РОССИЙСКАЯ АКАДЕМИЯ НАУК

СИБИРСКОЕ ОТДЕЛЕНИЕ

А В Т О М Е Т Р И Я

2004, том 40, № 1

УДК 681.324

В. Г. Хорошевский, С. Н. Мамойленко, Ю. С. Майданов, С. В. Смирнов

(Новосибирск)

ОБ ОРГАНИЗАЦИИ ФУНКЦИОНИРОВАНИЯ КЛАСТЕРНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ*

Предлагается подход к организации режима (само)диагностики распределенных вычислительных систем (ВС). Развиваются теоретико-игровые методы организации диспетчеризации ВС в режиме обслуживания потока параллельных задач. Описывается архитектура и программное обеспечение живучей распределенной кластерной вычислительной системы.

В общем случае распределенная вычислительная система (ВС) – это композиция сосредоточенных параллельных ВС и сети связей, через которую осуществляются системные взаимодействия [1, 2]. Компоненты распределенных систем могут находиться на значительных расстояниях друг от друга, поэтому при передаче информации по каналам связи могут возникать значительные временные задержки (существенно превышающие среднее время выполнения операции). Число элементарных машин (ЭМ) или процессоров в таких системах допускает варьирование и может изменяться в пределах от 10 до 10^6. Например, вычислительная система Cray X1 (52,4 Tera-FLOPS) может иметь в своем составе 4096 процессоров (49152 векторных конвейера и скалярных блока обработки информации); создаваемая система IBM Blue Gene (1 PetaFLOPS) будет состоять из 1 000 000 процессоров. Именно поэтому подобные ВС относят к масштабируемым и большемасштабным.

В качестве параллельных ВС могут выступать кластерные системы с любой архитектурой, при построении которых применяются, как правило, серийные коммерческие аппаратные и программные средства. Кроме того, в кластерных ВС целесообразно использовать и специальные программные средства, поддерживающие, например, отказоустойчивые параллельные вычислительные технологии. Такие системы позволяют достичь высокой производительности при оптимальном соотношении цены и показателей качества функционирования.

* Работа выполнена при поддержке Российского фонда фундаментальных исследований (гранты № 02-07-90379, № 02-07-90380).

Концептуальные основы построения распределенных ВС с программируемой структурой развиваются в Сибирском отделении РАН с 60-х годов 20-го столетия. Первые реализации таких систем из серийных компьютеров, по сути, и были кластерными ВС [1–4]. Функционирование этих систем основывается на фундаментальных принципах модели коллектива вычислителей [1, 3]:

– массовом параллелизме (параллельном выполнении большого числа операций);

– программируемости (автоматическом перестраивании или реконфигурировании) структуры;

– конструктивной однородности.

1. **Режимы функционирования ВС.** Распределенные ВС по своей природе (например, вследствие отказов ресурсов) являются стохастическими объектами. Такие ВС предназначаются в общем случае для обслуживания вероятностных потоков заданий, представленных параллельными программами со случайными параметрами (рангом – числом ветвей, временем решения и т. п.). Решение поступающих задач может быть организовано на распределенных ВС (или их подсистемах) в следующих режимах:

– решение одной сложной задачи;

– обработка набора задач;

– обслуживание потока задач.

В первом режиме (моно- или однопрограммном) все ресурсы ВС выделяются для решения одной сложной задачи. Во втором и третьем режимах (мультипрограммных) ресурсы системы разделяются между несколькими задачами. В этих режимах имеет место и распределение «пространства» ресурсов, и разделение времени ВС. В любой момент функционирования в системе существует множество подсистем, на каждой из которых решается своя параллельная задача. В общем случае множество подсистем реконфигурируется во времени (по мере решения задач).

Ясно, что в условиях большемасштабности и неабсолютной надежности компонентов ВС актуальными являются проблемы обеспечения их живучести [3] и оптимизации функционирования в мультипрограммных режимах. Решение проблемы живучести связано с поиском таких архитектурных и программных решений, которые позволили бы использовать все работоспособные ресурсы ВС для решения задач.

