АВТОМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА. – 2004. – No.5. – С.55-61.

ПОДХОД К ОПТИМАЛЬНОМУ НАЗНАЧЕНИЮ ЗАДАНИЙ В РАСПРЕДЕЛЕННОЙ СИСТЕМЕ

Р.М. АЛГУЛИЕВ, *доктор технических наук*
Р.М. АЛЫГУЛИЕВ, *кандидат физико-математических наук*
Р.К. АЛЕКПЕРОВ, *заведующий отделом*

Институт информационных технологий
Национальной Академии наук Азербайджана
ул. Ф. Агаева 9, AZ-1141 Баку, Азербайджан

В данной статье на основе использования метода математического программирования предложена методика оптимального назначения заданий в распределенной системе, учитывающая загруженность компьютеров и межкомпьютерных связей, с целью минимизации времени выполнения заданий.

Ключевые слова: распределенная система, оптимальное назначение заданий, математическое программирование, нейронная сеть

1. ВВЕДЕНИЕ

При решении задач большой размерности многопроцессорные вычислительные системы являются мощными средствами. Однако, при реализации таких задач в многопроцессорных системах существует проблема эффективного распределения заданий между процессорами, поскольку на производительность системы влияет два конфликтующие факторы: равномерная загрузка и межпроцессорные взаимодействия. Равномерная загрузка является ключевым фактором в достижении высокой параллельной эффективности, особенно в системах с большим числом процессоров. При этом число заданий, предназначенных для каждого процессора, должно определяться так, чтобы время выполнения было минимальным. Для параллельного распределения продолжительность выполнения заданий определяется как максимум всего времени, необходимого процессорам для завершения работы. Отсюда следует, что для минимизации времени выполнения заданий нужно увеличить число процессоров. Однако, как показывает закон Амдала [1], производительность системы перестает расти пропорционально числу используемых процессоров. Оказывается, этот эффект вызван насыщением межпроцессорных взаимодействий. Следовательно, при распределении заданий в микропроцессорной системе необходимо одновременно учитывать эти конфликтующие факторы.

Известны работы [2-6], посвященные решению данной проблемы. Однако, в этих работах задача решается при некоторых допущениях: либо межпроцессорные взаимодействия не учитываются, и при этом эффективность системы получается за счет равномерной загрузки процессо-

ров; либо на каждый процессор заранее распределяется равное число заданий и эффективность получается за счет минимизации межпроцессорных взаимодействий; либо система считается однородной.

В данной статье с целью минимизации времени выполнения заданий предлагается методика эффективного их распределения в системе с учетом загруженности компьютеров и межкомпьютерных взаимодействий.

2. ЗАДАЧА МИНИМИЗАЦИИ ВРЕМЕНИ ВЫПОЛНЕНИЯ ЗАДАНИЙ

Пусть заданы m неделимых заданий и распределенная система, состоящая из n компьютеров. Предполагается, что задания по объему вычислений, а компьютеры по производительности разные.

Введем следующие обозначения:

t_{ij} – продолжительность выполнения i-го задания в j-м компьютере $(i = 1,...,m;\ \ j = 1,...,n)$,

w_{kl} – вес связи между компьютерами k и l $(k,l = 1,...,n)$,

a_{pq} – число связей между заданиями p и q $(p,q = 1,...,m)$,

v_i – память, необходимая для решения i-го задания,

V_j – память j-го компьютера, причем $V_j \geq v_i$.

Не нарушая общности, будем считать, что число заданий больше или равно числу компьютеров $m \geq n$. В противном случае, введем дополнительные фиктивные задания $m+1,\ \ m+2,...,n$, положив для них $t_{ij} = 0$ при $i \geq m+1$ и $a_{pq} = 0$ при $p \geq m+1$ или $q \geq m+1$.

Нашей целью является назначение каждого из m заданий одному из компьютеров таким образом, чтобы общая продолжительность выполнения заданий с учетом загруженности компьютеров и межкомпьютерных взаимодействий была минимальной. Известно, что каждое такое назначение представляет собой перестановку $(s_1, s_2, ..., s_m)$, составленную из чисел $(1,2,...,n)$, $s_j \in \{1,2,...,n\}$, $j = 1,...,m$. Здесь подразумевается, что если $m = n$, то $s_i \neq s_j$ при $i \neq j$, т.е. каждому компьютеру назначается только одно задание. А если $m > n$, то возможен случай $s_i = s_j$ при $i \neq j$, т.е. одному компьютеру могут назначаться несколько заданий.

