УДК 681.512.54

МИРОШНИК М.А. д.т.н., профессор,
КЛИМЕНКО Л.А. к.т.н., доцент (УкрГАЖТ)

Размещение подзадач в распределенных вычислительных системах кластерно-метакомпьютерного типа

В статье проведена классификация стратегий размещения подзадач в распределенных вычислительных системах и разработана стратегия размещения узлов распределенной вычислительной системы, универсальная расчетно-обменная характеристика задач, решаемых в системе распределенных вычислений и статическая стратегия размещения подзадач в системе распределенных вычислений. Проведена проверка полученных теоретических результатов на реальной вычислительной системе как для расчетно-ориентированных задач (вычисление экстремумов функций и численное интегрирование), так для обменно-ориентированных задач (умножение матриц), которая доказала эффективность предложенных решений.
Ключевые слова: *распределенная вычислительная систем, кластерная система, метакомпьютерный тип.*

Введение

В настоящее время фундаментальные и прикладные проблемы, эффективное решение которых возможно только с использованием высокопроизводительных вычислений, объединены понятием "Grand challenges", которое включает в себя, например, задачи предсказания климата и глобальных изменений в земной коре, задачи аэродинамики для самолетостроения и создания; реактивных двигателей, распознавание изображений при навигации движущихся объектов, и т.д.

При решении этих задач возникает потребность в значительно больших вычислительных ресурсах, чем может предоставить обычный компьютер. Таким задачам необходимо либо большее быстродействие, либо возможность обрабатывать и хранить большие объемы информации.

Большого быстродействия требуют сложные, многомерные задачи; которые необходимо решить в течение определенного, достаточно ограниченного времени. Один из наиболее характерных примеров - задачи прогноза погоды. Область решения (атмосфера) разбивается на отдельные пространственные; ячейки, причем для расчета временных изменений, вычисления в каждой ячейке повторяются много раз. Если объем ячейки равен 1 км3, то для моделирования слоя атмосферы высотой в 10 км потребуется 8 * 5x10 таких ячеек. Предположим, что вычисления в каждой ячейке потребуют 200 операции с плавающей точкой, тогда за один временной шаг потребуется выполнить 1011 операций с плавающей точкой. Для того чтобы произвести расчет прогноза погоды с заблаговременностью 10 дней с 10-ти минутным шагом по времени, компьютеру производительностью 100 MFlops потребуется свыше 100 дней. Для того чтобы произвести расчет за 10 мин, потребуется уже компьютер производительностью 1.7 TFlops.

Таким образом, поскольку для решения подобных задач существует спрос на вычислительные системы повышенной мощности, современный рынок предлагает суперкомпьютеры, отвечающие достаточно высоким требованиям. К ним относятся комплексы, собранные из специальных комплектующих, с использованием, как правило, векторных или матричных процессоров, позволяющих выполнять за 1 такт несколько операций; а также специальные коммуникативные средства, дающие, в отличие от стандартных средств, гарантированную полосу пропускания для каждого подсоединенного к ним устройства. Такие специальные средства производят фирмы Cray research, IBM, Compaq, NEC и некоторые другие.

Логично сделать вывод, что основной задачей при создании суперкомпьютеров было «производительность любой ценой». Специальные процессоры, дорогостоящая сверхбыстрая память, нестандартное периферийное оборудование - всё это приводит к увеличению стоимости таких систем в десятки и сотни раз. Естественно, позволить себе купить суперкомпьютер может далеко не каждое предприятие, которое занимается деятельностью, требующей интенсивных расчетов.

Но на сегодняшний день уже есть выход из этой ситуации.

Из-за быстрого роста производительности персональных компьютеров и развития интрасетей, стало возможным создание высокопроизводительной вычислительной системы при помощи объединения в

сеть существующих персональных компьютеров. Получаемая при этом вычислительная система - кластер - обладает многими преимуществами, главными из которых являются дешевизна, доступность, возможность постепенного расширения и модернизации.