2. **Обеспечение живучести ВС.** Пусть распределенная ВС состоит из не абсолютно надежных элементарных машин. Реализация параллельных программ осуществляется на основной подсистеме из n ЭМ; $(N - n)$ машин составляют структурную избыточность [3]. Для обеспечения живучести ВС необходимо предусмотреть следующие операции:

– контроль работоспособности ресурсов основной подсистемы;

– диагностирование (локализация неисправностей);

– восстановление поврежденных ресурсов основной подсистемы (реконфигурация ВС);

– возобновление вычислительных процессов после реконфигурации ВС.

Операции контроля ВС должны обеспечивать безошибочное определение состояний составляющих ресурсов. Для обнаружения нарушения функций ресурсов ВС используются методы функционального контроля (встроенные схемы, таймеры, тесты), а также независимый контроль другими модулями, граничное сканирование и т. д. В качестве «ресурса» в распределенных ВС выступает, как правило, элементарная машина.

В случае обнаружения отказа в основной подсистеме ВС реализуются процедуры диагностирования и восстановления или поиска в пределах основной подсистемы отказавшей ЭМ и реконфигурации структуры ВС в целом. Поиск отказавшей ЭМ может осуществляться только средствами самой ВС или, как говорят, средствами самодиагностики.

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 реконфигуратором.

Средства (само)диагностики и реконфигуратор являются компонентами операционной системы (ОС). Эта композиция, по сути, является виртуальным восстанавливающим устройством (ВУ). В соответствии с принципом децентрализованного управления можно считать, что в каждой ЭМ есть свое ВУ и, следовательно, в распределенной ВС количество ВУ равно количеству ЭМ, входящих в ее состав.

В распределенных вычислительных системах элементом замены является ЭМ – функционально-конструктивный модуль, который сам по себе располагает средствами поддержки живучести. Структура системы должна удовлетворять требованиям локальности и однородности [1–3]. В силу однородности каждый элемент окрестности любой ЭМ в структуре ВС является для нее эталоном.

Опишем программные компоненты для организации (само)диагностирования ВС. Целью создания диагностических средств является организация процессов выявления неработоспособных ресурсов и их локализации. При разработке таких средств основополагающими являются следующие принципы [5]:

– в начальный момент функционирования все ресурсы ВС считаются исправными;

– процесс поиска неисправных ресурсов производится через определенные промежутки времени;

– процедура диагностики децентрализована, т. е. каждая ЭМ обеспечивает механизм самодиагностики ВС в целом;

– функции самодиагностирования ВС реализуются программными диагностическими модулями элементарных машин;

– все диагностические модули идентичны, в состав каждого из них входит набор тестовых задач, по результатам выполнения которых можно судить о работоспособности ЭМ;

– функции самодиагностирования должны быть максимально прозрачными для пользователей системы.

В процессе решения пользовательских задач функционирование каждой ЭМ системы основывается на ветви параллельной программы, дополненной диагностическим модулем (ДМ). Этот модуль состоит из нескольких компонентов, основным из которых является контролер. Контролер ДМ осуществляет настройку программы пользователя для работы с диагностическими компонентами, запуск тестовых задач, формирование и отправку данных о своем состоянии другим диагностическим модулям, принимает решения о состоянии «соседних» ДМ (модулей соседних ЭМ) и передает информацию пользовательской программе об изменении структуры ВС. Соседними ЭМ считаются машины, между которыми имеется непосредственная связь в структуре ВС.

Следующим компонентом ДМ является диагностический сервер. Он осуществляет асинхронный прием от соседних ДМ информации об их состоянии и при необходимости передачу этой информации контролеру. Кроме того, в состав ДМ входит набор тестовых задач, выполнение которых позволяет оценить работоспособность самого ДМ и его соседей.

Таким образом, благодаря работе диагностических модулей и периодическому взаимодействию между ветвями параллельной программы осуществляется (само)диагностика ВС.

