Распределение заданий в интегрированной кластерной системе на основе их классификации

И. В. Бычков, Г. А. Опарин, А. Г. Феоктистов, А. С. Корсуков
Институт динамики систем и теории управления СО РАН, Иркутск, Россия
e-mail: bychkov@icc.ru, oparin@icc.ru, agf@icc.ru, alexask@mail.ru

Рассматривается подход к классификации заданий в интегрированной кластерной системе Института динамики систем и теории управления СО РАН. Использование классификации заданий при распределении вычислительных ресурсов для выполнения этих заданий позволяет в ряде случаев существенно улучшить показатели функционирования интегрированной кластерной системы.

Ключевые слова: классификация заданий, интегрированная кластерная система.

Введение

В настоящее время вычислительные кластеры являются неотъемлемым технологическим элементом процесса эффективного решения фундаментальных и прикладных задач. Вычислительные кластеры организуются на базе разных программно-аппаратных платформ, архитектур и коммуникационных сред и в силу этого существенно различаются по показателям производительности, эффективности и надёжности. При объединении нескольких кластеров в единую вычислительную среду (интегрированную кластерную систему), представляющую собой частный случай вычислительной Grid, возникает проблема оптимизации распределения потока ресурсоёмких вычислительных задач, поступающего в эту среду, по кластерам.

К вычислительным ресурсоёмким задачам относятся задачи, процесс решения которых осуществляется путём взаимодействия пользователя (специалиста-“предметника”) с высокопроизводительной вычислительной средой, характеризуется, как правило, относительно небольшим набором данных и значительным объёмом вычислений над ними и включает два основных этапа: формирование задания вычислительной среде для решения задачи и выполнение этого задания в вычислительной системе, что в свою очередь осуществляется с помощью специальных утилит системы управления заданиями, используемой в вычислительной среде. К таким системам относятся, во-первых, глобальные планировщики (метапланировщики) заданий на уровне Grid типа GridWay [1], во-вторых, локальные системы управления прохождением заданий (СУПЗ) вычислительных кластеров, например, PBS [2] или Condor [3]. Само задание представляет собой спецификацию процесса решения задачи, содержащую информацию о требуемых вычислительных ресурсах, исполняемых прикладных программах, входных/выходных данных и другие необходимые сведения.

Анализ результатов исследований, в той или иной степени касающихся вопросов классификации вычислительных задач и способов их постановки [4–7], а также рассмотрение способов спецификации процессов решения задач в известных метаплани-

ровщиках Grid и кластерных СУПЗ позволяют выделить целый ряд важных характеристик заданий. В их числе способ решения задачи (использование готовой программы, построение алгоритма решения задачи на основе библиотек стандартных программ, исполнение программы в режиме интерпретации или компиляции), место размещения программы (в машине пользователя или в машине системы), используемой для решения задачи, число прогонов программы, наличие взаимосвязанных подзаданий, вид параллелизма алгоритма решения задачи (мелкозернистый, крупноблочный и смешанный), степень ресурсоёмкости вычислений (малые, средние и большие задания), необходимость управления процессом вычислений в пакетном или интерактивном режимах, срочность вычислений и ряд других особенностей процесса выполнения заданий. Значение той или иной характеристики может быть явно задано пользователем вычислительной среды, формирующим свое задание, получено из переменной окружения этой среды, установлено системой управления заданиями как значение по умолчанию или взято из конфигурационных и статистических файлов (системных или пользовательских).