Однако, при создании кластерной системы нужно сделать выбор: либо система обеспечивает повышенную надежность вычислений, любо дает увеличение производительности. Эти термины обычно исключают друг друга, т.е. повышение надежности всегда сопровождается увеличением кода программы, большими требованиями к памяти, вследствие дублирования информации или хранения параметров для проверки корректности данных, дополнительными проверками и т.п. Все это, естественно, снижает быстродействие системы. Наоборот, если считать оборудование абсолютно надежным, то можно не учитывать возможность его отказа и работать без дополнительных проверок, кодирования и дублирования информации. Таким образом, некую абстрактную кластерную систему можно настроить либо на большее быстродействие, либо на большую надежность.

Многие существующие системы распределенных вычислений изначально настроены либо на одно, либо на другое.

Так, узким местом Beowulf-кластеров (Bèowulf - технология организации параллельных вычислений на Linùx-кластерах) является головная машина-сервер. На ней хранится информация о структуре кластера и с нее осуществляется запуск параллельных программ, поэтому кластер не сможет работать в случае ее отказа.

Популярная в настоящее время технология параллельного; программирования MPI (Message Passing Interface – Интерфейс передачи-сообщений) позволяет организовать распределенные вычисления в многомашинном комплексе на основе передачи сообщений. Данная технология: (в частности ее свободно-распространяемая версия MPICH (MPI CHameleon)) настроена на оптимальное быстродействие - в ней нет низкоуровневых средств слежения за отказами узлов, и неполадки с одним из них приводят к краху системы в целом и необходимости начинать вычисления с начала.

Кроме того, в вышеупомянутых системах практически не имеется средств смены конфигурации кластера во время вычислительного процесса, и они слабо ориентированы на гетерогенную сеть, состоящую из множества вычислительных узлов самых разных конфигураций - от персонального компьютера с процессором i486 до многопроцессорной стойки с процессорами Alpha. При написании MPI-программ для таких систем приходится прибегать к разного рода ухищрениям, учитывая разброс скоростных и архитектурных характеристик вычислительных узлов.

Этих недостатков не имеют системы другого класса – метакомпьютеры. Это распределенные вычислительные системы, в которых нет постоянного соединения между вычислительными узлами, и которые могут динамически менять свою конфигурацию. Обычно в таких системах вычислительному узлу передаются данные для расчета, и он «отключается» от системы и решает свою задачу. После того, как данные готовы, он подключается к головной машине, возвращает результат и берет следующую порцию данных. Налицо преимущества такой системы: возможность оптимального размещения задачи по вычислительным узлам разной архитектуры и различной мощности, возможность динамического подключения и отключения произвольного количества вычислительных узлов. Недостаток метакомпьютерной системы - ориентированность на задачи; переборного и поискового типа, где вычислительные узлы не взаимодействуют друг с другом.

В настоящее время во всем мире идет активная работа по> совершенствованию теоретических и практических основ функционирования кластерных и метакомпьютерных систем, созданию оптимальных методов размещения задач в распределенной вычислительной системе и механизмов планирования вычислений и оптимизации вычислительного процесса.

Разработано множество средств планирования вычислительного процесса и управления заданиями в распределенных вычислительных системах, но ни одно из них, естественно, не является универсальным. Кроме того, основная масса исследований проводится в области глобальных метакомпьютерных систем с гигантским множеством пользователей и огромным количеством вычислительных узлов. При этом упор делается на эффективное обслуживание потока заданий от множества, пользователей. И очень мало разработано методик анализа вычислительного процесса в кластерно-метакомпьютерных системах, которые не являются столь глобальными, как метакомпьютеры и столь простыми, как кластеры, и соответственно, требуют к себе особого подхода.

