РОССИЙСКАЯ АКАДЕМИЯ НАУК
СИБИРСКОЕ ОТДЕЛЕНИЕ
АВТОМЕТРИЯ

№ 3 2000

УДК 681.324

В. Г. Хорошевский, В. В. Власюк
(Новосибирск)

ТЕОРЕТИКО-ИГРОВОЙ ПОДХОД К ОРГАНИЗАЦИИ СТОХАСТИЧЕСКИ ОПТИМАЛЬНОГО ФУНКЦИОНИРОВАНИЯ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ *

Ставится задача организации стохастически оптимального функционирования большемасштабных распределенных вычислительных систем (ВС). Применен теоретико-игровой подход к решению задачи. Фактически построена матричная игра, моделирующая «диспетчеризацию», и разработан параллельный вычислительный метод ее решения. Метод основывается на композиции симплекс-метода и метода Брауна – Робинсон. Построен параллельный алгоритм организации стохастически оптимального функционирования распределенных ВС, осуществлено его моделирование на мультитранспьютерной системе. Проведено сравнительное тестирование указанного алгоритма и симплекс-метода решения игр.

Объектом исследования в данной работе являются распределенные вычислительные системы (ВС). Функциональная структура таких ВС [1] представляется как композиция из множеств элементарных процессоров (или машин) и модулей памяти (локальной памяти для процессоров), коммуникационной сети (для межпроцессорных взаимодействий) и средств управления. Характерная архитектурная особенность распределенных ВС – их большемасштабность. Так, например, в системе Connection Machine фирмы "Thinking Machines Corp." число элементарных процессоров может быть равно 65536. Ясно, что любая из распределенных ВС по своей природе – сложный стохастический объект: в ней отказы и восстановления компонент могут происходить в случайные моменты времени. Следовательно, естественной является постановка задачи об организации стохастически оптимального функционирования распределенных ВС. Далее, сложность решения такой задачи приводит к необходимости разработки параллельных методов и алгоритмов и применения параллельных ВС для ее решения. Итак, в операционную систему любой большемасштабной распределенной ВС должен быть вложен параллельный алгоритм, организующий ее стохастически оптимальное функционирование.

В данной работе исследуется теоретико-игровой подход к эффективному распределению потока задач между элементарными машинами (ЭМ) в рас-

* Работа выполнена при поддержке Российского фонда фундаментальных исследований (гранты № 97-01-00883, № 99-07-90206).

пределенных вычислительных системах. Возможность такого подхода была впервые указана в [2].

Процесс распределения задач из потока можно представить в виде матричной игры, матрица платежей которой строится на основе стоимостей эксплуатации, простоя и ремонта ЭМ системы и не зависит от решаемых задач. Решением этой игры будет оптимальная стратегия, придерживаясь которой можно достичь стохастически оптимального режима работы ВС (при котором достигается минимум затрат на эксплуатацию системы в единицу времени). Цена построенной матричной игры будет представлять собой минимальное гарантированное математическое ожидание затрат на эксплуатацию ВС в единицу времени, что позволяет оценить, например, ожидаемый доход от эксплуатации данной ВС.

В работе представлен параллельный ускоренный алгоритм решения матричных игр (ПУМИ). В основе этого алгоритма лежит симплекс-метод. Ускорение симплекс-метода происходит за счет параллельно выполняющегося алгоритма отсеивания плохих чистых стратегий. Проведено численное моделирование, показывающее преимущество в скорости полученного алгоритма перед симплекс-методом.

1. **Постановка задачи.** Имеется распределенная ВС, состоящая из n ЭМ. На ВС поступает поток параллельных задач. Каждая задача представлена парой чисел $\langle t, j\rangle$, где t – время решения задачи, j – ее ранг. Считается, что задача имеет ранг $1 \le j \le n$, если для ее решения необходимо использовать j машин.

Далее, имеется также операционная система (ОС), распределяющая задачи по элементарным машинам ВС. Затраты на решение той или иной задачи будем представлять как «платеж» ОС вычислительной системе, получаемый на основе известных стоимостей эксплуатации, простоя и ремонта каждой ЭМ в единицу времени. Требуется построить такой алгоритм функционирования ОС, чтобы затраты на решение задач в единицу времени были минимальны.

