УДК 004.032.24

РАЗРАБОТКА АЛГОРИТМОВ УПРАВЛЕНИЯ ЗАДАНИЯМИ ПРИ ОРГАНИЗАЦИИ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ В КЛАСТЕРНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ*

А.А. АЛЬ-ХУЛАЙДИ, Ю.О. ЧЕРНЫШЁВ
(Донской государственный технический университет)

Рассмотрена задача оптимизации распределения заданий по исполнителям в кластере. Разработаны алгоритмы, осуществляющие управление задачами. Проверка предложенных алгоритмов выполнена при решении транспортной задачи на кластере локальной сети. Получены данные, позволяющие судить об эффективности предложенных алгоритмов.
Ключевые слова: *кластеризация, планировщик задач, менеджер ресурсов.*

Введение. В последние годы активно ведутся работы, связанные с использованием технологии параллельной обработки, которая позволяет значительно повысить скорость выполнения работы. Поскольку суперкомпьютеры остаются достаточно дорогими, то только большие компании способны обладать такими вычислительными возможностями [1]. Решением проблемы является кластеризация законченных процессорных единиц воедино. Таким образом, существует возможность создать систему, вычислительная мощность которой будет сравнима с суперкомпьютером, а стоимость значительно меньше.

Постановка задачи. При параллельных вычислениях в кластерном пакете MPI/MPICH [2] есть одна проблема, а именно, отсутствие организации управления заданиями. Была поставлена задача разработать расширение для пакета MPICH, позволяющее управлять заданиями в кластере локальной сети. Для этого были разработаны и реализованы алгоритмы управления заданиями. Предложенные алгоритмы проверены на выбранном классе транспортных задач. Для этого разработан параллельный алгоритм нахождения опорного плана транспортной задачи на основе метода Фогеля (штраф) [3].

Модель транспортной задачи. Транспортная задача является специальным типом задач линейного программирования. Постановка этой задачи следующая. Имеется m поставщиков и n потребителей некоторой продукции. Заданы тарифы (стоимость) перевозок единицы продукции от поставщиков к потребителям, известны объемы запасов у поставщиков и потребности каждого потребителя в продукции [4].

Требуется составить план поставок продукции от поставщиков к потребителям так, чтобы суммарная стоимость перевозок была минимальной.

Математическая постановка этой задачи имеет вид:

$$\min \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} X_{ij};$$

$$\sum_{i=1}^{m} X_{ij} = b_j, \quad j = 1,2,...,n; \qquad (1)$$

$$\sum_{j=1}^{n} X_{ij} = a_i, \quad i = 1,2,...,m;$$

$$x_{ij} \geq 0,\ i = 1,2,...,m\ ,\ j = 1,2,...,n.$$

* Работа выполнена при поддержке Российского фонда фундаментальных исследований (грант № 10-01-00481-а, г/б №1.21.11).

Здесь X_{ij} – объем; c_{ij} – тариф поставки продукции от i-го поставщика к j-му потребителю; b_j – потребности потребителей в продукции; a_i – запасы продукции у поставщиков.

Модель (1) является задачей линейного программирования со специальной матрицей. В этой задаче имеется $m \times n$ неизвестных X_{ij} и $(m+n)$ уравнений.

Решение транспортной задачи называется оптимальным планом перевозок (поставок) продукции.

Алгоритмы управления очередями заданий.

Алгоритм распределения ресурсов между процессами. Задание представляет собой абстрактную сущность, состоящую из набора команд и параметров. Оно представляется пользователю в виде скрипта, содержащего требования к ресурсам, атрибутов задания и набора команд, которые необходимо выполнить. Единожды создав скрипт задания, им можно пользоваться столько раз, сколько необходимо, также возможна его модификация. Задание сначала необходимо поставить в очередь планировщика, затем из этой очереди оно будет передано на один из узлов для выполнения.

Задание может быть обычным (regular) и интерактивным (interactive). Обычное задание ставится в очередь и затем ожидает своего выполнения, результат будет записан в указанное пользователем место. Интерактивное задание отличается тем, что потоки ввода и вывода перенаправляются соответственно на экран и клавиатуру. Команды задания вводятся непосредственно с клавиатуры.

В каждом задании для запуска запрашивается множество разных ресурсов, таких, как процессоры, память, время (обычное и процессорное). Может понадобиться дисковое пространство. С помощью менеджера ресурсов можно установить лимит каждого ресурса. Если лимит не устанавливается для какого-либо ресурса, то он считается равным бесконечности.