Для управления заданиями в распределенной вычислительной среде предлагается идея метадиспетчера (или грид-диспетчера), который планирует размещение заданий в группах вычислительных узловметакомпьютера в соответствии с требованиями каждого задания к ресурсам и с имеющимися ресурсами в узлах, при этом используется распределенная база данных, имеющая информацию о ресурсах. При этом, так как предполагается, что метадиспетчер имеет дело с большим количеством разнородных заданий,

приходящих с разных направлений, большее внимание уделено определению очередности запуска и предсказанию моментов старта заданий в узлах, когда выполнение будет наиболее эффективно.

Такой подход не совсем обоснован для кластерно-метакомпьютерных систем, в которых множественный гетерогенный поток заявок как таковой отсутствует из-за централизованной структуры управления системой, но присутствует задача, которую необходимо распределить между вычислительными узлами таким образом, чтобы обеспечить минимальное время расчета. При этом должна учитываться разная производительность узлов и каналов связи между узлами и сервером (т.е. гетерогенность системы).

В некоторых работах предлагается решение задачи эффективного распределения процессов в многомашинном вычислительном комплексе путем решения оптимизационной задачи методом ветвей и границ. При этом учитываются пропускные способности каналов связи. Полученное улучшение за счет перераспределения процессов достигает 12%. Однако, недостатком такого подхода является необходимость иметь историю вычислений программы для каждого из процессов. Кроме того, в этих исследованиях не учитываются характеристики надежности узлов и каналов связи.

В других работах рассматриваются способы повышения эффективности параллельных и распределенных вычислений за счет переработки алгоритма параллельной программы с использованием моделей недетерминированных цифровых автоматов и теории марковских процессов. При этом рассматривается возможность анализа различных вариантов реализации алгоритма задачи по трудоемкости и выбора наиболее эффективного из них. Также предложен способ организации вычислительной системы, позволяющий осуществлять динамическую балансировку загрузки отдельных ее узлов за счет разделения программы на заведомо большое число ветвей и асинхронного запуска параллельных процессов.

Однако, данный подход не учитывает пропускные способности каналов связи, что может снизить эффективность предложенных методов. Кроме того, не учитываются надёжностные характеристики элементов системы.

Кластерные системы

Часто выделяют три технологии обеспечения параллельной работы: симметричные многопроцессорные системы (SMP - symmetrical multiprocessing), кластерные конфигурации и распределенные вычислительные системы (Grid). SMP требует поддержки как со стороны аппаратуры, так и со стороны операционной системы, а кластеры и Grid-среды больше зависят от организации сетевого взаимодействия.

С точки зрения ядра операционной системы поддержка кластеров и распределенных систем заключается в эффективной работе с сетью. С некоторым упрощением любую современную высокопроизводительную вычислительную систему можно представить как множество многопроцессорных вычислительных узлов, связанных одной или несколькими коммуникационными сетями. Важная общая характеристика таких систем - логическая организация оперативной памяти, с которой работают вычислительные узлы. Оперативная память может быть: разделяемой для всех узлов; распределенной - доступной только для процессоров своего узла; распределенной разделяемой - доступной для процессоров своего узла и из других узлов, но с применением специальных программно-аппаратных средств.

Важные преимущества кластеров - доступность технологий сборки и возможность экономически эффективного получения достаточно высокой производительности. Но насколько высока их производительность по отношению к тем системам, которые принято называть суперкомпьютерами? Сегодня наивысший уровень производительности суперкомпьютеров измеряется десятками терафлоп, достигаемых на многопроцессорных векторно-конвейерных системах (NEC Earth Simulator System на базе SX-6 и Cray XI), массово-параллельных системах, в том числе, и системах кластерного типа от HP, IBM, Intel и Cray. Кластерные системы с числом процессоров в несколько тысяч и производительностью около 1 TFLOPS можно со всем основанием считать суперкомпьютером. Кластерные системы с производительностью от нескольких десятков до нескольких сотен GFLOPS будем называть просто высокопроизводительными системами. Именно этот класс машин широко используется в современных высокотехнологичных отраслях промышленности, а также в социально-экономической сфере и представляет наибольший практический интерес.

