УДК 681.3.06:65.012.122

ОРГАНИЗАЦИЯ МУЛЬТИОБРАБОТКИ ПРОБЛЕМНЫХ ЗАДАЧ В СИСТЕМЕ ПРОЕКТИРОВАНИЯ

ВИШНЯКОВ В. А.

(Минск)

Разработан метод предварительного распределения информационно-связанных задач между процессорами однородной вычислительной системы (ВС) (разделами операционной системы ОС ЕС) с учетом степени связанности. Приведена организация обработки связанных программных модулей задачи внутри раздела операционной системы в автоматическом и диалоговом режимах.

1. Введение

В настоящее время многообразие задач, программных модулей, данных затрудняет организацию вычислительного процесса в системах проектирования. В то же время создание новых задач связано с большими временными и материальными затратами. Все это требует использования диалогового управляющего комплекса для оптимизации хода вычислительного процесса и потоков данных в системах [1]. При этом большое значение имеет получение близкого к оптимальному распределения ресурсов вычислительной системы, в первую очередь, времени процессора и памяти между решаемыми программами [2]. Эта задача в зависимости от характера решения, рода вычислительных ресурсов представляет собой ту или иную модификацию проблемы расписания [3]. Особый интерес представляет случай, когда распределяются информационно-связанные задачи, поскольку они составляют основную часть задач, решаемых в современных системах управления и проектирования. В то же время входящие в отдельную задачу программные модули находятся в той или иной информационной связи. Чаще всего программные модули, входящие в состав задачи, образуют древовидные структуры, а задачи, образующие подсистему,— сетевые, хотя возможно и обратное. Существующие ЭВМ серии ЕС обеспечены мультипрограммными операционными системами ОС ЕС, распределение в которых как между разделами, так и внутри раздела осуществляется по классам и приоритетам [4]. Имеется ряд эффективных алгоритмов распределения связанных задач [2, 3] и систем для их реализации [5], однако при этом не учитывается степень связанности, а физическое представление структуры связей выполнено неоптимально по объему памяти и времени обработки. В настоящей статье рассматриваются математические модели для организации распределения связанных задач в мультипрограммной операционной системе ОС ЕС в режиме разделения времени (мультипроцессорной ВС) и обработки программных модулей внутри раздела (процессора ВС).

2. Постановка задачи

Определим иерархию программных средств следующим образом: пакет программ, подсистема, задача, программный модуль. Последний имеет один вход и выход, сравнительно невелик, обладает единственной функцией [6]. Пусть на основе N-го пакета программ в системе проектирования решается M подсистем. В состав j-й подсистемы ($j=\overline{1, M}$) включается i задач ($i=\overline{1, n}$), которые в общем случае являются информационно-свя-

занными. Каждая из решаемых задач j подсистемы в свою очередь может включать l ($l=\overline{1, k}$) связанных программных модулей. Для математического описания данной структуры используем двухступенчатую иерархию. Структуру j проблемы ($j=\overline{1, M}$) интерпретируем направленным графом $G_j=(U_j, V_j)$, где множеству вершин $U=\{u_1, \ldots, u_n\}$ поставим в однозначное соответствие множество задач $A=\{a_1, \ldots, a_n\}$, а множеству дуг $V=\{v_1, \ldots, v_p\}$ — информационные связи между задачами из A. На второй ступени иерархии задачу $a_i \in A$ ($i=\overline{1, n}$) интерпретируем графом $\Gamma_i=(P, W)$, где множеству вершин $P=\{p_1, \ldots, p_k\}$ поставим в соответствие множество программных модулей $B=\{b_1, \ldots, b_k\}$, а множеству дуг W — информационные связи, в которые вступают между собой модули из B. Необходимо распределить задачи из A с учетом их степени связанности по разделам операционной системы (процессорам ВС), а программные модули из B необходимо распределять и обрабатывать внутри раздела (процессора ВС).

3. Разработка модели диспетчирования

Обычно граф $G=(U, V)$ кодируется матрицей следования $S=[s_{ij}]$ с элементом $s_{ij}=1$, если вершины i, j инцидентны, в противном случае $s_{ij}=0$ [2, 7]. По матрице смежности осуществляется близкое к оптимальному планирование выполнения заданного набора задач [2]. Однако это кодирование является неэффективным в силу большой плотности нулевых элементов и, следовательно, значительного времени обработки матрицы и занимаемой при этом памяти. Рассмотрим применение списочного кодирования [8] графа для использования при построении модели распределения задач. Введем кортежи C_k^1 и C_k^2, причем $C_k^1=\{c_k^1 \mid k=\overline{1, n+p}\}$, где n — число вершин, а p — число дуг графа $G=(U, V)$. Элементы C_k^1 определяют номера вершин переходов для каждой вершины графа G следующим образом: если $\operatorname{sign} c_k^1=(+1)$, то из этой вершины возможны переходы в те вершины, для которых $\operatorname{sign} c_l=(-1)$, $l=k+1, \ldots, r$, где r — порядковый номер ближайшего к C_k^1 положительного элемента кортежа C_k^1. Элементы C_k^2 — степени вершин графа для выходящих дуг. Пусть имеется множество $T=\{t_1, \ldots, t_n\}$, элементы которого — трудоемкости каждой задачи из A.

