" ВЦ сумму c_{ir}. Элементы c_{ir}, $i, r \in E^*$, составляют матрицу $\mathbf{C}$ платежей.

Предложенная модель функционирования ВС относится к классу антагонистических игр двух объектов с нулевой суммой, так как интересами ВЦ является получение максимального дохода от выделенных ресурсов, а диспетчера – минимизация затрат на решение задач.

Подбор элементов платёжных матриц $\mathbf{C}$ должен осуществляться с учётом конкретных условий эксплуатации ВС; в частности, эти элементы могут быть рассчитаны по следующей формуле:

$$c_{ir} = \begin{cases} rc_1 + (i-r)c_2 & \text{при} \quad i \geq r, \\ ic_2 + (r-i)c_3 & \text{при} \quad i < r, \end{cases} \tag{7}$$

где c_1 – цена эксплуатации одной ЭМ системы в единицу времени (платёж вычислительному центру за решение задачи единичного ранга); c_2 – величина штрафа, выплачиваемого диспетчером в единицу времени за одну простаивающую ЭМ; c_3 – величина штрафа, налагаемого на диспетчер в единицу времени, если номер его стратегии больше на единицу номера стратегии ВЦ, т.е. если $r = i + 1$, $i, r \in E^*$.

Практически наиболее вероятной и общей ситуацией является та, при которой в потоке имеются задачи всех рангов. Поэтому алгоритм функционирования ВС, состоящий в назначении задач одного ранга, представляется неэффективным. Следовательно, рассматриваемая игра не должна иметь решения в чистых стратегиях, т.е. матрица $\mathbf{C} = \|c_{ir}\|$ не должна иметь седловых точек [30]. Легко доказать следующее *утверждение*: матрица $\mathbf{C}$ не имеет седловых точек тогда и только тогда, когда $c_1 < \min\ \{c_2,\ c_3\}$.

Ясно, что при стратегии диспетчера $r > i,\ i,\ r \in E^*$, ресурсов ВС недостаточно для решения назначенной задачи, следовательно, имеет место простой всех i выделенных машин. Данное замечание позволяет осуществлять расчёт элементов матрицы $\mathbf{C}$ платежей по формуле

$$c_{ir} = \begin{cases} rc_1 + (i-r)c_2 & \text{при } i \ge r, \\ ic_4 & \text{при } i < r, \end{cases} \tag{8}$$

где c_4 – удельные потери диспетчера из-за невозможности решения задачи вследствие того, что её ранг больше количества выделенных ЭМ. Матрица $\mathbf{C}'$ не будет иметь седловых точек, если неравенство $c_1 < \min\{c_2, c_4\}$ выполняется.

В [27, 28] разработаны игровые модели и параллельные алгоритмы отыскания оптимальных смешанных стратегий ВЦ и диспетчера при заданных матрицах платежей.

Модель 1. Пусть заданы матрица $\mathbf{C} = \|c_{ir}\|$ платежей, $i,\ r \in E^*$, и смешанная стратегия $\boldsymbol{P} = \|p_0,\ p_1,\ ...,\ p_i,\ ...,\ p_n\|$ вычислительного центра. Пусть также $\boldsymbol{P}$ является распределением вероятностей (6) состояний ВС, тогда все исправные ЭМ системы автоматически выделяются для решения задач. Требуется найти оптимальную смешанную стратегию $\boldsymbol{\pi}^* = \|\pi_0^*,\ \pi_1^*,\ ...,\ \pi_r^*,\ ...,\ \pi_n^*\|$ диспетчера. Предложенная модель суть игра с “природой”, в которой в качестве последней выступает ВС.

Оптимальная смешанная стратегия $\boldsymbol{\pi}^*$ диспетчера есть решение следующей экстремальной задачи:

$$F(\boldsymbol{\pi}^*) = \min_{\boldsymbol{\pi}}\ F(\boldsymbol{\pi}),\ F(\boldsymbol{\pi}) = \sum_{r=0}^{n} \pi_r \sum_{i=1}^{n} p_i c_{ir}, \tag{9}$$

где целевая функция $F(\boldsymbol{\pi})$ суть расходы диспетчера при применении смешанной стратегии $\boldsymbol{\pi} = \|\pi_r\|,\ r \in E^*$.

Так как вероятности состояний ВС известны, то оптимальная смешанная стратегия $\boldsymbol{\pi}^*$ диспетчера имеет координаты $\pi_r^* = 0,\ r \in E^*,\ r \ne r^*;\ \pi_{r^*}^* = 1,$ где r^* – любая стратегия диспетчера, для которой выполняется условие

$$\sum_{i=0}^{n} p_i c_{ir^*} = \min_{r} \left\{ \sum_{i=0}^{n} p_i c_{ir} \right\}. \tag{10}$$

Таким образом, в данной ситуации наиболее эффективно решать задачи ранга r^*, определяемого из условия (10). При решении трудоёмких задач этому требованию легко удовлетворить. В самом деле, для достаточно широкого круга задач с большим объёмом вычислений [15 – 18] могут быть составлены адаптирующиеся параллельные программы, которые могут настраиваться на любое число ЭМ, в частности, равное r^*.

Модель 2. В отличие от модели 1, здесь будем считать, что если вычислительная система находится в состоянии $k \in E^*$, то ВЦ может выделить i элементарных машин для решения задач, $0 \le i \le k$.

Рассматривается следующая трёхходовая игра. Первый ход делает случайный механизм, который выбирает число $k \in E^*$ с вероятностью p_k. Вычислительный центр, зная *k*, выставляет для решения *i* машин, где $0 \le i \le k$ (2-й ход). Диспетчер, независимо от ВЦ, назначает задачу ранга *r* (3-й ход) и платит ВЦ c_{ir} денежных единиц (c_{ir} определяется только *i* и *r*, не зависит от p_k ,).

Модель 2 позволяет организовать матричную игру, в которой информации о вероятностях состояний системы "скрыта" от диспетчера. Последнее достигается путём изменения платежей

$$\tilde{c}_{ir} = c_{ir} \sum_{k=1}^{n} p_k \ .$$

Итак, требуется для матрицы платежей $\tilde{C} = \|\tilde{c}_{ir}\|$, $i,\ r \in E^*$, найти решение и цену V игры, где $\boldsymbol{\mathcal{P}}^*$ и $\boldsymbol{\pi}^*$ – оптимальные смешанные стратегии ВЦ и диспетчера, соответственно; $V = \boldsymbol{\mathcal{P}}^* \tilde{\mathbf{C}} (\boldsymbol{\pi}^*)^T$, $(\boldsymbol{\pi}^*)^T$ – транспонированный вектор $\boldsymbol{\pi}^*$; p_k определяется формулой (6); в качестве c_{ij} используются либо элементы, рассчитываемые по формуле (7), либо элементы $c_{ij}^{'}$ (8).

Для решения "теоретико-игровых" проблем организации функционирования большемасштабных распределённых ВС и суперВС (с массовым параллелизмом) эффективен параллельный алгоритм [21], основанный на композиции симплекс-метода и модифицированного метода Брауна-Робинсон [30].

5.3. Организация функционирования ВС и стохастическое программирование

Убедимся в плодотворности стохастического программирования [31] для мультипрограммирования. Пусть ВС состоит из N ЭМ, а L – число терминалов, воспринимающих поток задач. Под терминалом здесь понимается не только физическое устройство, предназначенное для загрузки программ, но и виртуальное устройство, формирующее поток задач. Такая постановка особенно актуальна, когда работа с ВС осуществляется через локальную сеть или Internet.

Каждая из задач, поступающих в систему, характеризуется рангом, т.е. числом ветвей в её параллельной программе. Считается, что на каждый терминал поступает поток задач всевозможных рангов. Спросом на подсистемы ранга j с терминала l назовем величину a_{jl}, выражающую количество подсистем, требующихся за некоторый промежуток времени T терминалу l для решения задач ранга j, $j \in \{1, 2, ..., N\}$, $l \in \{1, 2, ..., L\}$. Другими словами, a_{jl} – среднее число подсистем ранга j, которые могут быть загружены с терминала l за время T. Таким образом, для каждого терминала спрос на подсистемы различных рангов заранее неизвестен. Допустим, известно, что a_{jl} – случайная величина с плотностью распределения вероятностей $p_{jl}(a)$; очевидно, что

$$\int_0^\infty p_{jl}(a)da = 1, \quad j \in \{1, 2, ..., N\}, \quad l \in \{1, 2, ..., L\}.$$

Полагаем, что разбиение ВС на подсистемы происходит в фиксированные моменты времени, причём T – интервал между разбиениями. Особенность такого подхода заключается в том, что для каждого промежутка времени T расчёт разбиения производится только один раз, и считается, что в течение этого промежутка состав подсистем не меняется.