3. **Оптимизация функционирования ВС.** Допустим, что на распределенную вычислительную систему поступает случайный поток параллельных задач. Каждая задача $I_j^r \in \Im$ представлена парой чисел $\langle t_j^r, r\rangle$, где t_j^r – время решения задачи, а r – ее ранг. Считается, что задача имеет ранг $r \in E^* = \{0,1,2,\ldots,n\}$, если для ее решения необходимо использовать r машин ВС (подсистему ранга r).

В случае невысокой интенсивности потока алгоритмы распределения поступающих задач по элементарным машинам ВС могут быть построены на основе методов математического программирования [1–3]. Анализ эффективности и организация функционирования ВС при высокой интенсивности потока задач могут быть выполнены с помощью теории массового обслуживания, теории игр и стохастического программирования [1, 6, 7].

Опишем теоретико-игровую модель функционирования ВС. Для решения на ВС поступает поток задач достаточно высокой интенсивности, такой, что на входе системы имеется их конечная очередь. Каждая ожидающая решения задача $I_j^r \in \Im$ представлена парой чисел $\langle t_j^r, r\rangle$. Время решения задачи t_j^r – случайная величина, распределенная по некоторому закону $F(t)$. Пусть $\Im^r$ – подмножество задач I_j^r ранга r: $\bigcup_j I_j^r = \Im^r$, $r \in E^*$. Очевидно, что $\Im^r \subset \Im$, $\bigcup_r \Im^r = \Im$, $\Im^r \cap \Im^k = \varnothing$ для $r \neq k$, $r, k \in E^*$. Считается, что число элементов в подмножестве $\Im^r \subset \Im$ достаточно большое, такое, что допускается разбиение $\Im^r$ на группы G^r задач с суммарным временем решения, не превышающим некоторой величины T.

Имеется диспетчер – компонент распределенной операционной системы; он выбирает из очереди пакет ранга r и выделяет ему время Θ для решения на подсистеме ранга r.

Следуя игровой терминологии, будем называть объекты «очередь» и «диспетчер» игроками. Фактически эти игроки являются виртуальными, они представлены соответствующими компонентами ОС.

Затраты на решение той или иной задачи G^r, $r \in E^*$, будем интерпретировать как «платеж» очереди диспетчеру ОС. Требуется создать такой алгоритм работы комплекса «очередь – диспетчер», который в условиях случайности параметров входящих задач обеспечивал бы в единицу времени минимум затрат на их решение.

Далее, считаем, что очередь и диспетчер используют чистые стратегии n и Θ соответственно, если первый игрок для решения формирует пакет из n задач со средним временем решения $\sum_j t_j^r = nM \leq T$, где M – математическое ожидание величины t_j^r, а второй игрок назначает время решения пакета рав-

ным Θ. Если очередь выбирает стратегию с номером n, а диспетчер стратегию с номером Θ, то очередь «платит» диспетчеру сумму $c_{n\Theta}$. Элементы $c_{n\Theta}$ составляют матрицу C платежей и могут быть определены следующим образом:

$$c_{n\Theta} = \begin{cases} nMc_1 + (\Theta - nM)c_2, & \text{если } \Theta \geq nM, \\ \Theta c_2 + (nM - \Theta)c_3 & \text{в противном случае,} \end{cases}$$

где c_1 – платеж очереди за решение задач в единицу времени; c_2 – потери диспетчера в единицу времени за простой ресурсов ВС; c_3 – штраф, накладываемый на диспетчера в единицу времени за досрочное прерывание решения задач.

Предложенная модель является одной из возможных интерпретаций теоретико-игрового подхода к организации функционирования распределенных ВС. Модель относится к классу непрерывных (бесконечных) антагонистических игр двух объектов с нулевой суммой, так как интересами диспетчера является получение максимального дохода от выделенных ресурсов, а очереди – минимизация затрат на решение задач.