Известные на сегодня метапланировщики Grid базируются на двух способах распределения ресурсов [8]: путём взаимодействия метапланировщика с кластерными СУПЗ (например, GridWay) либо за счёт реализации планировщика, занимающегося поиском свободных ресурсов в Grid для выполнения конкретного приложения (например AppLeS [9]). В обоих случаях, как правило, они не производят классификацию заданий на основе характеристик заданий и осуществляют оптимизацию выделения ресурсов, разделяемых между всеми заданиями потока, для выполнения конкретного задания, оперируя информацией, касающейся только этого задания. Такой подход зачастую не позволяет добиться высоких показателей функционирования интегрированной кластерной системы в целом: её пропускной способности, эффективности загрузки ресурсов и ускорения выполнения некоторого набора задач. Отдельные системы, которые поддерживают классификацию заданий в том или ином виде (например система управления заданиями WLMS комплекса gLite [10]), имеют, как правило, достаточно ограниченный, нерасширяемый набор встроенных типов (классов) заданий и не осуществляют предварительную виртуальную декомпозицию вычислительных ресурсов относительно этих типов с целью оптимизации процесса выбора ресурса для выполнения заданий определённого типа.

В настоящей работе развивается подход к классификации заданий [11], определяющих процесс решения ресурсоёмких вычислительных задач в интегрированной кластерной системе [12]. Извлечение, обобщение и учёт всей доступной информации о характерных вычислительных особенностях заданий при их распределении позволяют в ряде случаев существенно улучшить показатели функционирования интегрированной кластерной системы.

1. Модель

Пусть имеется конечное множество характеристик заданий $H = \{h_1, h_2, ..., h_k\}$. Каждая характеристика h_i описывается информационной структурой, имеющей следующие компоненты: D_i — область допустимых значений характеристики h_i, включающая символ неопределённости θ; $r_i \geq 1$ — целочисленный ранг характеристики h_i, определяющий степень важности данной характеристики; $w_i \geq 0$ — вес характеристики h_i, представляющий собой численное выражение важности данной характеристики. Бу-

дем считать, что элементы множества H частично упорядочены по их рангу в порядке убывания: $r_i \geq r_{i+1}$, $\forall i \in \{1, 2, ..., k-1\}$.

Определим конечное множество классов заданий $C = \{c_1, c_2, ..., c_m\}$. Каждый класс c_j определяется базовым (обязательным) и дополнительным (необязательным) наборами характеристик из H. Характеристики базового и дополнительного наборов для классов заданий удобно представить в виде булевых матриц A и B размерности $k \times m$, элементы которых $a_{ij} = 1$ или $b_{ij} = 1$ означают, что характеристика h_i принадлежит базовому или дополнительному набору, используется в определении класса c_j и для этого класса имеет конкретизированную область допустимых значений $D^*_{ij} \subseteq D_i \setminus \{\theta\}$. Если $a_{ij} \vee b_{ij} = 0$, то $D^*_{ij} \equiv \{\theta\}$. Матрицы A и B должны удовлетворять условиям

$$\bigvee_{j=1}^{m} \bigwedge_{i=1}^{k} \overline{a_{ij}} = 0, \tag{1}$$

$$\bigvee_{i=1}^{k} \bigvee_{j=1}^{m} (a_{ij} \wedge b_{ij}) = 0. \tag{2}$$

Формирование множества характеристик H (определение имён, областей допустимых значений, рангов и весов характеристик) и создание на их основе множества классов заданий C (определение для каждого класса множеств обязательных и необязательных характеристик из множества H с указанием их областей значений, допустимых для конкретного класса) выполняются администратором интегрированной кластерной системы. Как правило, характеристикам классов заданий, используемым в качестве обязательных, присваиваются более высокие ранги и веса. При поступлении задания в интегрированную кластерную систему классификатор заданий автоматически проверяет, какие характеристики из множества H использованы в спецификации этого задания, и определяет для каждой из них область значений, требуемую в его спецификации для выполнения задания.

Задание со всеми своими характеристиками представляется булевым вектором x размерности k. Между индексами элементов вектора x и индексами характеристик из H установлено взаимнооднозначное соответствие. Значение i-го элемента вектора x определяется следующим образом:

$$x_i = \begin{cases} 0, & \text{если } D'_i \equiv \{\theta\}, \\ 1, & \text{если } D'_i \subseteq D_i \setminus \{\theta\}, \end{cases}$$

где D'_i — область значений характеристики h_i, требуемых для выполнения данного задания.