В общем-то, существуют три подхода к решению данной проблемы: графо-теоретический, эвристический и подход по методу математического программирования [3,4].

В предлагаемой работе задача решается с использованием метода математического программирования.

2. СВЕДЕНИЕ К ЗАДАЧЕ МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Очевидно, что любое из назначений задания i компьютеру s_i описывается соответствием $i \rightarrow s_i$ $(i = 1,...,m)$. При этом для любого назначения мы имеем, во-первых, интервал времени, равный t_{is_i}, и во-вторых, время взаимосвязи между заданиями. Будем предполагать, что это время при назначении задания i компьютеру s_i и задания k компьютеру s_k равно произведению число связей a_{ik} между заданиями i и k на вес связи $w_{s_i s_k}$ между компьютерами s_i и s_k, т.е. составляет $a_{ik} w_{s_i s_k}$. Таким образом, задача сводится к нахождению перестановки $(s_1, s_2, ..., s_m)$, составленной из чисел $(1,2,...,n)$ и минимизирующей суммарное время

$$\sum_{i=1}^{m} t_{is_i} + \sum_{i=1}^{m}\sum_{k=1}^{m} a_{ik} w_{s_i s_k} . \qquad (2.1)$$

Введем переменное x_{ij}, равное 1, если i-ое задание назначено компьютеру j, или равное 0 в противном случае. Тогда формулу (2.1) можно записать в таком виде:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} t_{ij} x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n} a_{ik} w_{jl} x_{ij} x_{kl} \to \min . \qquad (2.2)$$

Отсюда видно, что если все задачи независимы, т.е. $a_{ik} = 0$, тогда задача минимизации (2.2) по всем перестановкам превращается в задачу о назначениях. Наоборот, если все времена t_{ij} одинаковы, тогда речь будет идти только о минимизации суммарного времени по взаимодействию, т.е. второго слагаемого (2.2).

Так как каждое из m заданий назначается только одному из компьютеров, то должно выполняться следующее условие:

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1,...,m , \qquad (2.3)$$

где

$$x_{ij} \in \{0,1\}, \qquad \forall i, j. \qquad (2.4)$$

С другой стороны, чтобы память компьютеров не была перегружена, должно выполняться следующее условие:

$$\sum_{i=1}^{m} x_{ij} v_i \le V_j , \qquad j = 1,...,n. \qquad (2.5)$$

Таким образом, поставленная задача сведена к задаче целочисленного программирования (2.2)-(2.5).

Для дальнейших целей задачу (2.2) записываем в развернутом виде:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} t_{ij} x_{ij} + \sum_{j=1}^{n}\sum_{i=1}^{m}\sum_{k=1}^{m} a_{ik} w_{jj} x_{ij} x_{kj} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{\substack{l=1 \\ l\neq j}}^{n} a_{ik} w_{jl} x_{ij} x_{kl} \to \min . \qquad (2.6)$$

Здесь более наглядно отражается суть поставленной задачи. А именно, первые два слагаемых выражают загруженность компьютеров, а третье слагаемое – межкомпьютерные взаимодействия.

Теперь допустим, что компьютеры равны по производительности, а задания – по объему вычисления. В таком случае очевидно, что $t_{ij} = t_0$ для всех i, j. Кроме того, считаем, что $w_{kl} = w_0$ для всех k, l.

При сделанных допущениях следует, что равномерную загруженность можно получить за счет одинаковых чисел заданий, распределенных на каждый компьютер:

$$\begin{cases} \sum_{i=1}^{m} x_{ij} = m_j, & j = 1,...,n; \\ \sum_{j=1}^{n} m_j = m; \\ m_1 = m_2 = ... = m_n \approx \left[\frac{m}{n}\right]. \end{cases} \tag{2.7}$$

Учитывая вышеизложенное, приходим к выводу о том, что первые два слагаемых в задаче (2.6) можно опустить, заменяя их с условием (2.7):

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{m} \sum_{\substack{l=1 \\ l \neq j}}^{n} a_{ik} x_{ij} x_{kl} \to \min. \tag{2.8}$$

Таким образом, получим задачу целочисленного программирования (2.8), (2.3)-(2.5) и (2.7), минимизирующую межкомпьютерные взаимодействия с учетом равномерной загруженности компьютеров. Эта задача, если не учитывать условие (2.5) (это – существенное условие), полностью соответствует задаче целочисленного программирования, полученной в работе [5].