4. **Программное обеспечение распределенных ВС.** Программное обеспечение (ПО) – это композиция связанных программ, позволяющих эффективно использовать ресурсы вычислительной системы (рис. 1). Под ресурсами понимается совокупность технических и программных средств ВС, используемых многократно. Примерами вычислительных ресурсов могут служить элементарная машина, процессор, память, внешние устройства, некоторая библиотека программ и другие.

Программное обеспечение условно можно разделить на три части:

– операционную систему;

– систему параллельного программирования;

– комплекс пакетов прикладных программ.

Операционная система – это программа, предназначенная для управления вычислительными ресурсами. Иногда ОС называют управляющей системой, управляющим ядром или просто ядром.

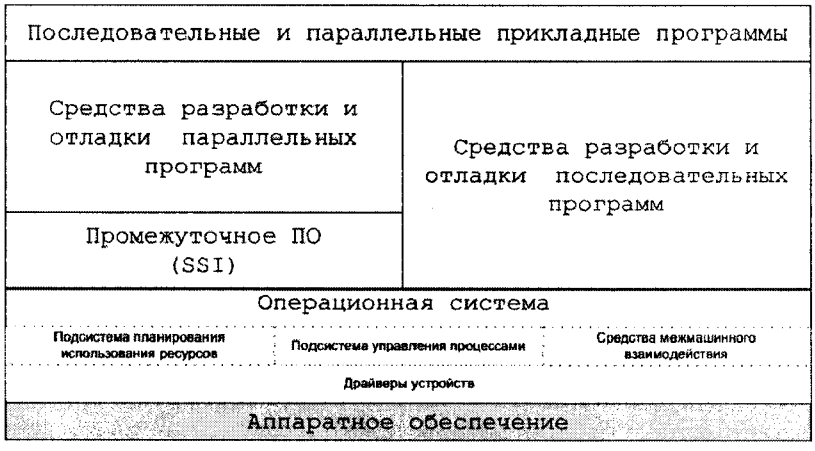

Рис. 1. Структура программного обеспечения кластерной вычислительной системы

Управление ресурсами ВС включает решение следующих общих проблем:

– планирование использования ресурса (кому, когда и на какое время следует выделить данный ресурс);

– удовлетворение запросов на ресурсы;

– отслеживание состояния и учет использования ресурса (занят или свободен ресурс и какая доля ресурса уже распределена);

– разрешение конфликтов доступа к ресурсу;

– организация отказоустойчивого использования ресурсов.

Для решения этих задач в ОС используется комплекс алгоритмов, которые и определяют состав и структуру ОС в целом. Задача управления ресурсами ВС является весьма сложной. Как уже отмечалось, сложность в большой степени определяется случайным характером возникновения запросов на ресурсы.

В распределенной ВС каждая ЭМ работает под управлением собственной локальной ОС, которая и определяет функционирование ее ресурсов. Для осуществления функций управления распределенными ресурсами ВС каждая локальная ОС взаимодействует с операционными системами соседних ЭМ. Распределенная ОС представляется композицией локальных операционных систем; эта композиция функционирует как единое целое, но каждая из локальных ОС решает свой набор функций по управлению ресурсами только в своей ЭМ. Систему по управлению распределенными ресурсами ВС также называют программным обеспечением создания единого образа системы (Single System Image (SSI)) или промежуточным программным обеспечением.

В основе организации параллельных вычислений на ВС лежит представление программ в виде композиции совместно протекающих асинхронно взаимодействующих процессов [2, 8–10]. Под процессом понимается совокупность последовательных действий (операций) при реализации некоторого алгоритма. Совокупность процессов означает не только обычную для мультипрограммных систем или систем разделения времени реализацию алгоритмически независимых процессов, но и существование связи между отдельными процессами, которая обусловлена тем, что они представляют собой части одного сложного алгоритма. Базовый набор, обеспечивающий управление процессами, составляют средства для порождения и уничтожения процессов и для реализации взаимодействия между ними.