Пусть d_{jl} – цена эксплуатации подсистемы ранга j, а c_{jl} – стоимость её формирования и обслуживания для терминала l; y_{jl} и x_{jl} – количества подсистем ранга j, выделяемых терминалу l соответственно в обязательном порядке и дополнительно.

Ожидаемая прибыль от эксплуатации подсистем ранга j с терминала l определяется величиной

$$r_{jl}(x_{jl}) = (d_{jl} - c_{jl})(x_{jl} + y_{jl}) - d_{jl} \int_0^{x_{jl}+y_{jl}} (x_{jl} + y_{jl} - a)\, p_{jl}(a)\, da,$$

причём $x_{jl} = 0$ при $j = \overline{n+1, N}$.

Задача стохастической оптимизации функционирования ВС имеет вид:

$$\sum_{j=1}^{n} \sum_{l=1}^{L} r_{jl}(x_{jl}) \to \max_{\{x_{jl}\}}, \quad j = \overline{1, n}, \quad l = \overline{1, L};$$

$$\sum_{j=1}^{n} \sum_{l=1}^{L} j x_{jl} \le n, \quad n = N - \sum_{j=1}^{N} \sum_{l=1}^{L} j y_{jl}.$$

Для решения поставленной задачи эффективен параллельный алгоритм, базирующийся на технике динамического программирования [29].

Таким образом, описанный стохастический инструментарий параллельного мультипрограммирования не трудоёмок, он оптимизирует загрузку ресурсов ВС множеством задач, представленных параллельными программами с произвольными количествами ветвей, и может быть положен в основу распределённых операционных систем.

6. Практика вычислительных систем с программируемой структурой

Интерес к практической реализации ВС с программируемой структурой постоянно проявлялся начиная с 60-х годов XX века. Первоначально он поддерживался прежде всего необходимостью проверки теоретических основ построения ВС, необходимостью отработки архитектурных решений и функциональной структуры ВС, а также параллельных вычислительных технологий. Позднее возрастающую роль стал играть утилитарный компонент целей создания ВС, в 1970-х годах этот компонент стал превалировать над исследовательским. Последнее обосновывается потребностью в ВС, обладающих и высокой производительностью, и надёжностью, и живучестью.

Работы по построению ВС, основанных на принципах модели коллектива вычислителей, были инициированы в ИМ СО АН СССР в 1964 г.; вскоре в институте было организовано и мини-производство ВС [32].

6.1. Вычислительная система “Минск-222”

Система “Минск-222” [15, 17] – первая в мире ВС с программируемой структурой. В проекте “Минск-222” были отработаны архитектурные, технические и программные решения, значительная часть из которых была “канонизирована” разработчиками не-фон-неймановских вычислительных средств.

Система “Минск-222” была разработана и построена Отделением вычислительной техники ИМ СО АН СССР совместно с Конструкторским бюро завода им. Г.К. Орджоникидзе Министерства радиопромышленности СССР (г. Минск). Руководитель работ по созданию ВС “Минск-222” – Э.В. Евреинов; основные разработчики: В.Г. Хорошевский, Б.А. Сидристый, Г.П. Лопато (1924 – 2001; чл.-корр. РАН с 1979 г.), А.Н. Василевский. Работы по проектированию ВС “Минск-222” были начаты в 1965 г., а первый её образец был установлен в апреле 1966 г. в Институте математики АН БССР. Системы “Минск-222” были смонти-

рованы в нескольких организациях Советского Союза (в частности, на плавбазе “Феликс Кон”) и эксплуатировались более 15 лет.

Архитектура ВС:

- MIMD, распределённость ресурсов;
- параллелизм, однородность, программируемость структуры;
- одномерная (кольцевая) топология;
- масштабируемость: 1–16 элементарных машин;
- быстродействие: $\Omega = AN\omega$, где N – число ЭМ, ω – быстродействие одной ЭМ, $A \geq 1$ (при крупноблочном распараллеливании сложных задач);
- использование промышленных ЭВМ второго поколения.

6.1.1. Элементарная машина ВС

В системе “Минск-222” каждая ЭМ состояла из вычислительного модуля (ВМ) и системного устройства (СУ). В качестве ВМ были использованы конфигурации серийных ЭВМ “Минск-2” или “Минск-22”, выпускавшиеся заводом им. Г.К. Орджоникидзе (г. Минск). Указанные ЭВМ имели одну и ту же двухадресную архитектуру, “Минск-22” в сравнении с “Минск-2” обладала магнитной памятью удвоенной ёмкости (8К 37-разрядных слов).

Обращаем внимание читателей на то, что подход к построению параллельных ВС, ориентированный на применение серийных ЭВМ, был впервые применён в Сибирском отделении АН СССР [15], а не за рубежом (см., например, разработки 1970-х годов Университета Карнеги–Меллона [17], а также современные кластерные ВС).

В состав *системного устройства* (рис.4) входили локальный коммутатор (ЛК) каналов связи и блок операций системы (БОС). Коммутатор ЛК$_i$ состоял из вентилей, которые открывали или закрывали канал связи, идущий к соседней справа ЭМ, т. е. к коммутатору ЛК$_j$, где $j = (i+1) \bmod N$. Вентили управлялись сигналами, поступавшими из БОС.

Блок операций системы включал в себя регистр настройки (РН) и узел, реализующий системные команды. Содержимое РН определяло вид соединительной функции коммутатора и степень участия ЭМ при системных взаимодействиях. Регистр настройки состоял из трёх разрядов: $TR, TQ, T\Omega$.

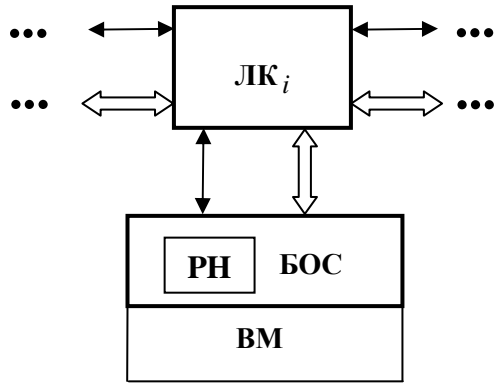


Рис. 4. ЭМ системы “Минск-222”:
⇨ – рабочий канал, → – управляющий канал

Триггер TR позволял разбивать систему на функционально изолированные подсистемы. Триггеры TQ и $T\Omega$ конкретизировали степень участия машин в выполнении некоторых системных команд. В частности, триггеры $T\Omega$ использовали выработки обобщённых признаков

$$\Omega_k = \mathop{\&}_{i \in E} \omega_{ki}, \quad k = 1, 2, 3,$$

где E – подмножество номеров машин, управлявших ходом вычислений (т. е. отмеченных единицей в разряде $T\Omega$), $E \subset \{0, 1, ..., N-1\}$; ω_{ki} – признаки, вырабатываемые ЭМ с номером i.

Системное устройство было реализовано на 80 стандартных элементах и составляло менее 1.5 % объёма оборудования устройств управления и арифметико-логического ЭВМ “Минск-22”.

6.1.2. Системные команды ВС

К системным относятся команды, обеспечивающие организацию и реализацию параллельных вычислительных процессов, в частности, обменов управляющей информацией и данными между ветвями параллельной программы. Набор системных команд ВС “Минск-222” составляли *команды настройки, обмена, обобщённых безусловного и условного переходов* [15, 17]. *Команды настройки* позволяли программировать структуру ВС и задавать степень участия каждой ЭМ в реализации параллельных процессов.

Команды обмена **–** это команды передачи (П) и приёма (ПР). По команде П передающая ЭМ выдавала в (предварительно настроенный) канал межмашинной связи определённое количество слов из своей памяти. При выполнении команды ПР машиной осуществлялся приём из канала заданного количества слов. Такой способ организации обменов информацией между машинами ВС подобен функционированию радио- и телепередающих систем. Он не зависит от числа ЭМ в системе и позволяет избежать трудностей с адресацией машин. Описанные команды межмашинного обмена информацией не требуют сложной аппаратурной поддержки, они адекватны большемасштабным и масштабируемым ВС.

Команды обобщённого безусловного перехода (ОБП) предназначались для принудительного управления работой ВС. Управляющая машина при выполнении ОБП засылала в канал содержимое заданной ячейки своей памяти, которое воспринималось как команда только ЭМ, содержащими единицу в разрядах TQ–регистров настройки. Команды ОБП позволяли осуществлять инициирование работы ВС и её загрузку данными из любой ЭМ, а также вмешиваться в параллельные вычислительные процессы и принудительно управлять работой подмножеств машин системы из любой ЭМ. В частности, при помощи команды ОБП из любой ЭМ можно было и инициировать программирование структуры ВС.