Вектор x должен удовлетворять условию $\bigwedge_{i=1}^{k} \overline{x_i} = 0$. Соответствие требуемых областей допустимых значений характеристик задания областям допустимых значений характеристик j-го класса заданий из C устанавливается с помощью характеристической функции

$$\chi_j(x) = \begin{cases} 0, & \text{если } \exists i : (a_{ij} \vee b_{ij} = 1) \wedge D'_i \not\subseteq D^*_{ij}, \\ 1 & \text{в противном случае,} \end{cases}$$

где $i \in \{1, 2, ..., k\}$, $j \in \{1, 2, ..., m\}$.

Для выполнения первичной классификации задания достаточно применения функции χ. В результате первичной классификации задание может быть соотнесено сразу с несколькими классами заданий из C. Однако может возникнуть необходимость в более

детальной конкретизации классификации задания, полученной с помощью функции χ. В этом случае следует принимать во внимание количество характеристик задания, требуемые области допустимых значений которых удовлетворяют соответствующим областям допустимых значений характеристик того или иного класса, и учитывать ранги и веса этих характеристик.

Ниже введён ряд вспомогательных функций.

Будем считать, что i-я характеристика используется как в спецификации задания, так и в описании класса c_j, если выполняется условие

$$\overline{x_i} \vee \overline{a_{ij}}\overline{b_{ij}} = 0. \tag{3}$$

Область требуемых значений i-й характеристики соответствует области допустимых значений этой характеристики для класса c_j, если выполняется условие

$$D_i^{'} \subseteq D_{ij}^{*}. \tag{4}$$

Функция $\rho_j(x)$ вычисляет количественную оценку возможности (вероятность) отнесения задания к классу c_j на основе характеристик этого задания, удовлетворяющих по всем параметрам характеристикам данного класса c_j. Значение функции определяется выражением $\rho_j(x) = \dfrac{V_j}{S}$, где $V_j = \dfrac{k_j}{n_j}$, $S = \sum\limits_{l=1}^{m}\dfrac{k_l}{n_l}$, k_j (k_l) — число характеристик задания h_i, удовлетворяющих условиям (1)–(4) относительно j-го (l-го) класса, n_j (n_l) — число характеристик, определяющих класс c_j (c_l), k_j, $k_l \in \{0, 1, ..., k\}$, n_j, $n_l \in \{1, ..., m\}$. Из этого следует, что $\rho_j(x) \in [0, 1]$ и $\sum\limits_{j=1}^{m}\rho_j(x) = 1$.

Функция $\sigma_j(x)$ вычисляет агрегированный числовой показатель важности характеристик задания для класса c_j с учётом рангов этих характеристик. Значение функции определяется выражением

$$\sigma_j(x) = \begin{cases} \sum\limits_{l=q}^{t} n_l s_l + k \sum\limits_{l=q+1}^{t} s_l & \text{при } q < t, \\ n_t s_t & \text{при } q = t, \end{cases}$$

где $s_q, s_{q+1}, .., s_t$ — упорядоченные по убыванию значения рангов характеристик; n_l (n_v) — число характеристик заданий h_i ранга s_l (s_v), удовлетворяющих условиям (1)–(4) относительно j-го класса; $q = \min\limits_{v=1}^{t}\{v : n_v > 0\}$; t — число рангов характеристик в классификаторе заданий; $t \le k$; $\sigma_j(x) \ge 1$.

Функция $\omega_j(x)$ вычисляет сумму весов характеристик задания для класса c_j. Значение функции определяется выражением $\omega_j(x) = \sum\limits_{i \in I}\omega_i$, где I — множество индексов характеристик задания h_i, удовлетворяющих условиям (1)–(4) относительно j-го класса, $\omega_j(x) \ge 0$.