2. **Игровая модель.** Представим процесс распределения задач по ЭМ вычислительной системы в виде игры двух объектов – ВС и ОС. Будем считать, что ВС использует чистую стратегию с номером $i \in \{0,1,\dots,n\}$, если она для решения задач отводит i машин, и что ОС применяет чистую стратегию с номером $j \in \{0,1,\dots,n\}$, если она назначает для ВС задачу ранга j. Если ВС выбирает стратегию с номером i, а ОС – с номером j, то ОС «платит» ВС сумму c_{ij}. Числа c_{ij} составляют матрицу платежей C.

Из теории игр [3] следует, что если ОС использует оптимальную стратегию при назначении задач, то при любых стратегиях ВС операционная система выплатит в единицу времени в среднем не более чем цена игры. Такая стратегия ОС гарантирует вполне определенные затраты на решение задач в единицу времени независимо от того, как ведет себя ВС.

Если ВС и ОС применяют смешанные стратегии соответственно $p = (p_0, p_1, \dots, p_n)^T$, $\pi = (\pi_0, \pi_1, \dots, \pi_n)^T$, то средний платеж вычислительной системе будет равен

$$\sum_{j=0}^{n} \sum_{i=0}^{n} c_{ij} p_i \pi_j = p^T C \pi.$$

Согласно [3], оба игрока всегда имеют оптимальные смешанные стратегии (p^* для ВС и π^* для ОС) такие, что $p^T C\pi^* \leq v$ для всех p, $(p^*)^T C\pi \geq v$ для всех π, где $v = (p^*)^T C\pi^*$ – цена игры.

Определим матрицу C. В [1] предлагается взять следующую матрицу:

$$c_{ij} = \begin{cases} jc_1 + (i-j)c_2 & \text{при} \quad i \geq j; \\ ic_2 + (j-i)c_3 & \text{при} \quad i < j, \end{cases}$$

где c_1 – платеж за использование одной машины в течение единицы времени; c_2 и c_3 – штрафы в единицу времени соответственно за простой одной машины и при $j - i = 1$. Эта матрица означает, по сути, следующее. Если после назначения задач в вычислительной системе остаются еще незанятые ЭМ, то ОС платит за эксплуатацию ЭМ, решающих задачу, плюс за простой свободных ЭМ. Если же исправных машин не хватает для решения назначенной задачи, то ОС, во-первых, платит за простой свободных исправных машин и, во-вторых, тратит определенную сумму на восстановление неисправных ЭМ, недостающих для решения задачи.

В работе [1] отмечено, что алгоритм функционирования ОС, состоящий в назначении задач одного ранга, представляется неэффективным, следовательно, матрица C не должна иметь седловых точек. Там же доказана

Теорема. Матрица C не имеет седловых точек тогда и только тогда, когда $c_1 < \min\{c_2, c_3\}$.

3. **Решение матричных игр.**

Определение 1. В матричной игре игроком I (первым игроком) будем называть игрока строками, который в каждой партии выбирает одну из строк платежной матрицы, соответственно игроком II (вторым игроком) – игрока, который играет столбцами.

Таким образом, в нашей постановке ВС является игроком I, а ОС – игроком II.

Определение 2. Чистая стратегия любого игрока называется плохой, если она имеет нулевую вероятность применения в любой оптимальной стратегии этого игрока.

3.1. *Предварительные замечания.* Существует много методов для точного и приближенного вычислений оптимальных стратегий в матричных играх. Для наших целей метод решения игр должен быть ориентирован на матрицы больших размеров (порядки которых могут исчисляться сотнями тысяч, что является следствием большемасштабности ВС), рассчитан на поиск только одной (любой) стратегии из всего множества оптимальных стратегий, а также должен позволять находить точные решения.