Алгоритм состоит из следующих процедур.

1. Определяются типы ресурсов, используемых данным заданием, их количество, а также приоритет задачи.

2. Задания распределяются по очередям (подгруппам) и упорядочиваются в порядке приоритета.

3. Службе управления очередью отсылается задание в исполняемый скрипт. Привязка к существующему резервированию осуществляется по ACL, т. е. резервирование ресурсов к этому моменту должно быть уже выполнено. Ресурсы могут быть следующих видов: количество процессоров, объем памяти, требуемое программное обеспечение, объем виртуальной памяти, количество времени и др.

Алгоритм управления запуском заданий для кластерных систем. Планировщик работает итерационно, т.е. перемежая процесс планирования с ожиданием или выполнением внешних команд [5]. Каждый цикл начинается при осуществлении одного из следующих событий:

– меняется состояние задания или ресурса;

– достигнута граница резервирования;

– получена внешняя команда;

– с начала предыдущего цикла прошло время, определенное как максимальное.

После того как сделано некоторое, конфигурируемое количество резервирований, начинает работу алгоритм обратного заполнения (backfill) [6]. Он выясняет, какие узлы и на какое время, начиная с текущего, свободны. После этого свободные узлы объединяются в окна. Суммарная «ширина» окон может оказаться больше количества свободных в данный момент узлов, так как некоторые узлы могут входить в несколько окон. Затем из всех окон выбирается одно, как правило, самое широкое. Среди всех оставшихся заданий выбирается наиболее точно удовлетво-

ряющее этому «окну» задание и запускается. Если есть возможность, то запускается не одно задание. Так как при запуске заданий алгоритмом backfill учитываются сделанные ранее резервирования, то соответствующие им задания не будут задержаны.

Рассмотрим теперь один цикл работы разработаного алгоритма. Он состоит из следующих шагов.

1. Получая сведения от системы пакетной обработки заданий (СПО) о машинах, стартовавших/закончившихся заданиях, конфигурации системы, планировщик обновляет свою внутреннюю информацию о состоянии ресурсов.

2. Корректируются сделанные резервирования исходя из обновленной информации о занятости узлов. Конкретнее – отменяются ранние резервирования, соответствующие уже завершившимся заданиям. На этой же фазе запускаются те ожидающие задания, время действия для которых уже наступило.

3. Далее, из всего количества заданий, стоящих в очереди СПО, отбираются те, которые готовы к запуску в настоящий момент. При этом принимаются во внимание такие аспекты, как состояние задания (задержано оно или нет), достаточно ли ресурсов всего кластера для запуска этого задания и т. п. После этого, из получившегося списка, отсеиваются задания в соответствии с политикой кластера.

4. Получившееся множество заданий упорядочивается в соответствии с приоритетами, которые назначаются исходя из ряда условий. В частности, принимаются во внимание запрашиваемые ресурсы, принадлежность задания тому или иному пользователю, историческая информация (кто, сколько заданий запустил, чтобы избежать ситуации, когда кластер «оккупирован» кем-то и недоступен для всех остальных).

5. В соответствии с полученным упорядоченным списком производится планирование. Задания рассматриваются, начиная с самого приоритетного. Последовательно получая задания из списка, планировщик запускает их до тех пор, пока это возможно.

Параллельный алгоритм нахождения опорного плана транспортной задачи на основе метода Фогеля (штраф). Пусть u – вектор потребностей потребителей, v – вектор мощностей поставщиков, z – матрица стоимости перевозок, x – матрица решения (поставок).

Основные шаги алгоритма следующие.

1. Для каждой строки и столбца матрицы стоимости перевозок считаем разницу между наименьшим значением стоимости и ближайшим к нему (так называемый *штраф*). Поиск штрафа по каждой строке и столбцу оформляется в виде отдельной задачи и назначается свободным процессам, т.е. распараллеливается.

2. Во всех строках и столбцах с наибольшим штрафом находим ячейки с наименьшим минимальным значением стоимости (не зачеркнутые ранее). Выполнение этого шага также распределяется между свободными процессами.

3. Максимизируем поставку в ячейку $x\,[i, j]$, выбрав минимальное значение из потребности потребителя $u\,[i]$ и мощности поставщика $v\,[j]$.