Функция $\phi_j(x, z)$ вычисляет количественную оценку возможности (вероятность) отнесения задания к классу c_j с учётом его вычислительной истории z — информационной структуры, содержащей статистику о выполнении в интегрированной кластерной системе заданий с аналогичными характеристиками. Данная информация извлекается из статистических и информационных файлов СУПЗ, применяемых в интегрированной

кластерной системе. Значение функции определяется выражением

$$\phi_j(x,z) = \begin{cases} \dfrac{1}{m}, & \text{если } \gamma_l(x,z) = 0, \quad \forall l \in \{1,2,...,m\}, \\ \dfrac{\gamma_j(x,z)}{\sum\limits_{l=1}^{m}\gamma_l(x,z)} & \text{в противном случае,} \end{cases}$$

где $\gamma_l(x,z)$ и $\gamma_j(x,z)$ — функции, возвращающие число выполненных в интегрированной кластерной системе заданий классов c_l и c_j с аналогичными характеристиками. Нетрудно видеть, что $\phi_j(x,z) \in [0,1]$ и $\sum\limits_{j=1}^{m}\phi_j(x,z) = 1$.

Определим также верхние границы эквивалентности значений функций ρ, σ, ω и ϕ — δ_ρ, δ_σ, δ_ω, δ_ϕ. Два значения одной и той же функции будем считать эквивалентными, если модуль их разницы будет либо меньше соответствующей верхней границы эквивалентности, либо равен ей.

Пусть с помощью функции χ сформирован булев вектор y размерности m, элемент которого $y_j = 1$ означает, что задание относится к j-му классу. Введём следующие характеристические функции, конкретизирующие принадлежность задания j-му классу с учётом перечисленной выше дополнительной информации о характеристиках задания:

$$\chi_j^\rho(x,y) = \begin{cases} 0, & \text{если } \max\limits_{\forall l:y_l=1}\{\rho_l(x)\} - \rho_j(x) > \delta_\rho, \\ 1 & \text{в противном случае,} \end{cases}$$

$$\chi_j^\sigma(x,y) = \begin{cases} 0, & \text{если } \max\limits_{\forall l:y_l=1}\{\sigma_l(x)\} - \sigma_j(x) > \delta_\sigma, \\ 1 & \text{в противном случае,} \end{cases}$$

$$\chi_j^\omega(x,y) = \begin{cases} 0, & \text{если } \max\limits_{\forall l:y_l=1}\{\omega_l(x)\} - \omega_j(x) > \delta_\omega, \\ 1 & \text{в противном случае,} \end{cases}$$

$$\phi_j^\omega(x,y,z) = \begin{cases} 0, & \text{если } \max\limits_{\forall l:y_l=1}\{\phi_l(x,z)\} - \phi_j(x,z) > \delta_\phi, \\ 1 & \text{в противном случае.} \end{cases}$$

Следует заметить, что наличие у задания отдельно взятой характеристики с высокой значимостью (как обязательной, так и необязательной) не гарантирует отнесения этого задания к классу, в описании которого присутствует данная характеристика. Классификация осуществляется с учётом агрегированной значимости всех характеристик задания для каждого класса заданий.

2. Алгоритм классификации задания

Рассмотрим алгоритм классификации задания. На вход алгоритма поступает вектор x, представляющий задание. На выходе получаем вектор y, содержащий информацию о классах, к которым относится данное задание. Алгоритм включает следующие этапы работы.