Команды обобщённого условного перехода (ОУП) и содержимое $T\Omega$–регистров настройки элементарных машин ВС позволяли управлять параллельными вычислительными процессами по значениям обобщённых признаков Ω_k. Одна из команд ОУП служила для синхронизации параллельных процессов (машин, выполняющих ветви параллельной программы). По сути, эта команда позволяла реализовать *механизм синхронизации*, получивший позднее название “*Barrier*”. Команды ОУП использовались, в частности, для организации ветвлений в параллельных вычислениях, для реализации параллельных циклов.

6.1.3. Программное обеспечение “Минск-222”

В системе “Минск-22” программное обеспечение (ПО) было ориентировано лишь на реализацию в монопрограммном режиме параллельных программ (*P*-программ). Оно состояло из двух частей: *системы P-программирования* и *пакета прикладных адаптирующихся P-программ*. Система параллельного *P*-программирования включала средства автоматизации *P*-программирования, отладки, редактирования и анализа *P*-программ.

Средства автоматизации P-программирования – языки и трансляторы. В качестве входных в системе “Минск-222” использовались расширенные языки [17]: автокод АКИ, ЛЯПАС, ALGOL, BASIC. В расширенные языки были включены средства для описания взаимодействий между параллельными ветвями вычислений. Каждый транслятор для ВС состоял из двух частей: обычного транслятора для ЭВМ и системного блока для реализации межмашинных взаимодействий в ВС. Любой системный блок представлял собой совокупность программ для реализа-

ции операций настройки, обмена, ОБП и ОУП, которые были включены в библиотеки трансляторов. Системные блоки имели относительно небольшое количество команд. Так, отношение объёма системного блока к общему для трансляторов с АКИ и ЛЯПАС не превышало 0.1, а для транслятора с ALGOL (ТАМ-2/22) – 0.025.

Средства отладки и редактирования Р-программ – совокупность четырёх программ. Первая программа преобразовывала отлаживаемую *Р*-программу в последовательную и выявляла ошибки, не связанные с использованием системных команд.

Вторая программа служила для моделирования на одной машине выполнения *Р*-программы из двух ветвей. Всевозможные (допустимые и недопустимые) взаимодействия ветвей были представлены матрицей переходов к моделирующим или авостным подпрограммам. Эта же программа могла определять время простоев машин, время работы отдельных блоков *Р*-программы и точность вычислений.

Третья программа позволяла вывести на печать заданное количество раз содержимое интересующих областей памяти перед выполнением команд обмена в процессе контрольной реализации параллельной программы на ВС “Минск-222”.

Четвёртая программа служила для корректировки *Р*-программ.

Средства анализа Р-программ были представлены тремя программами.

Первая программа служила для анализа распределения памяти между блоками исследуемой программы.

Вторая программа предназначалась для измерения времени простоев машин ВС.

Третья программа применялась для измерения времени работы участков *Р*-программы.

Пакеты прикладных адаптирующихся Р-программ были ориентированы на решение задач повышенной сложности. Параметры таких задач, как правило, не позволяли решать их на ЭВМ “Минск-22” за удовлетворительное время.

Из опыта создания ПО для системы “Минск-222” установлено, что его объём отличается от объёма программного обеспечения ЭВМ “Минск-22” не более чем на 10 % .

6.1.4. Опыт эксплуатации ВС “Минск-222”

Полный набор схем обмена информацией между ветвями *Р*-программ, как показала эксплуатация ВС “Минск-222”, составляют: дифференцированный (ДО), трансляционный (ТО), трансляционно-циклический (ТЦО), конвейерно-параллельный (КПО) и коллекторный обмены (КО).

При *трансляционном обмене* (современный термин *“One-to-all Broadcast”*) осуществляется передача одной и той же информации из одной (любой) ветви одновременно во все остальные ветви *Р*-программы.

Трансляционно-циклический обмен (“All-to-all Broadcast”) реализует трансляцию информации из каждой ветви во все остальные. Следовательно, если трансляционный обмен выполняется за 1 такт, то трансляционно-циклический – за N тактов.

Конвейерно-параллельный обмен обеспечивает передачу информации между соседними ветвями; он выполняется за два такта. Так, например, при чётном N в первом такте осуществляется передача информации из ветвей $P_1, P_3, ..., P_{i-1}, ..., P_{N-3}, P_{N-1}$ соответственно в ветви $P_2, P_4, ..., P_i, ..., P_{N-2}, P_N$; во втором такте информация из последней последовательности ветвей поступает соответственно в ветви $P_3, P_5, ..., P_{i+1}, ..., P_{N-1}, P_N$.

Коллекторный обмен представляет собой инвертированный трансляционный обмен, в одну ветвь последовательно собирается информация из $l < N$ ветвей. Такой обмен требует l тактов и реализуется как последовательность из l дифференцированных обменов.

Статистическая информация, отражающая частоту использования схем обмена при реализации крупноблочных параллельных алгоритмов или программ, представлена ниже:

Схема обмена	ДО	ТО	ТЦО	КПО	КО
Частота использования, %	2	17	40	34	7

Таким образом, коллективные схемы обмена информацией между ветвями параллельных алгоритмов или программ (именно ТО, ТЦО, КПО) составляют более 90 % от общего количества обменов. Это позволяет создавать высокоэффективные параллельные алгоритмы и обеспечивает одновременную работу всех ЭМ системы.

Было установлено, что при решении задач на ВС “Минск-222” системные команды в *P*-программах составляли, как правило, менее 10 % их общего объёма. Следовательно, можно считать, что затраты при разработке параллельных и эквивалентных им последовательных программ имеют один и тот же порядок. Выявлено также, что для ВС “Минск-222” доля затрат времени на системные взаимодействия (включая синхронизацию) составляет, как правило, несколько процентов, что является следствием применения методики крупноблочного распараллеливания задач. Кроме того, выяснилось, что за счёт большей ёмкости оперативной памяти в системе “Минск-222” по сравнению с одной ЭВМ “Минск-22” и за счёт быстродействия каналов связи, сравнимого с быстродействием ЭВМ при выполнении операций, получается дополнительный значительный выигрыш во времени решения задач на ВС.

Следует обратить особое внимание на “*парадокс параллелизма*” – нелинейный рост производительности ВС при повышении количества *N* ЭМ (что противоречит якобы здравому смыслу). “*Парадокс параллелизма” был впервые обнаружен при работе именно на системе “Минск-222*” [17]. Реакцию на обнаруженный факт легко восстановить, если учесть, что в 1960-х годах существовало устойчивое мнение о невозможности создания параллельных средств обработки информации с большим количеством арифметико-логических устройств (АЛУ). Считалось и “доказывалось”, что увеличение быстродействия ВС происходит только при наращивании количества АЛУ до 10, после чего наблюдается замедление, обусловленное накладными расходами на организацию совместной работы нескольких АЛУ. Да, всё это было правильно, только неверными были концептуальные подходы к организации параллельных вычислительных систем.

Экспертные оценки показали, что сложность программирования для ВС “Минск-222” (по сравнению со сложностью программирования для одной ЭВМ “Минск-22”) возрастает на 10…20 %, а при развитой библиотеке стандартных параллельных программ – на 5…10 %. Вычислительные системы “Минск-222” в течение многолетней эксплуатации в различных организациях СССР показали высокую эффективность при решении широкого круга задач.

Следует особо подчеркнуть, что архитектурные решения, реализованные в ВС “Минск-222”, стали, по сути, каноническими. Схемы обмена информацией между ветвями *P*-программ и рассмотренные в данном параграфе системные команды нашли отражение в современном инструментарии, используемом при построении распределённых и параллельных ВС. Так, в MPI (Message Passing Interface) – библиотеке функций, предназначенной для поддержки параллельных процессов – применяются как дифференцированный (Point-to-point Communication), так и коллективные взаимодействия (Collective Communications). В табл. 4 приведены основные виды системных взаимодействий и реализующие их команды ВС “Минск-222” и функции MPI.

Полученный опыт по проектированию, математической и технической эксплуатации “Минск-222” был использован в последующих проектах вычислительных систем с программируемой структурой.