Всем этим свойствам удовлетворяет симплекс-метод (его применение здесь допустимо ввиду эквивалентности задач линейного программирования и решения матричных игр). Очевидно, что время решения игры сократится, если в игровой матрице удастся избавиться от плохих столбцов, т. е. столбцов, соответствующих плохим стратегиям. Описанный далее алгоритм ускорения решения матричных игр основан как раз на сокращении размера платежной матрицы за счет удаления плохих столбцов.

Найти плохие чистые стратегии аналитически, не решая игры, в общем случае не представляется возможным. Сразу при первоначальном анализе платежной матрицы можно отбросить лишь очень немногие плохие стратегии, опираясь, к примеру, на поэлементное доминирование некоторых столбцов над другими. Как выяснилось, существуют простые итеративные методы решения игр, которые непригодны для решения больших игр, но способны быстро и с большой вероятностью указать на плохие стратегии. Таким методом является метод Брауна – Робинсон [4]. Реально была использована модификация этого метода [5], в которой ускорена сходимость по сравнению с обычным методом Брауна – Робинсон.

3.2. *Модифицированный метод Брауна – Робинсон (МБР)*. Этот метод можно применять для решения только симметричных игр (т. е. игр с кососимметричной платежной матрицей). Это ограничение не уменьшает общности: как показано в [6], любую матричную игру можно заменить эквивалентной ей симметричной игрой.

3.2.1. Симметризация матричных игр. Пусть дана платежная матрица $C = (c_{ij})$ размером $m \times n$. Составим следующую кососимметричную матрицу размером $(m+n+2) \times (m+n+2)$:

$$A = \begin{pmatrix} 0 & C & -1 \\ -C^T & 0 & 1 \\ 1 & -1 & 0 \end{pmatrix},$$

где числами $K \in \{0, 1, -1\}$ обозначены матрицы соответствующих размеров со всеми элементами, равными K.

В работе [6] показано, что если цена игры с матрицей C больше нуля и оптимальные стратегии для обоих игроков игры A (в симметричной игре множества оптимальных стратегий для обоих игроков совпадают) равны $(x_0, \ldots, x_m; y_0, \ldots, y_n; \lambda)$, тогда оптимальная стратегия первого игрока игры с матрицей C будет равна $\frac{1}{\mu}(x_0, \ldots, x_m)$, а второго соответственно – $\frac{1}{\mu}(y_0, \ldots$ $\ldots, y_n)$, где $\mu = \sum_{i=0}^{m} x_i = \sum_{j=0}^{n} y_j$.

Может показаться, что переход к симметричной игре повлечет сильное ухудшение сходимости обычного метода Брауна – Робинсон, так как размерность матрицы при симметризации значительно возрастает. Однако из работы [7] следует, что при такой симметризации игры сходимость метода Брауна – Робинсон ухудшается на очень малую величину по сравнению со сходимостью метода для исходной матрицы, а если размеры игровой матрицы, как в нашем случае, велики, то это ухудшение абсолютно незаметно.

Таким образом, ясно, что МБР можно эффективно применять и для несимметричных игр, рост размеров платежной матрицы вследствие симметризации слабо повлияет на улучшение сходимости по сравнению с обычным методом Брауна – Робинсон.

3.2.2. Описание модифицированного метода Брауна – Робинсон. Из всех стратегий операционной системы π, удовлетворяющих неравенствам

$$\sum_{j=0}^{n} c_{ij}\pi_j \le \alpha, \quad i = 0, \ldots, m, \tag{1}$$

оптимальными стратегиями π^* будут такие векторы, для которых $\sum_{j=0}^{n} c_{ij}\pi_j^* \le$ $\le v$, $i=0,\dots,m$, а ценой v – наименьшее из чисел α, удовлетворяющих условию (1). Поэтому задачу поиска оптимальной стратегии для ОС можно записать в виде

$$\alpha \to \min_{\{\pi_j\}},\ \ C\pi \le \alpha,\ \ \pi \ge 0,\ \ \sum_{j=0}^{n} \pi_j = 1,$$

где все векторные неравенства выполняются поэлементно. После симметризации получим эквивалентную задачу с кососимметричной матрицей A:

$$\sigma \to \min_{\{x_j\}},\ \ Ax \le \sigma,\ \ x \ge 0,\ \ \sum_{j=0}^{2n+2} x_j = 1.$$

Если положить $\xi = Ax$, то задача сводится к минимизации функции $\sigma(x) =$ $= \max_j \xi_j = \max_j (Ax)_j$, неотрицательной на симплексе размерности n и обращающейся в нуль при $x = x^*$ (цена симметричной игры всегда равна нулю). Суть МБР заключается в последовательном уменьшении σ.