4. Если мощность поставщика полностью реализована или потребность потребителя полностью удовлетворена, вычеркиваем соответствующую строку или столбец.

5. Если не все потребности/мощности задействованы, повторяем алгоритм. Параллельная модификация алгоритма использует тот факт, что подсчет штрафов и поиск минимального элемента в пунктах 1, 2 соответственно поддаются параллелизации за счет разделения задач по поиску в разных строках и столбцах между несколькими потоками выполнения. Блок-схема параллельного алгоритма нахождения опорного плана решения транспортной задачи представлена на рис.1.

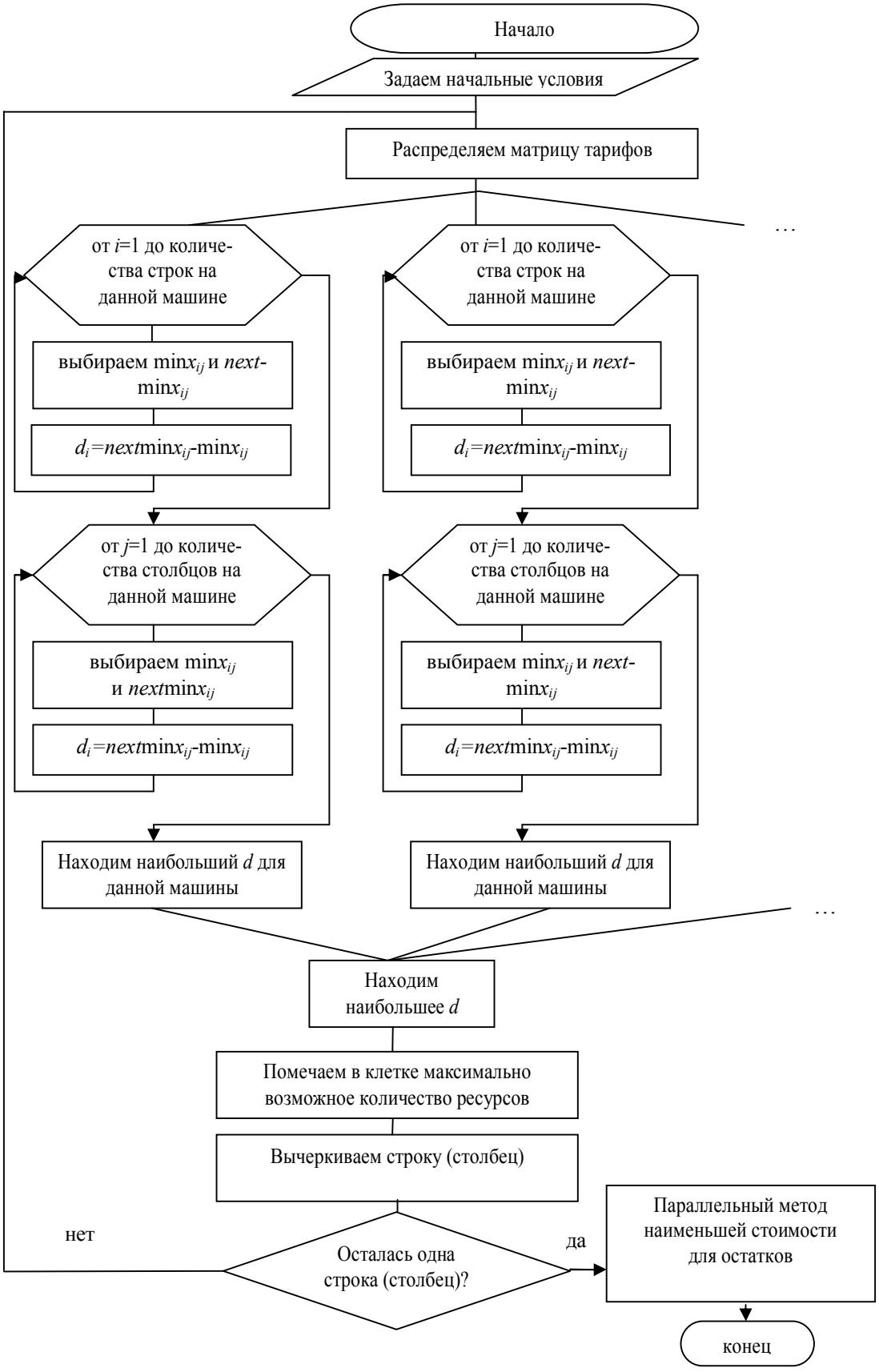

Рис.1. Блок-схема параллельного алгоритма нахождения опорного плана методом Фогеля