В общем случае ОС имеет иерархическую структуру. За начальное условие для решения задач управления берется описание операционной обстановки в ВС. Она включает сведения о текущем состоянии реализуемых процессов и о распределении между ними ресурсов ВС. Описание операционной обстановки разбивается на уровни, соответствующие функциональной обособленности ресурсов ВС (ресурсы одной ЭМ обособлены от ресурсов остальных машин). Каждому уровню функциональной обособленности ресурсов ВС соответствует свой уровень операционной системы. Каждый уровень ОС допускает представление в виде совокупности процессов, реализуемых в различных ЭМ.

Система параллельного программирования (Р-программирования) предназначается для разработки программ и включает в себя различные трансляторы (компиляторы), библиотеки программ и средства отладки и оценки производительности ВС.

Разработка параллельных программ может основываться на одной из двух парадигм: параллелизме данных или параллелизме задач. В первом случае параллельная программа реализует одну и ту же операцию над различными частями исходных данных. При разработке таких программ часто используются специализированные языки программирования и трансляторы для них, например: HPF, Cilk, C*, ZPL, FORTRAN M, MpC и другие.

Вторая парадигма предусматривает разбиение вычислительной задачи на множество слабо связанных подзадач – ветвей. Каждая подзадача-ветвь при этом выполняется на своей ЭМ [1]. Для координации процесса вычислений подзадачи обмениваются между собой сообщениями. Практически такой обмен осуществляется при помощи процедур специализированных библиотек, которые являются дополнением к обычным последовательным языкам программирования. Впервые такая парадигма нашла практическое воплощение в программном обеспечении ВС с программируемой структурой «Минск-222» в 1965 г. [1]. Последними примерами специализаций по разработке библиотек передачи сообщений являются MPI (Message Passing Interface) и PVM (Parallel Virtual Machine).

Спецификация MPI была разработана специальным комитетом MPIF (Message Passing Interface Forum) в 1994 г. На сегодняшний день существует несколько различных библиотек, реализующих спецификацию MPI. Наибольшую популярность получили MPICH (MPI CHameleon) и LAM (Local Area Machine). Спецификации PVM соответствует библиотека с одноименным названием PVM, которая разработана в Окриджской национальной лаборатории Университета Теннеси и Эмори (США).

5. **Кластерная система Центра параллельных вычислительных технологий СибГУТИ.** В Научно-учебном центре параллельных вычислительных технологий (ЦПВТ) Сибирского государственного университета телекоммуникаций и информатики создана и развивается живучая распределенная кластерная вычислительная система. Целью работы является решение следующих задач:

– построение аппаратурно-программного инструментария для проведения научных исследований и подготовки специалистов в области параллельных вычислительных технологий [11];

– создание методов и алгоритмов, обеспечивающих живучесть распределенных вычислительных систем;

– разработка методов и алгоритмов организации оптимального функционирования кластерных распределенных ВС и GRID-систем в мультипрограммных режимах;

– отработка средств анализа функционирования большемасштабных вычислительных систем.

5.1. *Архитектура кластерной ВС.* Для создания кластерной системы использовались широко распространенные персональные компьютеры с архитектурой IBM PC. Функционирующая кластерная ВС состоит из пяти сегментов, каждый их которых является сосредоточенным многомашинным кластером. Любой из сегментов способен функционировать как автономно, так и в составе ВС (рис. 2). Кластерная ВС ЦПВТ имеет выходы в Internet и модемные соединения с телефонной линией, что позволяет осуществлять взаимодействие с другими кластерами (в частности, с кластером ИФП СО РАН), т. е. допускает масштабирование. Таким образом, кластерная ВС ЦПВТ является базой для формирования распределенной в пространстве конфигурации.