6.2. Вычислительные системы из мини-ЭВМ

Вычислительные системы, которые формировались из аппаратурно-программных средств мини-ЭВМ, относились к группе мини-ВС. Построение таких ВС было одной из основных мировых тенденций развития вычислительной техники 1970-х годов. Опыт эксплуатации показал, что при решении большого круга задач мини-ВС были более

эффективны с точки зрения производительности, надёжности, живучести и стоимости, чем одна или даже несколько больших ЭВМ третьего поколения.

Таблица 4

Вид взаимодействия	Команды ВС “Минск-222”	Функции MPI
Дифференцированный обмен	**П, ПР**	MPI_Send, MPI_Recv
Трансляционный обмен	**П, ПР**	MPI_Bcast
Трансляционно-циклический обмен	**П, ПР,** N итераций	MPI_Alltoall или MPI_Allscatter
Коллекторный обмен	**П, ПР**, N – 1 итерация	MPI_Gather
Синхронизация элементарных машин	**ОУП**	MPI_Barrier
Разбиение ВС на подсистемы	**Н**	MPI_Comm_group MPI_Group_incl MPI_Comm_create MPI_Cart_create

Работы по созданию ВС из мини-машин достаточно интенсивно велись в США, однако общей концепции построения таких систем американские специалисты не выработали. Анализ проектов показывает, что использовались в основном три способа организации ВС:

1. системы с общей памятью;
2. ВС с общей шиной (или системой шин), к которой подключались процессоры, запоминающие и другие устройства;
3. системы, в которых машины взаимодействовали через общую группу устройств ввода-вывода информации. Как правило, системы не имели программируемой структуры и обладали ограниченными возможностями к наращиванию.

При создании мини-ВС в Советском Союзе за основу была взята концепция ВС с программируемой структурой. Архитектурные решения в области мини-ВС, опыт их проектирования, разработки системного и прикладного ПО нашли массовое применение только в конце XX века. Именно вычислительные кластеры являются, по существу, многопроцессорными или многомашинными ВС, конфигурируемыми из микропроцессоров или персональных ЭВМ (например, IBM PC). При этом следует отметить заметную архитектурную близость мини-ЭВМ и современных персональных компьютеров.

6.2.1. Вычислительная система МИНИМАКС

МИНИМАКС (МИНИМАшинная программно Коммутируемая Система) создана ИМ СО АН СССР (Отделом вычислительных систем) и Научно-производственным объединением “Импульс” Министерства приборостроения, средств автоматизации и систем управления СССР (г. Северодонецк). Технический проект МИНИМАКС разработан в 1974 г., а опытно-промышленный образец системы был изготовлен и отработан в 1975 г.

Архитектура системы:

- MIMD;
- распределённость средств управления, обработки и памяти;
- параллелизм, однородность, модульность;
- программируемость структуры;
- двумерная (циркулянтная) топология;
- масштабируемость;

• живучесть;

• максимальное использование промышленных средств мини-ЭВМ.

Функциональная структура мини-ВС МИНИМАКС – композиция из произвольного количества элементарных машин и программно настраиваемой сети связей между ними. Взаимодействия между ЭМ в системе МИНИМАКС осуществлялись через сеть связей (рис. 5), которая формировалась из одномерных *1* и двумерных *2* полудуплексных каналов. Одномерные каналы связи *1* были управляющими; они служили для программирования соединений между ЭМ по каналам связи *2*, а также для передачи между ЭМ управляющей информации, регламентирующей использование общих ресурсов (внешних устройств, сервисных программ, файлов и т. п.). Двумерные каналы связи *2* являлись рабочими; они применялись для следующих целей: реализации основных межмашинных взаимодействий, пересылки массивов данных между памятями передающей ЭМ и одной или нескольких принимающих ЭМ, передачи адресов из одной ЭМ в другую и обмена логическими переменными между машинами.

Межмашинные взаимодействия при функционировании мини-ВС реализовывались с помощью специальных подпрограмм – *системных драйверов*, которые, в свою очередь, использовали специальные команды (занесение кода на регистр настройки, считывание его содержимого, занесение информации в СУ о начальном адресе передаваемого массива данных и т. п.).

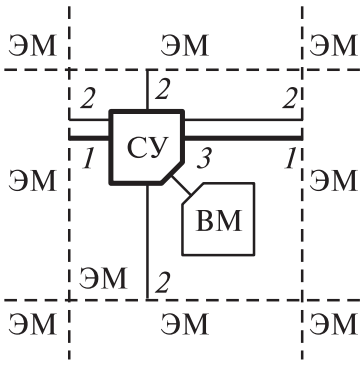


Рис. 5. ЭМ мини-ВС МИНИМАКС

Очевидно, что допускалось формирование конфигураций системы МИНИМАКС с тороидальной структурой; для специальных областей применения использовались оптимальные D_2-графы (рис.6).

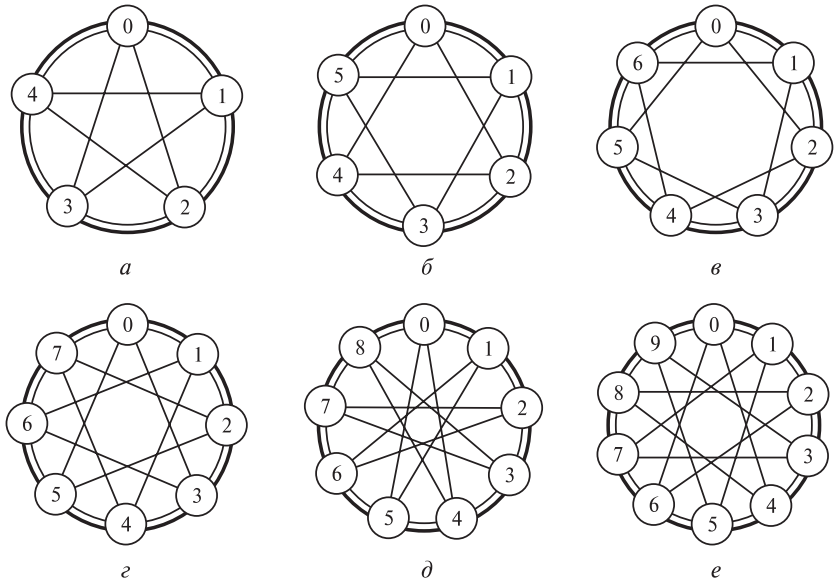


Рис. 6. Оптимальные структуры мини-ВС МИНИМАКС, D_2-графы вида:
а – {5; 1.2}; *б* – {6; 1.2}; *в* – {7; 1.2}; *г* – {8; 1.3}; *д* – {9; 1.4}; *е* – {10; 1.4}

В пределах мини-ВС МИНИМАКС допускалось формирование произвольного числа подсистем из любого количества ЭМ. Подсистему в мини-ВС составляли взаимодействовавшие друг с другом ЭМ вместе с машинами, которые использовались в качестве транзитных пунктов передачи информации. Каждая ЭМ мини-ВС могла входить только в одну из подсистем (ПС2), образованных по связям *2*. Вместе с тем, она могла входить и в одну из подсистем (ПС1), образованных по связям *1*. Машина, принадлежавшая подсистемам ПС1 и ПС2, не могла одновременно участвовать в нескольких взаимодействиях. Подсистемы ПС2 могли сохраняться в течение нескольких следующих друг за другом взаимодействий. Подсистемы ПС1 образовывались только на время одного взаимодействия и разрушались после его выполнения.

Элементарная машина МИНИМАКС – это композиция из ВМ и СУ (рис.5). Структура ЭМ данной мини-ВС не была жёстко заданной и определялась областью применения. Состав каждой ЭМ допускал варьирование; компоновка ЭМ проводилась по правилам, которые были приняты для агрегатных средств ВТ на микроэлектронной основе (АСВТ-М) или для средств системы малых ЭВМ (СМ ЭВМ). В качестве ВМ могли быть использованы любые конфигурации мини-ЭВМ на базе процессоров М-6000, М-7000, СМ-1П. Архитектура системы МИНИМАКС была рассчитана также на применение мини-ЭВМ моделей HP 2114–2116 семейства Hewlett-Packard.

Системное устройство было спроектировано как автономное устройство АСВТ-М. Оно подключалось к ВМ через связи *3* (рис. 5). При выборе способа реализации связей *3* учитывались принципы построения АСВТ-М и следующие отсюда ограничения:

- целесообразность построения системного устройства в виде отдельного модуля;
- недопустимость изменений в схемах и конструкции процессоров АСВТ-М.

Программное обеспечение мини-ВС МИНИМАКС состояло из управляющей системы, средств Р-программирования, пакетов Р-программ и комплекса программ технического обслуживания.