Экспериментальная проверка эффективности предложенных алгоритмов управления очередями заданий в кластерных системах. Практическая значимость работы заключается в том, что алгоритмы управления заданиями были применены для расширения библиотеки MPI/MPICH (по свободным лицензиям), т.е. в создании программной модели кластера, который содержит реализации предложенных алгоритмов. С использованием новой библиотеки MPI/MPICH_NEW была проверена эффективность параллельной программы нахождения опорного плана транспортной задачи на основе метода Фогеля (штраф) для кластера локальной сети.

Разработанный параллельный алгоритм проверялся путем реализации его на кластере следующей конфигурации:

– 4 вычислительных узла (Intel Pentium 4 2,4 ГГц);

– управляющий узел (Intel Pentium 4 2,4 ГГц).

Узлы объединены между собой сетью Infiniband (пропускная способность 4 Гбит/с).

Описанная программа реализована в среде C++. Обозначим время выполнения параллельного алгоритма на кластерном пакете MPI/MPICH – $T_{MPI/MPICH}$, а время выполнения параллельного алгоритма на улучшенном пакете MPI/MPICH_NEW – $T_{MPI/MPICH\_NEW}$. Производительность (ускорение) определялась по формуле:

$$У = \frac{T_{MPI/MPICH}}{T_{MPI/MPICH\_NEW}}.$$

Структурная схема пакета MPI/MPICH_NEW приведена на рис.2.

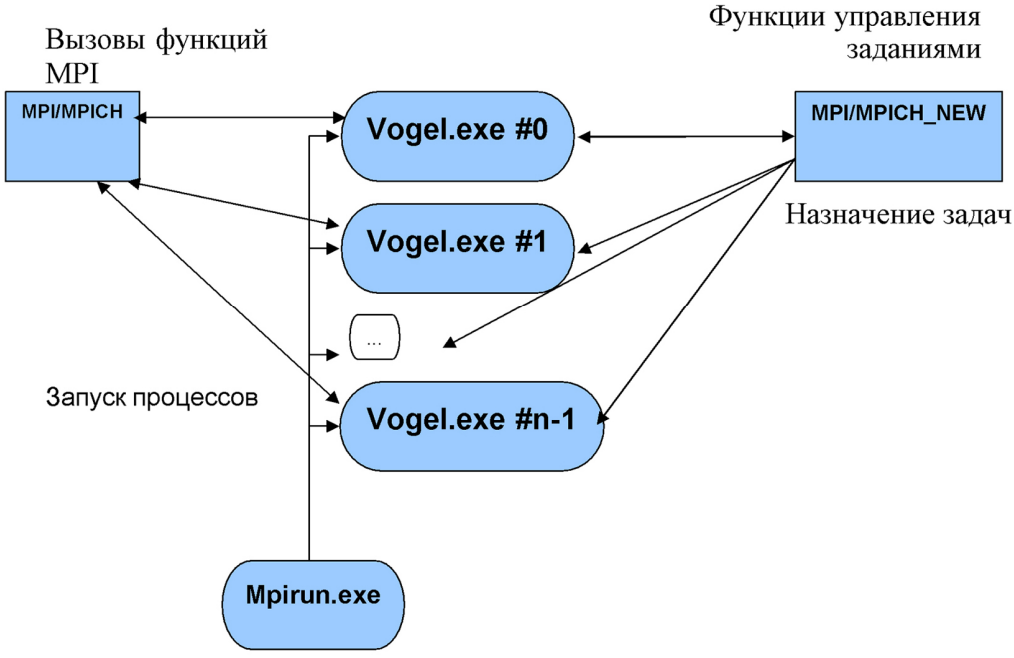

Рис.2. Структурная схема пакета MPI/MPICH_NEW

Библиотека MPI/MPICH_NEW содержит функции для управления заданиями задач и вычисляет назначение задач процессам. Она служит для запуска процессов на машинах кластера, для коммуникации между процессами, а также для вычислений в разделяемой памяти. Утилита

Mpirun.exe входит в реализацию MPI/MPICH и служит для запуска процессов,Vogel.exe – параллельные программы.