Пусть имеется некоторая стратегия x и $\xi = Ax$. Предположим, что ξ_i – единственная максимальная компонента ξ. Тогда от стратегии x можно перейти к стратегии $x' = (1-\alpha)x + \alpha e_i$, где $e_i = (0,\dots,0,1,0,\dots,0)^T$, $0 \le \alpha \le 1$ (отметим, что e_i – оптимальный ответ на стратегию x), которая при некоторых α будет лучше x в том смысле, что $\sigma(x') < \sigma(x)$. Действительно, $\xi' = Ax' =$ $= (1-\alpha)\xi + \alpha a_i$, где через a_i обозначен i-й столбец матрицы A. Так как $a_{ii} = 0$, то i-я компонента вектора ξ' убывает с ростом α. Однако ξ'_i при малых α будет оставаться максимальной компонентой вектора ξ', т. е.

$$\sigma' = \sigma(x') = \xi'_i = (1-\alpha)\xi_i = (1-\alpha)\sigma < \sigma.$$

Ясно, что α можно увеличивать до тех пор, пока ξ'_i остается максимальной компонентой, т. е. не наступит равенство $\xi'_i = \xi'_k$ при некотором $k \ne i$ (если ни при каком $0 < \alpha < 1$ это равенство не будет выполняться, то можно взять $\alpha = 1$, тогда $x' = e_i$ – решение игры, поскольку $\sigma' = 0$). Переписав это равенство

$$(1-\alpha)\xi_i = (1-\alpha)\xi_k + \alpha a_{ki}, \tag{2}$$

можно заметить, что конкурентной компонентой будет компонента с индексом

$$k = \arg\min_{j \ne i} \frac{\xi_i - \xi_j}{a_{ji} + \xi_i - \xi_j},$$

при этом величина шага будет равна

$$\alpha = \min_{j \ne i} \frac{\xi_i - \xi_j}{a_{ji} + \xi_i - \xi_j} = \frac{\xi_i - \xi_k}{a_{ki} + \xi_i - \xi_k}.$$

Отметим, что из равенства (2) следует, что $a_{ki} > 0$.

Перейдя от стратегии x к стратегии x', можно повторить описанный процесс с x' вместо x. Укажем на то обстоятельство, что теперь у нас, по крайней мере, две максимальные компоненты ξ': ξ'_i и ξ'_k. Однако в новом состоянии роль i будет играть k, а i отходит на второй план и не может составить конкуренции k, поскольку $a_{ik} = -a_{ki} < 0$.

З а м е ч а н и е. Реально мы будем применять МБР для решения игры с матрицей вида

$$A = \begin{pmatrix} 0 & C & -1 \\ -C^T & 0 & 1 \\ 1 & -1 & 0 \end{pmatrix},$$

где C – положительная матрица. Разобьем матрицу A на три горизонтальных участка:

$$\begin{matrix} 0 & C & -1 & \} & \mathrm{I} \\ -C^T & 0 & 1 & \} & \mathrm{II}. \\ 1 & -1 & 0 & \} & \mathrm{III} \end{matrix}$$

Тогда вектор ξ также разбивается на три части соответственно матрице A:

$$\xi = (\underbrace{\xi_1, \ldots, \xi_m}_{\mathrm{I}}, \underbrace{\xi_{m+1}, \ldots, \xi_{m+n}}_{\mathrm{II}}, \underbrace{\xi_{m+n+1}}_{\mathrm{III}})^T.$$

Заметим, что если изначально положить $\xi = a_1$ (a_1 – первый столбец матрицы A) и выполнять итерации МБР согласно изложенным правилам, то конкурентная компонента ξ_k, возникающая на каждой итерации, будет поочередно принадлежать частям II, I, III вектора ξ. Таким образом, чтобы найти номер k на каждой итерации, следует искать его в соответствующей группе.