Усилиями НИИ «Квант», ИПМ РАН и Межведомственного суперкомпьютерного центра разработан кластер класса МВС-1000М, содержащий 768 микропроцессоров Alpha 21264, объединенных коммуникационной сетью Myrinet 2000. Сегодня МВС-1000М входит в список Top 500 самых мощных компьютеров в мире. Другая реализация - типовая кластерная система ОАО НИЦЭВТ (серия ЕС 1720), структура 12-процессорного варианта которой представлена на рис. 1.

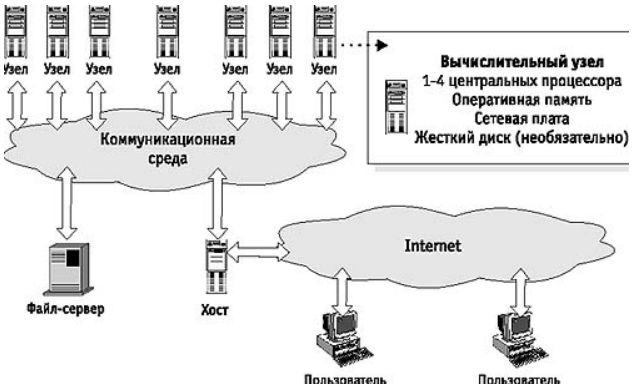

Рис. 1. Типовая кластерная система

Это система среднего класса производительности, которая является типовой для промышленных применений и ориентирована на серийный выпуск. Ее вычислительные узлы - стандартные серверные платы с процессорами Pentium 4 Xeon, оперативной памятью до 2 Гбайт, локальными дисками и сетевыми платами Fast Ethernet, Gigabit Ethernet и основной коммуникационной сетью для передачи данных SCI (Scalable Coherent Interface, стандарт IEEE 1596). Применяются аппаратные сетевые средства компании норвежской Dolphin и системное программное обеспечение от Scali. На опытном производстве НИЦЭВТ освоен выпуск отечественных адаптеров SCI, проведены работы по импортозамещению системных программных средств.

Топологию коммуникационной сети SCI могут образовывать двухмерный тор, трехмерный тор, либо коммутируемые через центральный коммутатор SCI-кольца вычислительных узлов. Топология «трехмерный тор» позволяет строить системы с количеством узлов более 64; имеются системы такого типа с несколькими сотнями узлов, обладающие терафлопным уровнем производительности. Топология соединяемых через коммутатор SCI-колец позволяет добиться повышенной производительности обменов типа «все-всем», что важно, например, для решения задач имитационного моделирования.

Коммуникационная сеть SCI позволяет передавать данные со скоростью до 300 Мбайт/с на уровне пользовательской программы при использовании Pentium 4 Xeon/2,4 ГГц и до 390 Мбайт/с для Itanium 2. Уникальной характеристикой SCI являются малые значения задержек передачи сообщений, например, задержка передачи сообщения нулевой длины занимает всего 3,5 мкс. Рекордно мало и время групповой барьерной синхронизации - несколько микросекунд, причем с увеличением числа узлов это время увеличивается незначительно. Таким образом, сеть SCI является не только высокоскоростной, но и высокореактивной, что важно на практике при решении ряда задач, а также позволяет реализовать перспективные модели вычислений с высокой грануляцией параллельных процессов и большой асинхронностью. Из других решений такими свойствами обладает лишь значительно более дорогая сеть Quadrics, используемая в некоторых высших моделях кластерных ВМВС американской программы стратегической компьютерной инициативы ASCI.