I) Инициализация элементов вектора $y : y_j = 0, \forall j \in \{1,2,...,m\}$;

II) выполнение первичной классификации задания — проверка принадлежности задания каждому из m классов: $y_j = \chi_j \{x\}$, $\forall j \in \{1, 2, ..., m\}$;
III) если $y_j = 0$, $\forall j \in \{1, 2, ..., m\}$, то завершение работы алгоритма (задание не может быть классифицировано);
IV) иначе — выполнение конкретизации результатов классификации задания, полученной на этапе II:

   a) редукция множества классов, к которым относится задание, с учётом количества характеристик классов: $y_j = \chi_j^{\rho} \{x, y\}$, $\forall j : y_j = 1$;
   b) редукция множества классов, к которым относится задание, с учётом рангов характеристик: $y_j = \chi_j^{\sigma} \{x, y\}$, $\forall j : y_j = 1$;
   c) редукция множества классов задания, к которым относится задание, с учётом весов характеристик: $y_j = \chi_j^{\omega} \{x, y\}$, $\forall j : y_j = 1$;
   d) редукция множества классов задания, к которым относится задание, с учётом вычислительной истории данного задания: $y_j = \chi_j^{\phi} \{x, y, z\}$, $\forall j : y_j = 1$;

V) завершение работы алгоритма (задание классифицировано).

Порядок выполнения этапов IV, a – d работы алгоритма, приведённый выше, определён исходя из степени доступности информации, необходимой для функций ρ, σ, ω и ϕ, используемых на этих этапах. Чем выше степень доступности информации, требуемой для применения той или иной функции, тем раньше выполняется соответствующий этап работы алгоритма. В программной реализации алгоритма классификации заданий предусмотрена возможность исключения любого этапа IV, a – d из алгоритма и/или изменения порядка применения этих этапов в алгоритме путём конфигурирования программы-классификатора администратором интегрированной кластерной системы. Исключение того или иного этапа IV, a – d работы алгоритма может быть обусловлено рядом причин, а именно: администратор системы не обладает достаточной информацией для назначения рангов и весов характеристик (по умолчанию характеристики имеют одинаковые ранги и веса); нет возможности получить полную вычислительную историю о выполнении заданий вследствие использования СУПЗ, работу с которыми программа-классификатор не поддерживает (в текущей версии классификатора заданий поддерживается работа с СУПЗ PBS, Condor и Cleo [13]); результаты применения исключаемого этапа являются избыточными с точки зрения администратора системы. Изменение порядка применения этапов IV, a – d работы алгоритма может быть обусловлено степенью эффективности редуцирующих преобразований, применяемых на этих этапах. Данный показатель определяется на основе статистики о работе классификатора заданий, сохраняемой в его информационных файлах.

3. Классификатор заданий

На основе проведённых исследований выполнена программная реализация имитационного прототипа классификатора заданий. Имитационное моделирование работы классификатора заданий было проведено на потоке заданий, соответствующем потоку реальных заданий, выполненных на кластере Blackford с СУПЗ Cleo (табл. 1). В таблице для каждого кластера указаны общее число его узлов n_n и ядер n_c, а также усреднённый коэффициент увеличения времени выполнения заданий k_{run}.

Всего было обработано 62 249 заданий. Список использованных классов заданий приведен в табл. 2. Для каждого класса заданий указаны диапазоны допустимых значений ряда его базовых характеристик: количество вариантов данных n_v, число требуемых ядер n_c, запрашиваемое время для выполнения задания t.

Перед классификацией каждому кластеру (см. табл. 1) назначены наиболее подходящие ему классы заданий. Все задания были классифицированы. Для части заданий выявлена потенциальная возможность их перемещения с кластера Blackford на другие кластеры с целью оптимизации загрузки вычислительных ресурсов. Перемещение задания возможно только на кластер, удовлетворяющий характеристикам класса задания, и при условии, что время выполнения этого задания останется в диапазоне времени выполнения заданий данного класса. Для заданий классов *univariant sequential large* и *univariant parallel large* учитывалось дополнительное ограничение: перемещённое задание не должно выполняться дольше суток. Количество классифицированных заданий n_{job} и перемещённых заданий n_r для каждого класса приведены в табл. 3.