Для учета степени связанности задач j подсистемы введем векторы K_{1i} и K_{2i}, элементы которых определяются так:

$$K_{pi}=\frac{\sum_{j \in \psi_1} K_{pj}}{l_j}, \qquad p=1, 2, \tag{1}$$

где ψ_1 — подмножество номеров вершин, для которых есть переходы в i-ю вершину; K_{pj} — мощность потока в j вершине; l_j — степень j вершины графа G для выходящих дуг.

Полагаем, что мощность потока в начальных вершинах равна единице, а при разветвлении в j-й вершине ($l_j>1$) мощности выходящих дуг одинаковы и равны сумме мощностей входящих дуг. Элементы вектора K_{1i} определяются для прямого графа G, а элементы K_{2j} — для обратного графа G^*. Значение K_{1i} для тупиковых и изолированных вершин равно нулю.

Рассмотрим последовательность распределения для однородных ВС [2] с учетом представления графа G кортежами C_k^1 и C_k^2 и введения векторов K_{1i}, K_{2i}. Пусть множество $\tau^v=\{\tau_1^v, \ldots, \tau_m^v\}$ определяет время занятости раздела ОС, множество $D^v=\{d_1^v, \ldots, d_e^v\}$ — номера задач из A, назначенных последними, $R^v=\{r_1^v, \ldots, r_p^v\}$ — множество задач, которые в t_i момент времени могут обрабатываться, $E^v=\{e_1^v, \ldots, e_r^v\}$ — номера разделов, обладающих минимальным временем занятости, $F^v=\{f_1^v, \ldots, f_p^v\}$ — подмножество номеров задач из E, назначенных последними. Решение бу-

дет представляться в виде класса множеств $\bigcup_{i=1}^{m} L_i$, в котором каждое множество L_i $(i=\overline{1, m})$ соответствует одному разделу. Тогда имеем следующую последовательность распределений.

1. $\tau_i^v=0$ $(i=\overline{1, m})$, номер шага $v=0$, $d_i=0$.
2. Введем элементы множества трудоемкостей $T=\{t_i \mid i=\overline{1, n}\}$.
3. Вычислим элементы векторов K_{1i} и K_{2i} по выражению (1).
4. Вычислим элементы множеств $T^*=\{t_1^*, \ldots, t_n^*\}$, где $t_i^*=k_{1i}\cdot k_{2i}\cdot t_i$; переход к п. 9.
5. Находим $\tau_E^v=\min(\tau_1, \ldots, \tau_n)$ и множество разделов E, обладающих этим временем занятости.
6. Находим подмножество $F^v\subset D^v$. В соответствующих позициях множества D записываем нули.
7. Если $F^v=\phi$, $R^v=\phi$, переход к п. 10, если $F^v\neq\phi$ — к п. 13; $R^v\neq\phi$ — к п. 8.
8. По номерам задач, образующим F^v, из кортежа C_k^1 определим номера задач, к которым возможны переходы. Из соответствующих элементов кортежа C_k^2, соответствующих номерам задач из F, вычитаем единицу.
9. Находим элементы множества R^v (им соответствуют нули в кортеже C_k^2). Из R^v удаляем номера задач, отмеченные в D. Если $R^v\neq\phi$, переход к п. 12, иначе — к п. 10.
10. Находим τ_j^v — значение времени занятости раздела, минимально превосходящего τ_E^v.
11. В строке L_i записываем простой, равный $\tau_j-\tau_E$. Обнуляем позиции E^v в множестве D^v. Переход к п. 5.
12. В множестве R^v задачи размещаются в порядке невозрастания их t_i^*. Если $t_i^*=t_j^*\neq 0$, то задачи i и j располагаются в порядке неубывания их времени выполнения t_i и t_j. Если $t_i=t_j=0$, то задачи располагаются в порядке невозрастания времени их выполнения.
13. $v=v+1$. Первая задача $\sigma_1\in R^v$ назначается первому разделу из E, с исключением ее из R и записью номера и времени выполнения в множество $L_i\in L$. Полагаем $\tau_E=\tau_E+\tau_{r_1}$.
14. Если $v\leqslant m$, переход к п. 5, иначе распределение заканчивается.

Таким образом, после выполнения вышеуказанной последовательности вычислений определится таблица распределения задач подсистемы по разделам операционной системы.