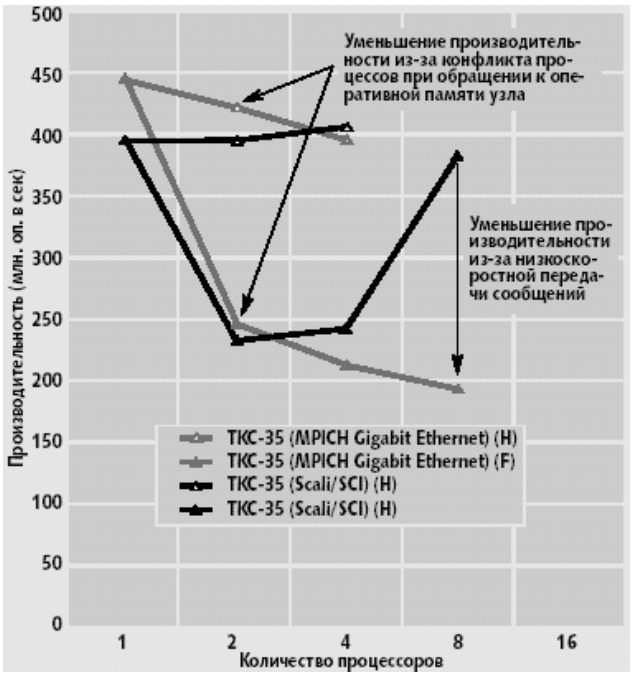

Рис. 2. Типовые характеристики производительности на один процессор

На рис. 2 приведены экспериментально полученные на кластере ТКС-35 оценки реальной производительности одного процессора в зависимости от числа процессоров, применяемых при параллельном счете. Результаты получены на задаче LU (класса сложности A) пакета NASA NPB 2.3. Это задача решения уравнения Навье-Стокса; для используемого метода решения характерно наличие большого количества передач коротких сообщений. Приводятся данные для сетей SCI и Gigabit Ethernet при условии использования одних и тех же вычислительных узлов с двумя процессорами Pentium 4 Xeon.2,4 ГГц и 400-мегагерцевой системной шиной. Характеристики с индексом H соответствуют режиму использования при решении одного процессора вычислительного узла, а с индексом F - двух процессоров.

Более высокое качество сети SCI сказывается при использовании уже двух процессоров, а особенно заметно при 8 процессорах. Резкий рост реальной производительности при 8 процессорах, зависимость ТКС-35 (Scali/SCI) (F), объясняется тем, что задача при распараллеливании разбилась так, что уже хорошо

локализуется в кэше, и скорость вычислений резко увеличилась, однако высокое качество сети позволило не потерять преимущества за счет роста скорости вычислений. Gigabit Ethernet этого не позволяет: для этой сети пропускная способность на уровне пользователя составляет около 50 Мбайт/с, а задержка передачи сообщения нулевой длины - около 46 мкс.

Различие реальной производительности для режимов H и F объясняется межпроцессорными помехами внутри платы вычислительного узла при доступе к общей памяти. Это досадное явление присуще Pentium 4 Xeon, который сегодня является стандартом де-факто для кластерных ВМВС в варианте двухпроцессорных серверных плат. Для AMD Opteron и Itanium 2 этого не наблюдается; вместе с тем, пока эти процессоры в существующих реализациях показывают обычно меньшую производительность, чем Pentium 4 Xeon, и их применение представляется преждевременным.

Практическое применение отечественных кластерных ВМВС связано с выполнением на них популярных наукоемких задач, например, инженерных расчетов.

Деятельность по созданию перспективных отечественных ВМВС с повышенной эффективностью на широком классе задач была начата в 2002 году.