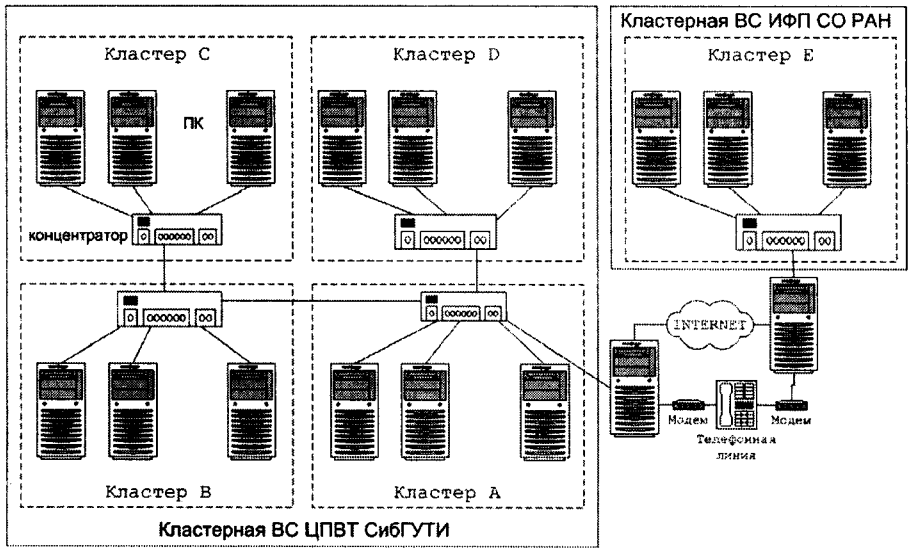

Рис. 2. Архитектура кластерной вычислительной системы

Для организации межмашинных связей в системе используется стандартная сетевая технология Fast Ethernet, позволяющая передавать данные по сети со скоростью 100 Мбит/с. В кластерной ВС использована топология типа звезда, для формирования кластеров-сегментов применены коммутаторы фирмы "3COM". Один из ПК кластера А (см. рис. 2) выполняет функции сервера локальной вычислительной сети и имеет модемный выход и выход во всемирную сеть.

Состав кластерной вычислительной системы неоднороден, поэтому ее можно рассматривать как GRID-систему. Архитектура кластерной ВС позволяет разрабатывать методы и алгоритмы функционирования, применимые как в сосредоточенных, так и в распределенных системах.

5.2. *Стандартное программное обеспечение кластерной ВС (рис. 3).* Все персональные компьютеры функционируют под управлением ОС Linux (дистрибутив ASPLinux v.7.3, ядро 2.4.19). Для разработки и реализации параллельных программ используется технология MPI (реализация LAM v.6.5.9). Для межкластерных взаимодействий через телефонный канал используется пакет программ, реализующий протокол взаимодействия Point-to-Point (PPP). В состав программного обеспечения также входят компоненты, позволяющие получить удаленный доступ к кластеру (ssh, telnet).

5.3. *Специальные компоненты для организации оптимального отказоустойчивого функционирования кластерных ВС.* Стандартное программное обеспечение кластерных ВС рассчитано на многопрограмный режим решения сложных задач. Однако это ПО не рассчитано на поддержку живучести ВС. Поэтому в ЦПВТ разрабатываются программные компоненты, которые позволят организовать живучее функционирование систем в режимах решения, набора и обслуживания потока задач, представленных параллельными программами с различным числом ветвей.