3.3. *Параллельный ускоренный метод решения матричных игр.* Будем считать, что задана симметричная игра с кососимметричной матрицей

$$A = \begin{pmatrix} 0 & C & -1 \\ -C^T & 0 & 1 \\ 1 & -1 & 0 \end{pmatrix}$$

порядка n. Цена такой игры равна нулю. Игроков можно не различать.

Заметим, что для того, чтобы k-я стратегия была плохой, достаточно, чтобы максимум ξ (из МБР) достигался на k-й компоненте лишь конечное число раз. Несомненно, это условие теоретически не является необходимым. Однако во всех проведенных экспериментах для всех плохих стратегий оно выполнялось. Поэтому авторы сочли достаточным найти лишь такие плохие стратегии.

Итак, зададимся целью найти такие k-е стратегии, для которых выполнено условие:

максимум ξ достигается на k-й компоненте конечное число раз. (3)

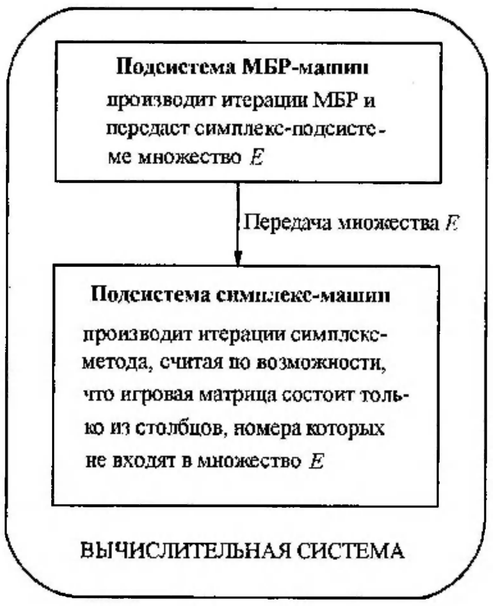
Пусть в текущий момент времени выполняется p-я итерация МБР. Зафиксируем некоторую k-ю компоненту вектора ξ. Возьмем подпоследовательность номеров уже совершенных итераций МБР: $i_0 = 1, i_1, i_2, \ldots, i_{q-1}, i_q = p$, где через $i_1, i_2, \ldots, i_{q-1}$ обозначены номера итераций МБР, на которых максимум ξ достигается на k-й компоненте.

Пусть $\lambda_j = \alpha_1 + \alpha_2 + \ldots + \alpha_j$, где α_m равно шагу МБР (см. п. 3.2.2) на m-й итерации. И наконец, пусть $\omega_k(p) = \max\limits_{j \le q}(\lambda_{i_j} - \lambda_{i_{j-1}})$.

В работе [5] замечено, что $\alpha_j \to 0$ при $j \to \infty$ и $\sum\limits_{j=1}^{\infty} \alpha_j = \infty$. Таким образом, если k-я стратегия удовлетворяет условию (3), то, очевидно, $\omega_k(p) \to \infty$ при $p \to \infty$. Если для какой-либо другой стратегии i условие (3) не выполнено, то $\omega_i(p)$ все равно теоретически может стремиться к бесконечности. Однако в этом случае $\omega_i(p) < \omega_k(p)$, если p больше некоторого p_0, которое, в свою очередь, больше номера той итерации МБР, в которой максимум ξ достигался на k-й компоненте в последний раз.

В связи с изложенным предлагается следующий план действий. Зададим вначале некоторую константу M (неплохие результаты дает, например, $M = 5$). На ВС запустим параллельно симплекс-метод решения поставленной игры и МБР, которые начинают работать независимо. Элементарные машины, занятые выполнением симплекс-метода или МБР, будем называть симплекс-машинами или МБР-машинами соответственно.