Исследования ведутся по следующим направлениям:

- выделение типовых проблем, возникающих при решении задач на современных вычислительных системах и обуславливающие их низкую реальную производительность, поиски путей их решения за счет использования мультитредовых (multithread) архитектур и соответствующих программ;

- разработка принципов организации мультитредовых процессоров с разной организацией (архитектура с управлением потоком данных - DF; мультитредовая архитектура - MT; параллельная мультитредовая архитектура - SMT, мультитредовая архитектура с управлением потоком данных - MT/DF или SMT/DF; чипмультитредовая архитектура - CMP; архитектура процессоров внутри кристалла памяти - PIM);

- разработка принципов организации коммуникационных сред с высокой пропускной способностью и малой задержкой передачи сообщений для мультитредовых вычислительных систем;

- разработка принципов организации мультитредовых систем с распределенной разделяемой памятью и динамической балансировкой загрузки процессоров;

- разработка принципов организации компиляторов языков программирования для мультитредовых процессоров и систем, обеспечивающих статическое и динамическое автоматическое распараллеливание;

- разработка принципов организации исполняемых мультитредовых программ;

- разработка принципов построения систем предобработки больших объемов сигнальной информации в реальном времени.

Остановимся на системах типов DF и SMT.

DF-система разработана на базе ОСВМ, но является менее «радикальной» и приближенной к современным системам с мультитредовой архитектурой. В значительной степени на это решение повлияла архитектура системы TeraMTA, а также очень близкие по тематике исследования работы Стэнфордского университета и Массачусетского технологического института.

В основе DF-системы лежит классический мультитредовый RISC-микропроцессор фон-неймановской архитектуры, обычным образом работающий с оперативной памятью, но имеющий одну особенность, связанную с поддержкой вычислений, управляемых потоком данных. Особенность эта состоит в том, что в состав блоков введен особый блок приема/выдачи DF-сообщений, который может принимать исполняемые пары, а также выдавать формируемые в токены. Простые операции узлов потокового графа программы при помощи соответствующих исполняемых пар передаются этим блоком непосредственно в функциональные устройства. После их выполнения функциональные устройства порождают и передают обратно в блок приема/выдачи DF-сообщений заявки на порождение токенов с результатами этих операций.

Инициализация вычислений, соответствующих сложным узлам DF-графа, производится также блоком приема/выдачи DF-сообщений. После поступления в него исполняемой пары этот блок порождает тред микропроцессора, который далее выполняется так, как если бы был явным образом порожден обычным средством типа fork. Эта возможность значительно повышает динамику порождения тредов. Существенно и то, что пользователь освобожден от порождения явным образом: они порождаются автоматически в соответствии с графом потока данных программы.

Если рассматривать работу на уровне микроархитектуры, то оказывается, что он фактически имеет два блока выборки команд для загрузки своих функциональных устройств. Один из них - обычный блок выборки команд запущенных тредов основного (фон-неймановского) набора команд, а другой - это блок приема/выдачи DF-сообщений. Он также передает исполняемые команды в функциональные устройства, но эти команды происходят из вычислений, управляемых потоком данных. В отличие от исполнительного устройства ОСВМ, микропроцессор выполняет сразу вычисления по множеству вычислительных узлов DF-графа программы, причем как простых, так и сложных. При

этом одновременно еще выполняются обычные программы, работающие с оперативной памятью, но в режиме мультитредовости.

Еще одна особенность предложенной DF-системы - резко усиленная по сравнению с OCBM по своим функциональным возможностям часть, связанная с реализацией потокового управления - спаривания токенов, приходящих в вычислительные узлы DF-графа программы (DF-часть). В OCBM эти функции выполняли модули ассоциативной памяти. В предложенной DF-системе вместо модуля ассоциативной памяти введен мультитредовый микропроцессор доступа к данным (процессор dp). Этот процессор работает с банками ассоциативной и локальной оперативной памяти, однако при этом выполняет значительно больший репертуар операций. Собственно, эти узлы были фактически введены и в OCBM, однако в DF-системе они значительно разнообразнее.