Полученные результаты показали, что дополнительное применение классификатора заданий в процессе распределения заданий совместно с метапланировщиком заданий интегрированной кластерной системы даёт возможность повысить коэффициент полезного использования узлов системы на 18 %.

Т а б л и ц а 1. Вычислительные ресурсы кластерной системы ИДСТУ СО РАН

Кластер	n_n, ед.	n_c, ед.	k_{run}
Blackford	20	160	1
МВС-1000/16	16	32	3
Кластер невыделенных рабочих машин, организованный на базе ПЭВМ научных лабораторий	12	12	2

Т а б л и ц а 2. Классы заданий в интегрированной кластерной системе

Класс заданий	Характеристика		
	n_v, ед.	n_c, ед.	t, мин
Univariant sequential small	1	1	1 – 5
Univariant sequential medium	1	1	5 – 60
Univariant sequential large	1	1	> 60
Univariant parallel small	1	≥ 2	1 – 5
Univariant parallel medium	1	≥ 2	5 – 60
Univariant parallel large	1	≥ 2	> 60

Т а б л и ц а 3. Результаты классификации заданий

Класс заданий	n_{job}	n_r
Univariant sequential small	10221	4675
Univariant sequential medium	7927	5861
Univariant sequential large	1211	972
Univariant parallel small	33453	492
Univariant parallel medium	6125	50
Univariant parallel large	3312	95

Таким образом, представленный в статье подход к классификации заданий позволяет расширить функциональные возможности метапланировщика интегрированной кластерной системы и осуществлять оптимизацию распределения вычислительных ресурсов системы не только на уровне отдельно взятого задания, но и на уровне целого класса заданий.

Список литературы

[1] Herrera J., Huedo E., Montero R., Llorente I. Porting of scientific applications to Grid computing on GridWay // Sci. Program. 2005. Vol. 13, No. 4. P. 317–331.

[2] Henderson R. Job scheduling under the portable batch system // Job Scheduling Strategies for Parallel Processing. Springer, 1995. P. 279–294.

[3] Litzkow M., Livny M., Mutka M. Condor — a hunter of idle workstations // In 8th Internat. Conf. of Distributed Computing Systems (ICDCS). IEEE CS Press, Los Alamitos, CA, USA, 1988. P. 104–111.

[4] Человек и вычислительная техника / Под ред. В.М. Глушкова. Киев: Наукова думка, 1971. 294 с.

[5] Тыугу Э.Х. Концептуальное программирование. М.: Наука, 1984. 256 с.

[6] Иванов В.В. Методы вычислений на ЭВМ: Справочное пособие. Киев: Наукова думка, 1986. 584 с.

[7] Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. СПб.: БХВ-Петербург, 2002. 608 с.

[8] Топорков В.В. Модели распределенных вычислений. М.: Физматлит, 2004. 320 с.

[9] Berman F., Wolski R., Casanova H. et al. Adaptive computing on the grid using AppLeS // IEEE Trans. on Parallel and Distributed Systems. 2003. Vol. 14, No. 4. P. 369–382.

[10] Коваленко В.Н. Комплексное программное обеспечение грида вычислительного типа. М., 2007 (Препр. ИПМ РАН. № 10. 39 с.).

[11] Бычков И.В., Корсуков А.С., Опарин Г.А., Феоктистов А.Г. Инструментальный комплекс для организации гетерогенных распределённых вычислительных сред // Информ. технологии и вычисл. системы. 2010. № 1. С. 45–54.

[12] Бычков И.В., Опарин Г.А., Новопашин А.П. и др. Высокопроизводительные вычислительные ресурсы ИДСТУ СО РАН: Текущее состояние, возможности и перспективы развития // Вычисл. технологии. 2010. Т. 15, № 3. С. 69–82.

[13] Система управления заданиями Cleo. Руководство пользователя. М.: НИВЦ МГУ, 2007. 19 с.

Поступила в редакцию 27 ноября 2012 г.,
с доработки — 18 января 2013 г.