Введем вектор $\chi = (\chi_1, \chi_2, \ldots, \chi_n)$ с числом компонент, равным размерности кососимметричной матрицы A. Изначально все $\chi_i = 0$. Этот вектор обрабатывается МБР-машинами следующим образом: после каждой итерации МБР полагаем $\chi_k = 0, \chi_i := \chi_i + \alpha$ для всех i, где k – номер максимальной компоненты вектора ξ в текущей итерации; α – шаг в текущей итерации МБР.



Общая схема выполнения ПУМИ на вычислительной системе

Как только окажется, что для некоторых i неравенство $\chi_i > M$ выполняется, тогда симплекс-машинам посылаются эти номера i (обозначим их множество через E). Симплекс-машины стараются по возможности игнорировать столбцы игровой матрицы, соответствующие номерам из множества E. Величина M увеличивается на малую константу (порядка 0,1), и все повторяется.

4. **Результаты моделирования.** Описанный алгоритм реализован на транспьютерной системе. Программа моделирования работает под управлением операционной системы МИКРОС-Т, созданной в отделе вычислительных систем Института физики полупроводников СО РАН. Общая структура программы показана на рисунке.

Программа обладает тем достоинством, что она допускает формирование конфигураций вычислительной системы из симплекс-машин и МБР-машин в произвольном соотношении. Это позволяет осуществлять адаптацию конфигурации программы в зависимости от размеров игровой матрицы.

4 симплекс-машины, t_1	3 симплекс-машины, 1 МБР-машина, t_2	$100 \cdot t_2/t_1$, %
44,36	36,01	81,17
41,73	32,80	78,60
35,97	32,51	90,39
40,34	31,92	79,14
39,85	29,30	73,52
39,27	34,48	87,79
44,17	31,49	71,27
36,18	35,35	97,70
43,57	28,50	65,41
44,93	32,47	72,26
38,74	33,14	85,55
52,20	51,02	97,74
58,39	47,27	80,95
40,35	36,44	90,32
36,60	26,30	71,86
40,93	26,22	64,07
43,37	39,10	90,17
34,86	32,93	94,46
44,39	40,41	91,04
45,20	29,11	64,40
42,27	34,34	81,39

Для иллюстрации приведем результаты решения 20 игр с матрицами размерами 400×400. В каждой строчке таблицы записано время решения в минутах одной и той же игры при разных конфигурациях программы. В первом столбце указаны времена решения игр на четырех симплекс-машинах. Во втором столбце приведены времена решения тех же игр, но методом ПУМИ на конфигурации из трех симплекс-машин и одной МБР-машины. Последний столбец содержит процентное отношение между первыми двумя величинами каждой строки. В последней строке приведены средние значения для каждого столбца. Игровые матрицы генерировались псевдослучайным образом.

Из таблицы видно, что в проведенных экспериментах ПУМИ оказался в среднем почти на 20 % быстрее симплекс-метода. Таким образом, задача организации стохастически оптимального функционирования большемасштабных распределенных ВС при применении теоретико-игрового подхода решается только один раз. Предложенный алгоритм решения эффективно распараллеливается и реализуется на рассматриваемой ВС.

СПИСОК ЛИТЕРАТУРЫ

1. **Евреинов Э. В., Хорошевский В. Г.** Однородные вычислительные системы. Новосибирск: Наука, 1978.
2. **Хорошевский В. Г.** Об алгоритмах функционирования однородных универсальных вычислительных систем // Вычислительные системы. Новосибирск, 1970. Вып. 39.
3. **Дрешер М.** Стратегические игры. Теория и приложения. М.: Сов. радио, 1964.
4. **Робинсон Дж.** Итеративный метод решения игр // Матричные игры. М.: ГИФМЛ, 1961.
5. **Беленький В. З., Волконский В. А., Иванков С. А. и др.** Итеративные методы в теории игр и программировании. М.: Наука, 1974.
6. **Гейл Д., Кун Х. У., Таккер А. У.** О симметричных играх // Матричные игры. М.: ГИФМЛ, 1961.
7. **Шапиро Г. Н.** Замечания о вычислительном методе в теории игр // Там же.

Институт физики полупроводников СО РАН,
E-mail: khor@isp.nsc.ru

Поступила в редакцию
1 апреля 1999 г.