6.2.2. Вычислительная система СУММА

В 1970-х годах для управления процессами в реальном времени широко применялись не только мини-машины, но и вычислительные сети и системы из мини-ЭВМ. В данном разделе описывается вторая отечественная мини-ВС: Система Управляющая Мини–МАшинная (СУММА).

Система СУММА[*)] была разработана ИМ СО АН СССР (Отделом вычислительных систем) совместно с Производственным объединением “Кварц” Министерства электронной промышленности СССР (г. Калининград). Техническое проектирование мини-ВС было выполнено в 1975 г., опытно-промышленный образец был изготовлен и отработан в 1976 г.

Данная Мини-ВС, как и система МИНИМАКС, имела программируемую структуру и свои архитектурные особенности:

- $Л(N,3,g)$ - структуру;
- единый канал для управляющей и рабочей информации;
- аппаратурно-программную реализацию системных взаимодействий.

Функциональная структура мини-ВС СУММА характеризовалась большой гибкостью. Её можно было легко расширить или сократить в соответствии с предъявляемыми требованиями. Принципиальные ограничения на структуру мини-ВС (количество ЭМ и порядок их соединения) не накладывались, однако при любой структуре каждая ЭМ могла взаимодействовать не более чем с тремя соседними машинами с помощью полудуплексных каналов

[*)] Непосредственными участниками проекта СУММА были аспиранты автора данной статьи: М.П. Желтов, генеральный директор ПО “Кварц”, А.П. Ерёмин, главный инженер, В.П. Афанасьев, начальник лаборатории. В ПО “Кварц” планировалась реализация ВС “Электроника СС БИС-1”, разрабатываемой под руководством В.А. Мельникова (1928–93; академик АН СССР с 1981 г.). Именно в ПО “Кварц” произошло слияние интересов двух научных школ по достижению их главной цели – создать в СССР индустрию суперкомпьютеров.

(рис. 7). В мини-ВС была заложена возможность “программировать” адресацию ЭМ, в частности, система могла быть настроена на относительную адресацию ЭМ.

Системы управления, на применение в которых была рассчитана мини-ВС СУММА, характеризуются стабильностью решаемых задач, нежёсткими требованиями к реактивности на изменение операционной обстановки (преимущественно детерминированный поток запросов на обслуживание). Следовательно, в системах управления перепрограммирование структуры мини-ВС требовалось выполнять редко и время обмена управляющей информацией в общем времени работы машин системы составляло незначительную часть. Эти факторы позволили ограничиться единым каналом для обмена управляющей (настроечной) информацией и данными между ЭМ мини-ВС.

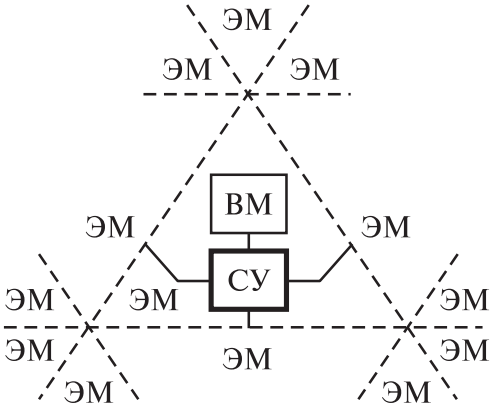


Рис. 7. ЭМ мини-ВС СУММА

Единый канал обмена управляющей и рабочей информацией между машинами системы СУММА вместе с программной реализацией некоторых функций позволили по сравнению с системой МИНИМАКС существенно упростить СУ (например, программными средствами в системе СУММА реализовывалась выработка обобщённого признака Ω).

Из-за использования для всех взаимодействий одних и тех же связей перепрограммирование структуры мини-ВС можно было осуществлять только в границах сформированных подсистем. Снятие границ образованной подсистемы производилось только “изнутри”.

К системам управления предъявляются повышенные требования по живучести, следовательно, их вычислительные средства должны обладать структурной живучестью. Для формирования мини-ВС СУММА использовались оптимальные $L(N, 3, g)$-графы (рис. 8).

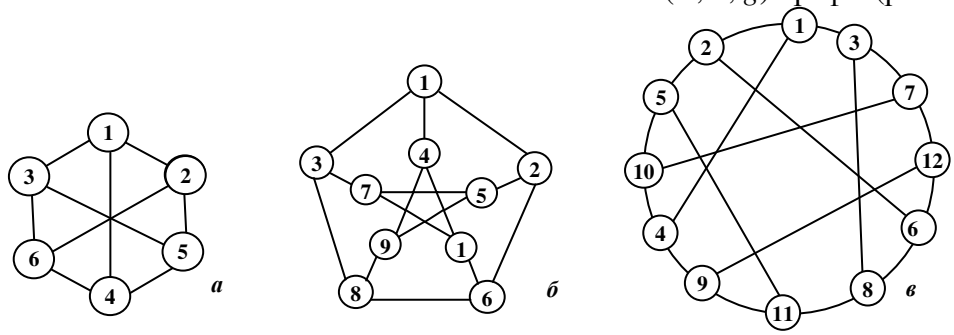


Рис. 8. Оптимальные структуры мини-ВС СУММА:
а – L(6, 3, 4); б – L(10, 3, 5); *в* – L(12, 3, 5)

Элементарная машина системы СУММА формировалась как “трёхполюсник”, или, точнее, композиция из ВМ и СУ, рассчитанного на три межмашинные связи (рис. 7).

Вычислительный модуль предназначался для выполнения всех операций, связанных с переработкой информации, в частности, для инициирования реализации системных операций. Системное устройство использовалось для реализации системных взаимодействий машин, в частности, для программирования структуры мини-ВС. В качестве ВМ использовали

произвольные конфигурации мини-ЭВМ “Электроника-100 И”. Следует заметить, что архитектура системы СУММА была ориентирована также на применение мини-ЭВМ PDP-8 фирмы Digital Equipment Corp.

Программное обеспечение мини-ВС СУММА – проблемно-ориентированное. В его состав входили супервизор (являвшийся программой управления процессами в реальном масштабе времени), система *Р*-программирования (включающая макроассемблер MACRO-8P), системы для автоматизированного управления технологическими процессами и комплекс программ технического обслуживания.

6.3. Вычислительные системы семейства МИКРОС

Прогресс в индустрии обработки информации неразрывно связан с достижениями в области элементной базы и в интегральной технологии. В конце 1970-х годов мини-процессоры вытесняются микропроцессорами, на смену мини-ЭВМ пришли микроЭВМ; создаются параллельные ВС как коллективы микропроцессоров.

В начале 1980-х годов в Отделе вычислительных систем СО АН СССР инициируются работы по научно-исследовательскому проекту МИКРОС [33], целью которых было создание МИКРОпроцессорных Систем с программируемой структурой (МИКРОС). Результатом работ явилось семейство МИКРОС, включающее модели МИКРОС-1 (1986); МИКРОС-2 (1992); МИКРОС-Т (1996). Разработка моделей семейства МИКРОС осуществлялась в содружестве с подразделениями Научно-производственного объединения “Алмаз” и Научно-исследовательского института “Квант” Министерства радиопромышленности СССР (г. Москва).

6.3.1. Функциональная структура МИКРОС

Возможности систем семейства МИКРОС определяются количеством ЭМ, входящих в их состав, конфигурациями ЭМ и топологией сетей межмашинных связей. Количество ЭМ в любой из моделей (МИКРОС-1, МИКРОС-2, МИКРОС-Т) не фиксировано. Каждая ЭМ – это многополюсник, число полюсов v в первых моделях систем составляло от 2 до 8, а в модели МИКРОС-Т $v = 4$.

Каждая генерация ВС семейства МИКРОС адекватно учитывала текущие возможности вычислительной техники и интегральной технологии. Для формирования конфигураций ЭМ моделей МИКРОС-1 и МИКРОС-2 использовались средства микроЭВМ отечественного семейства “Электроника”; элементарная машина (рис.9) представлялась композицией из модулей микроЭВМ и модулей системного устройства (МСУ).