Испытание предложенных алгоритмов управления очередями заданий на транспортной задаче. Для оценки практической значимости разработанных алгоритмов проведено испытание и сравнительная характеристика библиотеки MPI/MPICH и нового улучшенного пакета MPI/MPICH_NEW, который содержит разработанные алгоритмы управления заданиями. Испытания проводили в три этапа. На первом этапе при размерности задачи 50×50, 70×70 и числе процессов 20 изменение производительности не происходило. На втором этапе при размерности задачи 100×100, 150×150, 500×500 и числе процессов 20 увеличение производительности стало более заметным (в 1,07 раз быстрее при расчете матрицы размера 500×500). На третьем этапе при увеличении числа процессов с 20 до 40 увеличение производительности стало более заметным (в 1,13 раз быстрее при расчете матрицы размера 1000×1000). Результаты вычислительных экспериментов, исследования предложенных алгоритмов управления заданиями, использованных в кластерном пакете MPI/MPICH_NEW, и сравнение их с пакетом MPI/MPICH приведены в таблице и на рис.3.

Сравнительная характеристика времени нахождения опорного плана параллельным алгоритмом на MPI/MPICH и MPI/MPICH_NEW в зависимости от размерности задачи и количества процессов

Размерность задачи	Количество процессов	Время выполнения параллельного алгоритма нахождения опорного плана, с		Прирост производительности Y
		Стандартный MPI/MPICH	MPI/MPICH_NEW	
50×50	20	0,005	0,005	–
70×70	20	0,009	0,009	–
100×100	20	0,020	0,019	1,05
150×150	20	0,031	0,029	1,06
500×500	20	1,020	0,935	1,07
500×500	40	1,005	0,920	1,09
1000×1000	40	2,501	2,209	1,13

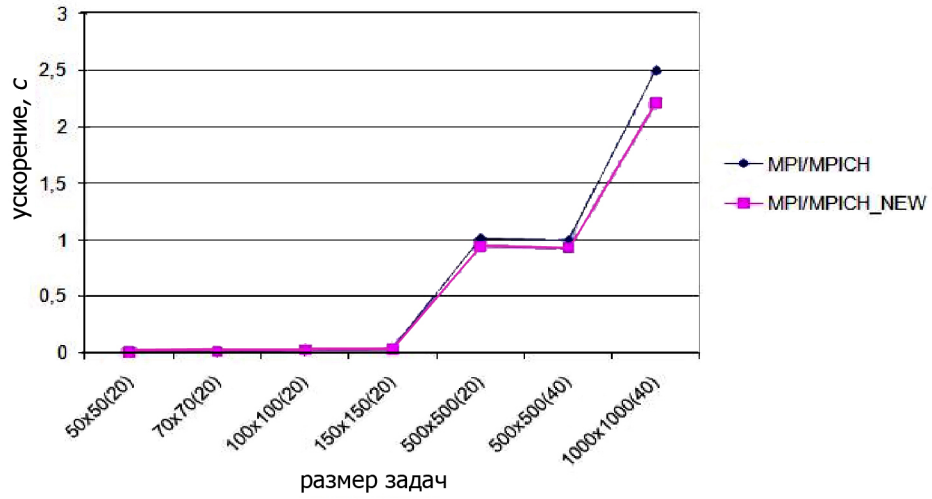

Рис.3. График времени выполнения программы в среде MPI
до и после применения разработанных алгоритмов
(в скобках указано количество процессов)

Из графиков виден незначительный прирост производительности при запуске программы в среде MPI/MPICH_NEW. Однако при увеличении числа процессов с 20 до 40 увеличение производительности становится более заметным (в 1,13 раз быстрее при расчете матрицы размера 1000×1000).

Выводы. Испытания алгоритмов управления заданиями в транспортной задаче на кластерной системе MPI/MPICH и новой модифицированной кластерной системе MPI/MPICH_NEW показали, что разработанные алгоритмы эффективно решают выбранный класс транспортных задач.

Библиографический список

1. Аль-хулайди А.А. Анализ существующих пакетов в кластерных сетях / А.А. Аль-хулайди, Н.Н. Садовой // Вестн. Донск. гос. техн. ун-та. – 2010. – Т.10, №3. – С.303-310.