3. РЕАЛИЗАЦИЯ ЗАДАЧИ ЦЕЛОЧИСЛЕННОГО ПРОГРАММИРОВАНИЯ НА НЕЙРОННОЙ СЕТИ

В общем случае задачи целочисленного программирования являются *NP*-полными. Известны алгоритмы их решения, характеризующиеся полиномиальной трудоемкостью. Для многих таких задач пока нет убедительных доводов в пользу существования алгоритмов их решения, в силу чего все они отнесены к классам *NP*-полных. Достаточно часто на практике решение таких задач требует недопустимых затрат времени и вычислительных ресурсов. Сократить время решения позволяет использование нейронных сетей с обратными связями, неотъемлемым качеством которых является определение состояния с минимальным уровнем энергии сети. При решении этих задач на нейронных сетях с обратными связями основная трудность заключается в конструировании энергетической функции сети. Прежде чем перейти к конструированию этой функции, введением следующего обозначения

$$b_{ijkl} = \begin{cases} t_{ij} + a_{ii} w_{jj}, & \text{если} \quad i = k \quad \text{и} \quad j = l; \\ a_{ik} w_{jl}, & \text{если} \quad i \neq k \quad \text{и} \quad j \neq l, \end{cases} \tag{3.1}$$

задачу (2.2) перепишем в более компактном виде:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{m} \sum_{l=1}^{n} b_{ijkl} x_{ij} x_{kl} \to \min. \tag{3.2}$$

Теперь переходим к конструированию энергетической функции сети.

Согласно [7], конструируемую энергетическую функцию следует строить таким образом, чтобы она обеспечивала и задачу оптимизации, и выполнение ограничений.

Исходя из этого, компонент, обеспечивающий задачу оптимизации (3.2), строится в следующем виде:

$$E_1 = -\frac{\alpha_1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n} b_{ijkl}\, y_{ij} y_{kl}\,, \tag{3.3}$$

а компонент, обеспечивающий выполнения ограничений (2.3)-(2.5), строится в таком виде:

$$E_2 = \frac{\alpha_2}{2}\sum_{i=1}^{m}\left(\sum_{j=1}^{n} y_{ij} - 1\right)^2 + \frac{\alpha_3}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij}(1 - y_{ij}) +$$

$$+\frac{\alpha_4}{2}\left(\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij} - m\right)^2 + \frac{\alpha_5}{2}\sum_{j=1}^{n} f^2\left(\sum_{i=1}^{m} y_{ij} v_i - V_j\right), \tag{3.4}$$

где $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5$ – положительные константы, y_{ij} – выходной сигнал ij-го нейрона, соответствующий переменной x_{ij}, а $f(t) = t + |t|$ – функция, которая обладает свойством $f^2(t) = 2tf(t)$.

Первое слагаемое в (3.4) соответствует тому ограничению, что каждая строка матрицы Y содержит не более одной единицы, второе слагаемое соответствует бинарности переменных y_{ij}, третье слагаемое соответствует тому ограничению, что в матрице Y содержится ровно m единиц, наконец, последнее слагаемое соответствует ограничению (2.5). Отсюда следует, что при удовлетворении условий (2.3)–(2.5), компонент E_2 принимает свое минимальное, равное нулю, значение.

Суммируя (3.3) и (3.4), после несложных преобразований, приходим к следующему виду энергетической функции нейронной сети:

$$E = E_1 + E_2 = -\frac{\alpha_1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n} b_{ijkl}\, y_{ij}\, y_{kl} + \frac{\alpha_2}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n}\delta_{ik}\, y_{ij}\, y_{kl} -$$

$$-\frac{\alpha_3}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n}\delta_{ik}\,\delta_{jl}\, y_{ij}\, y_{kl} + \frac{\alpha_4}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n} y_{ij}\, y_{kl} -$$

$$-\alpha_2\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij} + \frac{\alpha_3}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij} - m\alpha_4\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij} + \frac{\alpha_2}{2}m + \frac{\alpha_4}{2}m^2 +$$

$$+\alpha_5\sum_{j=1}^{n}\left(\sum_{i=1}^{m} y_{ij} v_i - V_j\right) f\left(\sum_{i=1}^{m} y_{ij} v_i - V_j\right), \tag{3.5}$$

где δ_{ij} – символ Кронекера.

В последнем выражении слагаемые, не зависящие от состояния нейронной сети y_{ij} можно исключить.

Канонический вид энергетической функции сети, соответствующий задаче (3.2), записывается в таком виде:

$$E_c = -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n} W_{ijkl}\, y_{ij}\, y_{kl} + \sum_{i=1}^{m}\sum_{j=1}^{n} T_{ij}\, y_{ij}\,, \tag{3.6}$$

где W_{ijkl} – синаптический вес между входом ij-го нейрона и выходом kl-го, T_{ij} -порог ij-го нейрона.