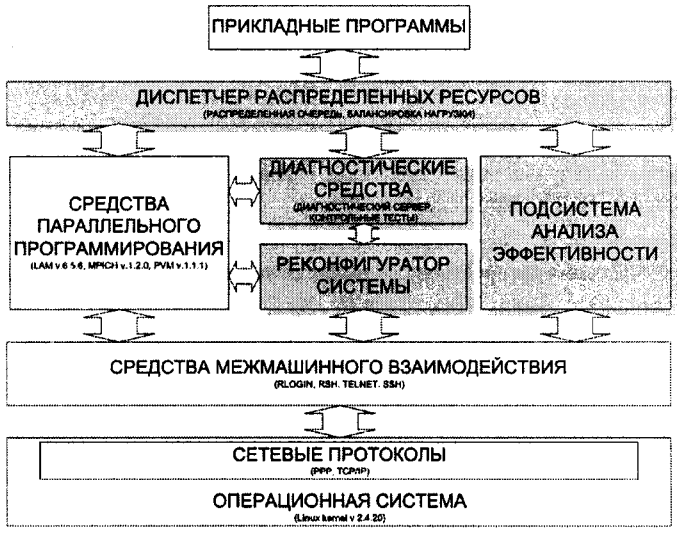

Рис. 3. Программное обеспечение кластерной ВС

Разработана программа, которая обеспечивает функции самодиагностики ВС, определяет состояние системы, выявляя неработоспособные ресурсы, и передает результат программе пользователя. Достоинством данной программы является возможность модификации используемых тестов, проверяющих работоспособность ресурсов, в зависимости от задач, решаемых на вычислительной системе. Кроме того, алгоритм, лежащий в основе дешифрации результатов теста, также допускает адаптацию. Пользователь имеет возможность настраивать параметры, определяющие работу программы, путем изменения файла конфигурации. Последнее обеспечивает возможность адаптации программы для вычислительных систем с различными характеристиками и позволяет учесть особенности решаемых задач. По сути, имеются средства для моделирования функционирования большемасштабных распределенных вычислительных систем со средствами диагностики.

Итак, создан аппаратурно-программный инструментарий для исследований в области параллельных вычислений и Grid-технологий.

СПИСОК ЛИТЕРАТУРЫ

1. **Евреинов Э. В., Хорошевский В. Г.** Однородные вычислительные системы. Новосибирск: Наука, 1978.
2. **Хорошевский В. Г.** Вычислительная система МИКРОС. Новосибирск, 1983. (Препр. /ИМ СО АН СССР: ОВС–19).
3. **Хорошевский В. Г.** Инженерный анализ функционирования вычислительных машин и систем. М.: Радио и связь, 1987.
4. **Корнеев В. В., Хорошевский В. Г.** Вычислительные системы с программируемой структурой // Электронное моделирование. 1979. № 1. С. 42.

5. **Майданов Ю. С.** Формирование базиса для исследования децентрализованных алгоритмов самодиагностики кластерных вычислительных систем // Материалы Междунар. науч.-техн. конф. «Информатика и проблемы телекоммуникаций». Новосибирск: СибГУТИ, 2002. С. 130.

6. **Смирнов С. В.** Параллельный стохастический алгоритм распределения набора задач // Материалы Междунар. науч.-техн. конф. «Информационные системы и технологии». Новосибирск: НГТУ, 2000.

7. **Хорошевский В. Г., Мамойленко С. Н.** Стратегии стохастически оптимального функционирования распределенных вычислительных систем // Автометрия. 2003. **39**, № 2. С. 81.

8. **Корнеев В. В., Хорошевский В. Г.** Архитектура вычислительных систем с программируемой структурой. Новосибирск, 1979. (Препр. /ИМ СО АН СССР; ОВС–10).

9. **Корнеев В. В., Хорошевский В. Г.** Структура и функциональная организация вычислительных систем с программируемой структурой. Новосибирск, 1979. (Препр. /ИМ СО АН СССР; ОВС–11).

10. **Воеводин В. В., Воеводин Вл. В.** Параллельные вычисления. С.-Пб.: БХВ-Петербург, 2002.

11. **Хорошевский В. Г., Майданов Ю. С., Мамойленко С. Н. и др.** Живучая кластерная вычислительная система // Тр. шк.-сем. «Распределенные кластерные вычисления». Красноярск: Изд-во Краснояр. гос. ун-та, 2001.

Институт физики полупроводников ОИФП СО РАН,
Сибирский государственный университет
телекоммуникаций и информатики,
E-mail: khor@isp.nsc.ru,
msn@neic.nsk.su

Поступила в редакцию
28 августа 2003 г.