4. Метод обработки задачи в разделе ОС

Пусть граф $F_i=(P_i, W_i)$ отражает структуру i-й задачи $(i=\overline{1, k})$. Определим степени δ_i выходящих дуг для всех вершин $\Gamma=(P, W)$. Выделим подграф $\Gamma_2=(P_2, W_2)$, такой, что $\delta_i(P_2)=0$, $W_2=0$. Получим подграф $\Gamma_1=(P_1, W_1)$, где $\delta_i(P_1)\geqslant 1$, $W_1\neq 0$. Для каждой вершины $p_i\in P_1$ введем подмножество p_i^*, элементы которого определяют вершины, в которые можно перейти из p_i $(i=\overline{1, k-l}$, l — число тупиковых вершин). Пусть после выполнения программного модуля, соответствующего вершине p_i, вырабатывается условие $s_i\in S$, где S — множество условий, вырабатываемых при решении j задачи $(j=\overline{1, n})$. Кроме того, для каждой вершины $p_i\in P$ введем подмножество X_i логических условий, определяющих переход как в данную вершину, так и из нее в другие вершины. Для задачи тогда имеем $X=\bigcup_{i=1}^{k} x_i$. Таким образом, задано соответствие $q=(P_1, P^*, Q)$, где $Q=P_1\times p^*$. Для однозначного определения вершины $p_j\in P_i^*$ вершине $p_i\in P_1$ необходимо выполнение условия

$$X_i\cap X_j=s_i,$$

где s_i — условие перехода из p_i в p_j.

Рассмотрим последовательность обработки связанных программных модулей j-й задачи в j-м разделе ОС в двух режимах. В первом режиме график соответствия уточняется управляющей системой (режим автоматический), во втором — график соответствия выбирается оператором в процессе интерактивного взаимодействия (диалоговый режим).

1. Если выполняется i-й $(i=\overline{1, k})$ модуль задачи, получаем условия s_i. Определим правильность выработанного условия s_i. Если $s_i \in S$, то условие перехода верно. Если $s_i \notin S$, условие перехода выработано неверно. В автоматическом режиме модуль просчитывается снова, в диалоговом оператору выдается соответствующее сообщение, дальнейшая работа выполняется в зависимости от его решения (снять, повторить, продолжить и т. д.).

2. Определяется $p_i \in P_1$. Если $p_i \in P_1$, то находится P_j^*, в которую можно перейти. Если $p_i \notin P_1$, то выполнение задачи закончено.

3. Для модуля, соответствующего p_i, находим $p_i^* \in P$ из соответствия $q_i=(p_i, p_i^*, Q_i)$.

4. Анализируется режим работы в разделе: автоматический — переход к п. 6, диалоговый — к п. 5.

5. Оператору выдается подмножество P_i^*, и оно конкретизирует график соответствия, определяя $p_j \in P_i^*$, в которую необходимо перейти из p_i.

6. Определяется композиция соответствий $q(y)$ следующим образом: $P(y)=(P_i, X_i, Q \circ Y) Q \circ Y=P_i^* \times X_i$, $x_i \in X$, на основании которой находим множество условий переходов для вершины p_i.

7. Путем попарного пересечения множеств x_i, x_i^* находим такое p_l, для которого

$$x_i \cap x_l = s_i.$$

8. По значению l определяется в множестве p_i^* вершина p_l, в которую автоматически осуществляется переход из p_i.

9. Осуществляется переход к п. 1.

Рассмотренная модель может быть обобщена на случай, если программный модуль, соответствующий вершине p_i, вырабатывает подмножество условий $S_{ij}=\{s_{i1}, \ldots, s_{ir}\}$. Тогда вместо X_i имеем класс множеств условий $\bigcup\limits_{i=1} x_i$ и обработка задачи производится в вышеописанной последовательности.

5. Оценки

Рассмотренные модели распределения связанных задач между разделами операционной системы и обработки программных модулей внутри раздела реализованы в виде управляющих модулей. Апробирование на ЭВМ позволило получить статистику планирования связанных задач для m-раздельной (m-процессорной) системы проектирования, которая показала, что данный метод более оптимален, чем описанный в [2]. Для оценки эффективности метода можно использовать следующую оценку:

$$\theta = \left[\left(\sum_{i=1}^{m} t_i^* - \sum_{j \in T} z_j^*\right)(m-K_T)\right] \Big/ \left[\left(\sum_{i=1}^{m} t_i - \sum_{j \in T} z_i\right) \times \right.$$