Повышение сложности работы с узлами в DF-части привело к тому, что обработка поступающих в нее токенов стала достаточно сложным и многошаговым процессом, для которого характерна интенсивная работа, как с ассоциативной, так и с локальной оперативной памятью. Проблема обеспечения интенсивной работы с памятью в dp решается стандартным для современных микроархитектур способом - введением мультитредовости. Каждый тред автоматически порождается по пришедшему в dp токену. Таким образом, dp может одновременно выполнять сложную обработку множества токенов.

Предложенная DF-система обладает свойствами современных мультитредовых систем, однако включает элементы потокового управления и использования ассоциативного доступа к данным. Теоретические оценки DF-системы, сделанные пока на простых примерах типа DAXPY с использованием реальных для реализации временных диаграмм работы устройств, показывают, что она может развивать сопоставимую с векторными машинами (Cray SV1, SV2 или Cray XI) скорость, причем даже над нерегулярно расположенными в памяти данными, что крайне важно для современных приложений.

Если проводить аналогию с современными системами, то предложенную DF-систему можно рассматривать как многопроцессорную мультитредовую, в которой применяется кэш-память с большим расслоением и возможностями сложной выборки данных. Такая сложная подсистема кэш-памяти - это и есть DF-часть.

Описанная DF-система послужила базой для разработки системы SMT-типа. Микропроцессор SMT-системы отличается лишь тем, что вместо блока приема/выдачи DF-сообщений в нем введен кэш команд и данных второго уровня, добавлен кэш данных первого уровня, а в блоке выборки команд введена возможность выборки команд на такте не из одного треда, а из нескольких. В SMT-системы также усилены возможности спекулятивного выполнения команд.

Выводы

В статье был проведен теоретический анализ систем распределенных вычислений и протекающих в них процессов с целью рассмотрения возможных способов повышения их эффективности за счет оптимального размещения узлов вычислительной системы в реальной сети, применения стратегий размещения подзадач в узлах вычислительной системы, имеющих определенные характеристики производительности и надежности. Разработана стратегия размещения узлов распределенной вычислительной системы, которая позволяет повысить общую эффективность вычислений за счет учета характеристик производительности и надежности узлов, входящих в систему. Разработана классификация стратегий размещения подзадач в распределенных вычислительных системах, которая позволяет выделить, основные параметры, достоинства и недостатки каждой стратегии и составляющих ее методов, а также области их применимости, что в конечном итоге позволит выбрать правильный метод размещения для определенных решаемых задач. Разработана универсальная расчетно-обменная характеристика задач, решаемых в системе распределенных вычислений. Разработана статическая стратегия размещения подзадач в системе распределенных вычислений, которая учитывает характеристики производительности и надежности узлов и каналов связи. Разработана динамическая адаптивная стратегия размещения подзадач в системе распределенных вычислений, которая учитывают изменения характеристик системы и перераспределяет объемы выдаваемых узлам подзадач во время вычислительного процесса. Проведена проверка полученных теоретических результатов на реальной вычислительной системе «Mathnet» как для расчетно-ориентированных задач (вычисление экстремумов функций и численное интегрирование), так для обменно-ориентированных задач (умножение матриц), которая доказала эффективность предложенных решений.

Литература

1. Мирошник М.А. Подход к проектированию компьютерных систем с интеллектуальной диагностической инфраструктурой. / С.Г. Карпенко, М.А. Ковалева, С.В. Панченко// Інформаційно – керуючі системи на залізничному транспорті. – 2011. – №6. – С. 51-59.