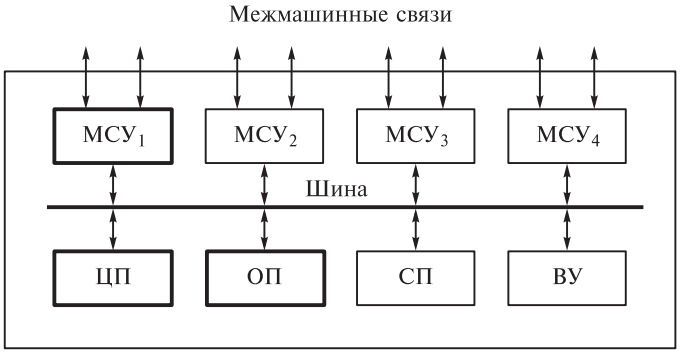


Рис. 9. ЭМ систем МИКРОС-1 и МИКРОС-2

Свойством масштабируемости обладали не только модели семейства МИКРОС, но и их ЭМ. Простейшая конфигурация ЭМ состоит из одного МСУ, центрального процессора (ЦП) и оперативной памяти (ОП). Модуль СУ обеспечивал реализацию системных операций в ВС и непосредственную связь данной ЭМ с двумя соседними машинами через полудуплексные

каналы. Модуль СУ позволял использовать в качестве каналов различные средства, в частности, экранированные провода (при расстоянии между ЭМ до 30 м), либо радиочастотные кабели (если расстояние между ЭМ не превышало 300 м), либо коммутируемые или выделенные телефонные каналы связи (с использованием аппаратуры передачи данных независимо от расстояния между ЭМ). Заложенная в модуль СУ схема обеспечения связности машин была равно пригодна для формирования пространственно сосредоточенных и распределённых ВС.

В моделях ВС МИКРОС-1 и МИКРОС-2 в качестве базовых машин были использованы микроЭВМ “Электроника 60М” и “Электроника 60-1” соответственно. Расширенные конфигурации ЭМ (см. рис. 9) систем МИКРОС-1 и МИКРОС-2 могли иметь до четырёх модулей СУ, специальный процессор (СП), дополнительные модули оперативной памяти, набор внешних устройств (ВУ). Специальные процессоры “Электроника МТ-70”, или “Электроника 1603” расширяли вычислительные возможности ЦП при решении научно-технических задач, связанных с обработкой значительных массивов данных и с выполнением больших объёмов однородных вычислений.

Модули системного устройства для системы МИКРОС-2 обладали бóльшими функциональными возможностями, чем в системе МИКРОС-1. Их аппаратура, в частности, позволяла осуществлять:

- обработку входных/выходных запросов для межмашинных связей;
- анализ семафоров; формирование пакетов выходных сообщений;
- управление входными и выходными портами при выполнении системных команд;
- мультиадресные передачи информации;
- совмещение межмашинных обменов информацией с вычислениями.

Система МИКРОС-Т базируется на транспьютерных технологиях [34]. Такие технологии позволяют формировать двумерные ВС с массовым параллелизмом. Двумерные структуры ВС формируются путём отождествления полюсов-линков (Link – связь).

Простейшая конфигурация ЭМ представляется транспьютером (например, Inmos T805) с памятью, развитые конфигурации ЭМ могли включать в себя: коммуникационные транспьютеры и высокопроизводительные микропроцессоры – Intel 860 (компания Intel), PowerPC (альянс компаний IBM, Apple и Motorola), Alpha (компании DEC и Compaq) и др. Для формирования ЭМ системы МИКРОС-Т могли быть использованы стандартные решения зарубежных и отечественных фирм-производителей транспьютерных модулей.

6.3.2. Программное обеспечение МИКРОС

Эффективная работа ВС и её пользователей немыслима без ОС и среды параллельного программирования. Любая система семейства МИКРОС, как и её ПО, были открыты к совершенствованию. Ряд моделей семейства ВС – МИКРОС-1, МИКРОС-2 и МИКРОС-Т – породил и соответствующий ряд генераций ПО [17]. .

В основу *операционной системы* МИКРОС положены следующие принципы:

- независимость от структуры ВС и от числа машин в ней;
- модульность построения;
- распределённость и децентрализованность модулей по машинам ВС;
- локальность связей между модулями;
- асинхронность взаимодействий модулей;
- развиваемость (изменяемость и пополняемость состава модулей, в частности, возможность замены программных модулей на аппаратурные);
- иерархичность построения: стратификация системы на уровни, каждый из которых строится на основе предыдущих и освобождает пользователя от специфических для уровня операций по погружению задачи в систему;
- преемственность с ОС базовых микропроцессорных средств (либо микроЭВМ “Электроника”, либо транспьютеров, в зависимости от моделей семейства МИКРОС).

Все генерации ОС (МИКРОС-1, МИКРОС-2, МИКРОС-Т) являются распределёнными и децентрализованными. Децентрализованная распределённая ОС МИКРОС способна функционировать в произвольной конфигурации ВС; ОС создаёт в каждой ЭМ “окружение”, позволяющее осуществлять динамическую настройку адаптирующейся параллельной программы на существующую конфигурацию ВС (или подсистемы). Децентрализованные процедуры маршрутизации обеспечивают передачу сообщений между любыми ЭМ системы. Указанные свойства ОС МИКРОС являются основой для поддержки живучести ВС (и, следовательно, для организации отказоустойчивых вычислений).

В среде *программирования* МИКРОС имеются языки параллельного программирования *Р*-ФОРТРАН и *Р*-С. Эти языки построены путём расширения соответствующих традиционных языков FORTRAN и С примитивами организации межмашинных взаимодействий и примитивами оценки параметров подсистем, на которых исполняются параллельные программы. Первые позволяют организовать взаимодействия между любыми ветвями программы, вторые дают возможность использовать параметры подсистемы для адаптации программы к текущей конфигурации последней. Это свойство существенно с двух точек зрения: простоты организации параллельных вычислений и отказоустойчивости. Реализация данных примитивов основывается на средствах распределённой децентрализованной операционной системы МИКРОС.

6.3.3. Архитектурные свойства систем семейства МИКРОС

Опишем архитектурные свойства ВС семейства МИКРОС.

Класс архитектуры любой модели ВС – это MIMD; допустима трансформация архитектуры MIMD в архитектуру MISD или SIMD путём программной перенастройки системы.

Класс ВС – система с программируемой структурой и с распределённым управлением.

Характер пространственного размещения вычислительных ресурсов – сосредоточенный или распределённый.

Основная функционально-структурная единица вычислительных ресурсов – элементарная машина.

Функции ЭМ – традиционные для ЭВМ функции по переработке информации плюс функции, связанные с управлением ВС в целом как коллектива (ансамбля) машин.

Масштабируемость ВС поддерживается аппаратурными средствами (системным устройством либо коммуникационным транспьютером) и программным обеспечением.

Количество N элементарных машин не фиксировано, что обеспечивает принципиально неограниченное наращивание производительности ВС.

Виды структуры сети межмашинных связей – произвольные (нерегулярные) графы.

Рекомендуемые структуры ВС:

- для сосредоточенных ВС: оптимальные D_n - и L(N, v, g)-графы (см. п. 4.7);
- для пространственно распределённых ВС: в условиях отсутствия жёстких технико-экономических ограничений те же оптимальные структуры, что и для сосредоточенных систем; в противном случае – любые структуры реально существующих сетей передачи информации или сетей, обеспечивающих при заданных ограничениях связность вычислительных ресурсов.

Наращиваемость (масштабируемость) *размерности* структуры ВС – $v = 1...8$.

Тип оперативной памяти – распределённая и общедоступная.

Аппаратурно-программная база системы:

- для моделей МИКРОС-1 и МИКРОС-2 – средства микромашинной техники и спецпроцессоров семейства “Электроника”;
- для модели МИКРОС-Т – транспьютерные средства семейства Inmos T800 (компании SGS-Thomson) и средства высокопроизводительных микропроцессоров.

Конфигурации ЭМ:

• для моделей МИКРОС-1 или МИКРОС-2 – всевозможные допустимые комплексы на основе микроЭВМ “Электроника 60М” или “Электроника 60-1” (или совместимых с ними других микроЭВМ), которые могут иметь в своём составе, в частности, спецпроцессоры “Электроника МТ-70” и “Электроника 1603”;

• для модели МИКРОС-Т – либо один из транспьютеров Т805 или Т800 (простейшая конфигурация), либо транспьютер и один из высокопроизводительных микропроцессоров: Intel 860, PowerPC, Alpha и дополнительная оперативная память (расширенная конфигурация).

Коммуникационные средства ЭМ для реализации функций управления системами:

• МИКРОС-1 или МИКРОС-2 – модули СУ, выполненные на полных платах конструктивов для микроЭВМ “Электроника 60М” или “Электроника 60-1”;

• МИКРОС-Т – транспьютер Inmos T805 (или Т800).

Число коммуникационных средств в одной ЭМ в системах:

• МИКРОС-1 или МИКРОС-2 – одно СУ в составе от одного до четырёх модулей;

• МИКРОС-Т – один транспьютер Inmos T805 (или Т800).