Сопоставляя выражения (3.5) и (3.6) и приравняв коэффициенты их линейных и квадратичных составляющих, находим параметры нейронной сети:

$$\begin{cases} W_{ijkl} = \alpha_1 b_{ijkl} - \alpha_2 \delta_{ik} + \alpha_3 \delta_{ik} \delta_{jl} - \alpha_4 \\ T_{ij} = -\alpha_2 + \dfrac{\alpha_3}{2} - m\alpha_4 + \alpha_5 v_i, \end{cases} \tag{3.7}$$

где $i,k = 1,...,m;\quad j,l = 1,...,n.$

Теперь вычислим параметры нейронной сети, решающей задачу (2.8), (2.3), (2.4) и (2.7).

Перепишем задачу (2.8) в следующем виде:

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n} a_{ik}(1-\delta_{jl})x_{ij}x_{kl} \to \min . \tag{3.8}$$

Тогда энергетическая функция сети задачи (3.8), (2.3), (2.4) и (2.7) будет записана в таком виде:

$$E = -\frac{\beta_1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n} a_{ik}(1-\delta_{jl})y_{ij}y_{kl} + \frac{\beta_2}{2}\sum_{i=1}^{m}\left(\sum_{j=1}^{n} y_{ij} - 1\right)^2 +$$

$$+\frac{\beta_3}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij}(1-y_{ij}) + \frac{\beta_4}{2}\left(\sum_{i=1}^{m}\sum_{j=1}^{n} y_{ij} - m\right)^2 + \frac{\beta_5}{2}\sum_{j=1}^{n}\left(\sum_{i=1}^{m} y_{ij} - m_j\right)^2, \tag{3.9}$$

откуда сравнивая с (3.6) находим параметры нейронной сети:

$$\begin{cases} W_{ijkl} = \beta_1 a_{ik}(1-\delta_{jl}) - \beta_2\delta_{ik} + \beta_3\delta_{ik}\delta_{jl} - \beta_4 - \beta_5\delta_{jl} \\ T_{ij} = -\beta_2 + \dfrac{\beta_3}{2} - m\beta_4 - m_j\beta_5, \end{cases} \tag{3.10}$$

где $\beta_1, \beta_2, \beta_3, \beta_4, \beta_5$ – положительные константы и $i,k = 1,...,m;\quad j,l = 1,...,n.$

Замечание. Так как задачу целочисленного программирования за определенное время можно реализовать на нейронной сети, то с целью повышения надежности системы число компьютеров можно взять переменным $n \in [n_1, n_2]$.

4. ЗАКЛЮЧЕНИЕ

Предложена методика эффективного распределения заданий между компьютерами в распределенной системе. Эта методика, учитывающая загруженность компьютеров и межкомпьютерные связи, позволяет минимизировать время выполнения заданий. Математическая реализация методики опирается на задачу целочисленного программирования. Представлена нейросетевая реализация задачи целочисленного программирования, позволяющая сократить время ее решения.

СПИСОК ЛИТЕРАТУРЫ

[1] *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. СПб.: Изд. "БХВ-Петербург", 2002.

[2] *Лобков С.Н., Ревин С.Б.* К вопросу оценки одного метода оптимального распределения заданий в многопроцессорных вычислительных системах.// Изв. вузов Северо-Кавказского региона. Естественные науки. 1998. № 2.

[3] *Gao H., Schmidt A., Gupta A., Luksch P.* Load balancing for spatial-grid-based parallel numeric simulations on clusters of SMPs. // Proc. of 11th Euromicro Conference on Parallel, Distributed and Network based Processing (PDP2003). February 5-7. 2003. Genoa. Italy.

[4] *Gao H., Schmidt A., Gupta A., Luksch P.* A graph-matching based intra-node task assignment methodology for SMP clusters. // Proc. of 7th World Multiconference on Systemics, Cybernetics and Informatics (SCI2003). July 27-30. 2003. Orlando. Florida. USA.

[5] *Лобков С.Н., Фатхи В.А.* Подход к минимизации межпроцессорных взаимодействий при организации вычислительного процесса в многопроцессорных системах.// Изв. РАН. ТиСУ. 2003. № 6.

[6] http://www-users.cs.umn.edu/~karypis/publications/index.html.

[7] Нейроматематика. Книга 6. Учебное пособие для вузов. Под общей редакцией А.И.

Рукопись получена 22.03.2004