2. Мирошник М.А. Отказоустойчивость распределенных телекоммуникационных систем. / М.А. Мирошник, В.Г. Котух, С.Н. Селевко // РАДІОТЕХНІКА: Всеукр. межвед. науч.-техн. сб. 2011. Вып. 168. С. 51–55.
3. Мирошник М.А. Отказоустойчивые вычислительные системы с реконфигурируемой структурой / М.А. Мирошник, Г.И. Загарий // Третя міжнародна науково-практична конференція «Методи та засоби кодування, захисту й ущільнення інформації». Вінниця. 29-31 трапвня 2012р. – С. 24.
4. Патент на винахід № 98395 (51)МПК G01F 11/28 (2006.01), G01R 35/00 - Пристрій для функціонального діагностування пристрою регулювання росту монокристалів, заеєстровано 11.05.2012.
5. Мирошник М.А. Решение задач диспетчеризации в распределенных телекоммуникационных системах. / М.А. Мирошник, В.Г. Котух, С.Н. Селевко // РАДІОТЕХНІКА: Всеукр. межвед. науч.-техн. сб. 2011. Вып. 169. С. 139–152.
6. Miroshnik M. Design of a Built-in Diagnostic Infrastructure for Fault-Tolerant Telecommunication Systems. / Miroshnik M., Miroshnik N., Panchenko S. // Інформаційна – керуючі системи на залізничному транспорті. – 2012. – №4 (приложение. (25 Міжнародна конф. «Перспективні комп'ютерні управляючі и телекомунікаційні системи»). – 59 с.
7. Мирошник М.А. Метод проектирования строго безопасных автоматов локомотивной сигнализации. / Малиновский М.Л., Мирошник М.А.// Інформаційно – керуючі системи на залізничному транспорті. – 2012. – №5 С. 25-42.
8. Miroshnik M. Application of software complex for query processing in the database management system with a view of dispatching problem solving in Grid systems. / Miroshnik M. Kotukh V.G., Selevko S.N. // Telecommunications and radio engineering. – 2013. Vol.27, № 10. – p 875-891.
9. Мирошник М.А. Синтез распределенных компьютерных сред на базе компьютерных сетей. / Систем и обробки інформації. – 2013 – №7 (114). 4 стр
10. Мирошник М.А. Разработка средств организации функционирования распределённых вычислительных систем и сетей. / М.А. Мирошник, Н.В. Мирошник // Інформаційно-керуючі системи на залізничному транспорті. – 2013. – №4. (Додаток. 26-а Міжнародна конф. «Перспективні комп'ютерні управляючі и телекомунікаційні системи») - 25с.

Мірошник М.А., Клименко Л.А. РОЗМІЩЕННЯ ПОДЗАДАЧ В РОЗПОДІЛЕНИХ ОБЧИСЛЮВАЛЬНИХ СИСТЕМАХ КЛАСТЕРНО-МЕТАКОМПЬЮТЕРНОГО ТИПУ. У статті проведено класифікацію стратегій розміщення подзадач в розподілених обчислювальних системах і розроблена стратегія розміщення вузлів розподіленої обчислювальної системи, універсальна розрахунково-обмінна характеристика завдань, що вирішуються в системі розподілених обчислень і статична стратегія розміщення подзадач в системі розподілених обчислень. Проведено перевірку отриманих теоретичних результатів на реальній обчислювальної системі як для розрахунково-орієнтованих завдань (обчислення екстремумів функцій і чисельне інтегрування), так для обмінно-орієнтованих завдань (множення матриць), яка довела ефективність запропонованих рішень.
Ключові слова: розподілена обчислювальна систем, кластерна система, метакомпьютерний тип.

Miroshnik M.A., Klymenko L.A. ACCOMMODATION SUBTASKS IN DISTRIBUTED COMPUTING SYSTEMS, CLUSTER-TYPE METACOMPUTING. We develop the strategy of deploying nodes distributed computing system ; classification subtasks placement strategies in distributed computing systems; universal cash- exchange characteristics of problems solved in a system of distributed computing ; static placement strategy subtasks in a distributed computing. Audited the theoretical results obtained on a real computer system for cash- oriented tasks (computation of extrema of functions and numerical integration), so for the exchange- oriented tasks (matrix multiplication), which demonstrated the effectiveness of the proposed solutions.
Key words: distributed computing systems, cluster system, metacomputing type.

Рецензент д.т.н., профессор Листровой С.В. (УкрГАЖТ)

Поступила 22.06.2014г.