Программное обеспечение ВС:

• МИКРОС-1 или МИКРОС-2:

– распределённые децентрализованные ОС, являющиеся расширением ОС микроЭВМ “Электроника 60М” или “Электроника 60-1”;

– языки параллельного программирования *P*-FORTRAN и *P*-PASCAL, являющиеся языками семейства микроЭВМ “Электроника”, дополненными средствами организации системных взаимодействий.

• МИКРОС-Т:

– распределённая децентрализованная ОС МИКРОС-Т;

– языки параллельного программирования *P*-FORTRAN и *P*-C.

Режимы функционирования ВС:

• монопрограммный, обеспечивающий решение сложной задачи, при котором все ресурсы ВС используются для реализации параллельных программ и обеспечения требуемого уровня надёжности и живучести;

• мультипрограммные (обработка наборов и обслуживание потоков параллельных задач, разделение “времени и/или пространства” и др.), при которых для решения любой задачи или для обслуживания любого задания используется лишь часть ресурсов системы.

Способы обработки данных в ВС:

• распределённый (параллельный), когда однородно расчленённые данные и ветви параллельной программы их обработки рассредоточиваются по ЭМ;

• матричный, при котором программа вычислений размещается в одной или нескольких ЭМ, а данные (однородно) распределяются по всем ЭМ;

• конвейерный, когда сегментированная программа распределяется по машинам предварительно настроенного конвейера (или “кольца” или “линейки”) и обеспечивается последовательное “пропускание” данных через все ЭМ конвейера.

Рекомендуемая методика распараллеливания сложных задач – крупноблочное распараллеливание, позволяющее за счёт минимизации затрат на межмашинные взаимодействия достичь линейной зависимости производительности ВС от числа ЭМ.

Требуемые уровни производительности, ёмкости памяти, надёжности и живучести ВС достигаются путём подбора количества ЭМ и их состава, выбора структуры сети межмашинных связей, использования широких возможностей системных аппаратурно-программных средств по статической и динамической реконфигурации структуры и по варьированию состава системы.

Области применения ВС:

• традиционные сферы применения ЭВМ и векторных процессов, в которых возросли требования по обеспечению производительности, ёмкости памяти, надёжности и живучести и где целесообразно сохранить совместимость вычислительных средств;

• сферы применения, связанные с решением трудоёмких задач, таких как сложные задачи физики, механики сплошной среды, аэродинамики, баллистики, метеорологии, обработки изображений и речевых данных, задачи организации баз знаний, искусственного интеллекта;

• сложные большемасштабные системы, среди которых системы управления энергетическими установками, системы управления динамическими объектами и другие системы, характеризуемые высокой эффективностью, безотказностью, живучестью, развиваемостью, компактностью либо распределённостью своих ресурсов и т. п.

Таким образом, ВС семейства МИКРОС основываются на перспективных принципах обработки информации, строятся из аппаратурно-программных средств микропроцессорной техники, обладают гибкими возможностями по статической и динамической реконфигурации своих структур, позволяют достичь высокой производительности, надёжности и живучести в широкой области применения.

Продолжением ряда ВС МИКРОС-1, МИКРОС-2 и МИКРОС-Т являются высокопроизводительные ВС с массовым параллелизмом семейства МВС.

Опыт, приобретённый при создании мини-ВС и микропроцессорных систем, может быть положен в основу будущих разработок суперВС как ансамблей микропроцессоров, размещаемых на крупномасштабных полупроводниковых пластинах.

6.4. Пространственно-распределённая мультикластерная ВС

Кластерные ВС – параллельные средства обработки информации, интуитивная оценка архитектурных возможностей которых вытекает из семантики слова кластер (Cluster – группа). Такие системы получили широкое распространение уже в 1990 годах. В списке Top500 кластерные системы доминируют; их количество приближается к 400.

Термин “вычислительный кластер”, по-видимому, был впервые введён DEC (Digital Equipment Corporation). По определению DEC, кластер – это группа компьютеров, которые связаны между собой и функционируют как единое средство обработки информации. Из приведённого определения видно, что корпорация DEC, по сути, ввела синоним термину “вычислительная система”, а не особый тип средств обработки информации. Для создания кластерных ВС используются и MISD-, и SIMD-, и MIMD-архитектуры, различные функциональные структуры и конструктивные решения.

В наиболее общей трактовке *кластерная ВС*, *или кластер*, – это композиция множества вычислителей, сети связей между ними и программного обеспечения, предназначенная для параллельной обработки информации (в частности, реализации параллельных алгоритмов решения сложных задач). При формировании кластерной ВС могут быть использованы как стандартные промышленные компоненты, так и специально созданные средства. Однако в кластерных ВС, как правило, превалируют массовые аппаратурно-программные средства. Последнее, по существу, является принципом конструирования кластерных ВС, обеспечивающим их высокую технико-экономическую эффективность.

Начало XXI века ознаменовалось созданием сосредоточенных мультикластерных ВС (IBM RoadRunner состоит из 18 кластеров) и характеризуется переходом от “виртуальных метакомпьютеров” (использующих ресурсы нескольких суперкомпьютеров) к большемасштабным пространственно-распределённым мультикластерным ВС как макроколлективам рассредоточенных кластеров, взаимодействующих между собой через локальные и глобальные сети (включая всемирную сеть Internet).

Объединённым коллективом Лаборатории вычислительных систем Института физики полупроводников им. А.В. Ржанова СО РАН и Центром параллельных вычислительных технологий Сибирского государственного университета телекоммуникаций и информатики

(СибГУТИ) создана масштабируемая GRID-модель – пространственно-распределённая мультикластерная ВС.

Текущая терафлопсная конфигурация пространственно-распределённой ВС имеет в своём составе более 10 кластеров, расположенных в институтах СО РАН и СибГУТИ. Для формирования кластеров использовались, в частности, процессоры AMD Opteron и Intel Xeon (включая двух- и четырёхядерные). Мультикластерная ВС оснащена операционной системой GNU/Linux и специально разработанными подсистемами: мультипрограммирования, оптимизации вложения параллельных программ в ВС, анализа MPI-программ, самоконтроля и самодиагностики ВС, организации распределённой очереди задач, удалённого доступа и мониторинга ВС. В вычислительной системе имеются средства для разработки последовательных программ:

- компиляторы: GNU GCC, SUN, Intel;
- библиотеки: GNU GSL, AMD ACML, Intel MKL;
- средства отладки и анализа;

и параллельных программ:

- MPI: MPICH2, OpenMPI, Intel MPI;
- OpenMP: GCC, SUN, Intel;
- средства отладки и анализа: VampirTrace.

Пространственно-распределённая ВС используется не только в научных исследованиях, но и при подготовке специалистов в области GRID- и параллельных вычислительных технологий.

7. Заключение

Современные суперВС (пространственно сосредоточенные и распределённые) полностью основываются на модели коллектива вычислителей. Для них характерно, в частности, следующее: большемасштабность (массовый параллелизм), масштабируемость и стохастический характер ресурсов. Анализ и оптимизация функционирования суперВС относятся к числу сложных проблем, которые не могут быть решены при помощи традиционного аппарата теории массового обслуживания и методов математического программирования.

Анализ эффективности ВС:

- живучая ВС (специальная виртуальная модель) и континуальные модели составляют базу для анализа эффективности функционирования большемасштабных распределённых ВС;
- полученные результаты описывают функционирование ВС как в переходном, так и стационарном режимах;
- нет никаких вычислительных трудностей при экспресс-анализе осуществимости параллельного решения и потенциальной эффективности (надёжности, живучести и технико-экономической эффективности) распределённых ВС с произвольным числом N элементарных машин;
- для выбора аппаратурной избыточности $N-n$ большемасштабной ВС и количества m “восстанавливающих устройств” достаточно воспользоваться формулами

$$1 \le (N-n) \le \lceil \lg N \rceil, \quad 1 \le m \le \lceil \lg N \rceil.$$

Параллельное мультипрограммирование:

- распределённая ВС – большемасштабный вероятностный объект, обслуживающий стохастические потоки параллельных задач;

- техника теории игр и стохастическое программирование составляют основу для организации стохастически оптимального использования ресурсов ВС;
- стохастическая оптимизация функционирования распределённых ВС осуществляется однократно для достаточно большого интервала времени;
- параллельные алгоритмы и теоретико-игровые, и стохастического программирования реализуются эффективно на распределённых ВС;
- нет сложных вычислительных проблем при создании распределённой операционной системы, поддерживающей параллельное мультипрограммирование;
- разработанный алгоритмический и программный инструментарий вложения параллельных программ в иерархические ВС эффективнее стандартных MPI-утилит.