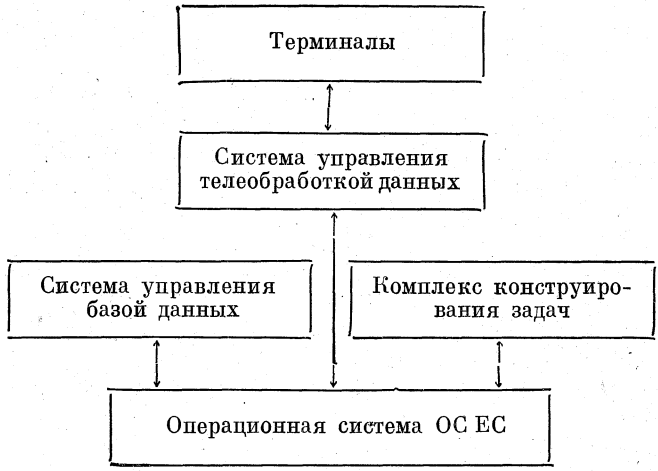
$$\left. \times (m-K_T-l)\right],$$

где z_j, z_j^* — трудоемкость и взвешенная трудоемкость j вершины графа $G(U, V)$; m — число всех вершин; T — множество тупиковых вершин; K_T — мощность множества T; l — число тупиковых вершин.

Для $\theta>1$ предложенный метод и метод, приведенный в [2], равноценны. При $\theta<1$ данный метод позволяет получить лучшее распределение. Для сравнения затрат памяти, необходимых при кодировании графов связанности, которыми интерпретируются задачи проблемы $G(U, V)$ и модули задачи $\Gamma(P, W)$, можно использовать оценку

$$Y=n^2-2n-p,$$

где n — число связей графа.



Если $Y>0$, рассматриваемое кодирование требует меньших затрат памяти, при $Y\leqslant 0$ (графов с малым числом вершин) меньших затрат памяти требует кодирование матрицей связанности.

6. Заключение

Предложенные принципы организации мультиобработки реализуются в ОС ЕС следующим образом. Программы планирования, реализующие метод распределения связанных задач по разделам операционной системы, подключаются к пакету прикладных программ КРОС, который позволяет изменить алгоритмы управления задачи в ОС ЕС, используя специальные методы доступа для работы с входными потоками заданий [9]. Обработка распределенных задач в разделах ОС ЕС в автоматическом или диалоговом режимах осуществляется в рамках системы с разделением времени, функционирующей в среде операционной системы. При этом способ управления вычислительным процессом в разделе реализуется посредством языка команд [9]. Программы, реализующие разработанные методы, входят в состав комплекса конструирования задач [10], который взаимодействует с системой управления базами данных, системой телеобработки данных и функционирует в операционной системе ОС ЕС 6.1 (рисунок). Основной задачей комплекса является реализация принципа универсальной обработки проблемных программных модулей по аналогии с базами данных, предназначенных для универсальной обработки данных [11]. Комплекс предназначен для проектирования подсистем АСУ.

ЛИТЕРАТУРА

1. *Глушков В. М.* Фундаментальные исследования и технология программирования.— Программирование, 1980, № 1, с. 3–13.
2. *Барский А. Б.* Планирование параллельных вычислительных процессов. М.: Машиностроение, 1980.
3. *Головкин Б. А.* Параллельная обработка информации, программирование, вычислительные методы, вычислительные системы.— Техническая кибернетика, 1979, № 2, с. 116–151.
4. Операционная система ОС ЕС / Под ред. Райкова Л. Д. М.: Статистика, 1980.
5. *Мамзелев И. А.* Архитектура параллельных вычислительных систем и тенденции их развития.— Зарубежная радиоэлектроника, 1980, № 11, с. 3–29.
6. *Хьюз Дж., Мичтом Дж.* Структурный подход к программированию. М.: Мир, 1980.
7. *Оре О.* Теория графов. М.: Мир, 1980.
8. *Ахо А., Хопкрофт Д., Ульман Д.* Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.
9. *Пржиялковский В. В., Ломов Ю. С.* Технические и программные средства единой системы ЭВМ. М.: Статистика, 1980.

10. *Вишняков В. А., Китайгора В. И.* Комплекс управления процессами в автоматизированной системе проектирования.– В кн.: Тез. докл. республиканской НТК «Проблемы применения современных радиофизических методов для повышения эффективности производства и автоматизации научных исследований». Минск: Высшая школа, ч. 2, с. 43–44.
11. *Мартин Дж.* Организация баз данных в вычислительных системах. М.: Мир, 1980.

Поступила в редакцию
10.VII.1981

MULTI-PROCESSING OF PROBLEM JOBS IN A DESIGN SYSTEM

VISHNYAKOV V. A.

A method for tentative allocation of data related jobs between processors of a uniform computing system (parts of an OS ES operating system) is developed which allows for the degree of connectivity. A sequence of processing related program modules of the job within a part of the operating system in automatic and dialog modes is described.