2. Аль-хулайди А.А. Распределенные вычисления (кластерные вычисления) с использованием пакета параллельного программирования MPI / А.А. Аль-хулайди // Современные наукоемкие технологии. – 2010. – №4. – С.28-29.

3. Аль-хулайди А.А. Разработка параллельного алгоритма построения опорного плана транспортной задачи / А.А. Аль-хулайди // Наука и образование (МГТУ им. Н. Э. Баумана) (электрон. журн.): электрон. науч.-техн. издание. – 2011. – №5. – С.1-14. – Режим доступа: technomag.edu.ru/pdf/out/182661.pdf.

4. Каныгин Г.И. Методы оптимизации / Г.И. Каныгин, Б.Ч. Месхи, Б.В. Соболь. – М.: Феникс, 2009. – 384 с.

5. Аль-хулайди А.А. Разработка алгоритма управления порядком запуска заданий для параллельной обработки / А.А. Аль-хулайди // Инновация, экология и ресурсосберегающие технологии на предприятиях машиностроения, авиастроения, транспорта и сельского хозяйства: Тр. IX междунар. науч.-техн. конф. «ИнЭРТ-2010», 6-8 окт. 2010 г. – Ростов н/Д: Издательский центр ДГТУ, 2010. – С.426-433.

6. Коваленко В.Н. Труды международной конференции «Распределенные вычисления и Грид-технологии в науке и образовании» (Дубна, 29 июня – 2 июля 2004 г.) / В.Н. Коваленко, Д.А. Семячкин. – Дубна: ОИЯИ, 2004. – С.139-144.

Материал поступил в редакцию 17.05.2011.

References

1. Al`-xulajdi A.A. Analiz sushhestvuyushhix paketov v klasterny`x setyax / A.A. Al`-xulajdi, N.N. Sadovoj // Vestn. Donsk. gos. texn. un-ta. – 2010. – T.10, #3. – S.303-310. – In Russian.

2. Al`-xulajdi A.A. Raspredelyonny`e vy`chisleniya (klasterny`e vy`chisleniya) s ispol`zovaniem paketa parallel`nogo programmirovaniya MPI / A.A. Al`-xulajdi // Sovremenny`e naukoyomkie texnologii. – 2010. – #4. – S.28-29. – In Russian.

3. Al`-xulajdi A.A. Razrabotka parallel`nogo algoritma postroeniya opornogo plana transportnoj zadachi / A.A. Al`-xulajdi // Nauka i obrazovanie (MGTU im. N. E`. Baumana) (e`lektron. zhurn.): e`lektron. nauch.-texn. izdanie. – 2011. – #5. – S.1-14. – Rezhim dostupa: technomag.edu.ru/pdf/out/182661.pdf. – In Russian.

4. Kany`gin G.I. Metody` optimizacii / G.I. Kany`gin, B.Ch. Mesxi, B.V. Sobol`. – M.: Feniks, 2009. – 384 s. – In Russian.

5. Al`-xulajdi A.A. Razrabotka algoritma upravleniya poryadkom zapuska zadanij dlya parallel`noj obrabotki / A.A. Al`-xulajdi // Innovaciya, e`kologiya i resursosberegayushhie texnologii na

predpriyatiyax mashinostroeniya, aviastroeniya, transporta i sel`skogo xozyajstva: Tr. IX mezhdunar. nauch.-texn. konf. «InE`RT-2010», 6-8 okt. 2010 g. – Rostov n/D: Izdatel`skij centr DGTU, 2010. – S.426-433. – In Russian.

6. Kovalenko V.N. Trudy` mezhdunarodnoj konferencii «Raspredelyonny`e vy`chisleniya i Grid-texnologii v nauke i obrazovanii» (Dubna, 29 iyunya – 2 iyulya 2004 g.) / V.N. Kovalenko, D.A. Semyachkin. – Dubna: OIYAI, 2004. – S.139-144. – In Russian.

ELABORATION OF TASK CONTROL ALGORITHMS WITH PARALLEL COMPUTING IN CLUSTER COMPUTING SYSTEMS

A.A. AL-KHULAIDI, Y.O. CHERNYSHEV
(Don State Technical University)

The optimum problem of the allocation of tasks for executing in the cluster is considered. Task control algorithms are elaborated. The algorithms validation is done in the performance of the transportation problem on the local network cluster. The data obtained permit to judge efficiency of the suggested algorithms.
Keywords: *clustering, task scheduler, resource manager.*