Практика ВС с программируемой структурой:

- результаты многолетней эксплуатации созданных систем показывают высокую эффективность архитектурных решений, присущих концепции ВС с программируемой структурой;
- эта концепция позволяет в условиях современных возможностей производства средств микропроцессорной техники строить промышленные ВС, множество конфигураций которых составляют семейства (ряды) совместимых экономичных моделей для широкого диапазона по производительности, надёжности и живучести;
- примером служит ВС МИКРОС, подход в реализации которой допускает формирование требуемых сосредоточенных и распределённых моделей из серийных средств микропроцессорной техники и обеспечивает на многие годы возможность эволюционного совершенствования системы в соответствии с развитием технологии БИС;
- результаты в области пространственно-распределённых мультикластерных ВС составляют основу при конфигурировании государственных и региональных информационно-вычислительных комплексов.

Концепция ВС с программируемой структурой позволяет создавать технико-экономически эффективные средства обработки информации, обладающие сверхвысокой производительностью, надёжностью и живучестью.

Литература

1. Нейман Дж. фон. Теория самовоспроизводящихся автоматов: Пер. с англ./ Под ред. В.И. Варшавского. – М.: Мир, 1971. – 382 с.
2. Сергей Алексеевич Лебедев. К 100-летию со дня рождения основоположника отечественной электронной вычислительной техники. – М.: Физматлит, 2002. – 440 с.
3. Мельников В.А., Митропольский Ю.И., Шнитман В.З. Архитектура высокопроизводительной вычислительной системы “Электроника СС БИС-1” // Программные продукты и системы. 1992. № 1.
4. Бурцев В.С. Параллелизм вычислительных процессов и развитие архитектуры суперЭВМ. – М.: ИВВС РАН, 1997. – 152 с.
5. Иванников В.П. Архитектура операционной системы суперЭВМ. – М., 1984.
6. Томилин А.Н. Использование моделирования в анализе и разработке вычислительных систем // Труды Шестого международного семинара “Распределённая обработка информации” / Под ред. В.Г. Хорошевского. – Новосибирск: СО РАН, 1998. С. 273–277.
7. Каляев А.В. Однородные коммутационные регистровые структуры. – М.: Советское радио, 1978. – 335 с.
8. Каляев А.В. Многопроцессорные системы с программируемой структурой. – М.: Радио и связь, 1984. – 240 с.
9. Каляев И.А., Левин И.И., Семерников Е.А., Шмойлов В.И. Реконфигурируемые мультиконвейерные вычислительные структуры. – Ростов-на-Дону: ЮНЦ РАН. – 320 с.

10. Поспелов Д.А. Введение в теорию вычислительных систем. – М.: Советское радио, 1972. – 280 с.
11. Евреинов Э.В., Прангишвили И.В. Цифровые автоматы с настраиваемой структурой (Однородные среды). М.: Энергия, 1974. – 240 с.
12. Прангишвили И.В., Стецюра Г.Г. Микропроцессорные системы. – М.: Наука, 1980.
13. Пухов Г.Е., Евдокимов В.Ф., Синьков М.В. Разрядно-аналоговые вычислительные системы. – М.: Советское радио, 1978. – 255 с.
14. Велихов Е.П. Об организации в Академии наук СССР работ по информатике, вычислительной технике и автоматизации // Вестник АН СССР. 1983. № 6. С. 24–38.
15. Евреинов Э.В., Хорошевский В.Г. Однородные вычислительные системы. – Новосибирск: Наука, 1978. 320 с.
16. Хорошевский В.Г. Инженерный анализ функционирования вычислительных машин и систем. – М.: Радио и связь, 1987. 256 с.
17. Хорошевский В.Г. Архитектура вычислительных систем. – 2-е изд., перераб. и доп. – М.: МГТУ им. Н.Э. Баумана, 2008. 520 с.
18. Евреинов Э.В., Косарев Ю.Г. О возможности построения вычислительных систем высокой производительности. – Новосибирск: СО АН СССР, 1962. – 39 с.
19. Евреинов Э.В., Косарев Ю.Г. Однородные универсальные вычислительные системы высокой производительности. – Новосибирск: Наука, 1966. – 308 с.
20. Яненко Н.Н. Перспективы развития вычислительной математики на основе вычислительных систем // Препринт “ЭВМ. Перспективы и гипотезы”. – Новосибирск: ИТПМ СО АН СССР, 1981. № 46. С. 3–6.
21. Хорошевский В.Г. Модели функционирования большемасштабных распределённых вычислительных систем // Электросвязь. 2004. № 10. С. 30–34.
22. Монахов О.Г., Монахова Э.А. Исследование топологических свойств регулярных параметрически описываемых структур вычислительных систем // Автометрия. 2000. № 2. С. 70–82.
23. Хорошевский В.Г. Архитектурные концепции, анализ и организация функционирования вычислительных систем // Труды конференции “Моделирование-2008”. – Киев: ИПМЭ им. Г.Е. Пухова НАН Украины. 2008. Т.1. С. 15–24.
24. Хорошевский В.Г., Курносов М.Г. Алгоритмы распределения ветвей параллельных программ по процессорным ядрам вычислительных систем // Автометрия. 2008. № 2. С. 56–67.
25. Хорошевский В.Г., Курносов М.Г. Моделирование алгоритмов вложения параллельных программ в структуры распределённых вычислительных систем // Труды конференции “Моделирование-2008”. – Киев: ИПМЭ им. Г.Е. Пухова НАН Украины, 2008. Т.2. С. 435–440.
26. Хорошевский В.Г., Седельников М.С. Эвристические алгоритмы распределения задач по машинам вычислительной системы // Автометрия. 2000. Т.40. № 4. С.76–87.
27. Хорошевский В.Г., Власюк В.В. Теоретико-игровой подход к организации стохастически оптимального функционирования распределённых вычислительных систем // Автометрия. 2000. № 4. С.17–25.
28. Хорошевский В.Г., Мамойленко С.Н. Стратегии стохастически оптимального функционирования распределённых вычислительных систем // Автометрия. 2003. Т.39. № 2. С. 81–91.
29. Хорошевский В.Г., Подаков М.Н. Поиск стохастически оптимального разбиения большемасштабных вычислительных систем // Автометрия. 2000. № 2. С. 52–59.
30. Дрешер М. Стратегические игры. Теория и приложения. – М.: Сов. Радио, 1964. – 352 с.
31. Хедли Дж. Нелинейное и динамическое программирование. – М.: Мир, – 1967. – 506 с.
32. Хорошевский В.Г. Вычислительные системы с программируемой структурой // Международная научная конференция, посвященная 80-летию со дня рождения акаде-

мика В.А. Мельникова. Сборник докладов. – М.: Научный Фонд “Первая исследовательская лаборатория имени академика В.А. Мельникова, 2009. С. 51–62”.
33. Хорошевский В.Г. Вычислительная система МИКРОС // Препринт. – Новосибирск: ИМ СО АН СССР, 1981. № 38 (ОВС-19). – 52 с.
34. Транспьютеры. Архитектура и программное обеспечение / Под ред. Г. Харпа. – М.: Радио и связь, 1993. – 303 с.

Статья поступила в редакцию 07.04.2010

Хорошевский Виктор Гаврилович

Член-корреспондент Российской академии наук, профессор, доктор технических наук, заслуженный деятель науки Российской Федерации, заведующий Лабораторией вычислительных систем Института физики полупроводников им. А.В. Ржанова СО РАН, директор Центра параллельных вычислительных технологий, заведующий Кафедрой вычислительных систем Сибирского государственного университета телекоммуникаций и информатики. Область научных исследований – архитектура распределённых вычислительных систем (ВС), параллельное мультипрограммирование, методы организации функционирования ВС в моно- и мультипрограммных режимах, методы анализа надёжности, живучести и технико-экономической эффективности функционирования большемасштабных ВС, осуществимость параллельного решения задач.

Тел.&факс: (383) 333-21-71, 269-82-75; e-mail: khor@isp.nsc.ru, khor@sibsutis.ru

Distributed programmable structure computer systems

V.G. Khoroshevsky

Conception and research results of Distributed Computer Systems (DCS) with programmable structure are stated. Functioning models and continuous methods are described to calculate indices of DCS efficiency. Results of analysis and synthesis of DCS structures are given. Stochastic strategies are suggested to optimize DCS functioning in parallel multiprogramming modes. Functional structures of embodied DCS (“Minsk-222”, MINIMAX, SUMMA, MICROS and geographical-distributed multicluster system) are considered. It is showed the given conception let us create technical-economic effective and robust supercomputers.

Keywords: computer systems, efficiency analysis, structure, parallel multiprogramming.