МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО
СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РСФСР

ВОРОНЕЖСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

МОДЕЛИРОВАНИЕ И ОПТИМИЗАЦИЯ СЛОЖНЫХ СИСТЕМ

(Сборник научных трудов)

Воронеж — 1982

Литература

I. Хилл Р. Математическая теория пластичности.-М.:ГИТТЛ,1956.

2. Ширшов А.А. Исследование пластического изгиба листа с учетом упрочнения.- Известия вузов. М.: Машиностроение, 1965, № 7.

3. Ширшов А.А. Пластический изгиб листа из анизотропного материала при больших деформациях. Известия вузов.-М.:Машиностроение, 1969, № 10.

4. Малинин Н.Н. Устойчивость двухосного пластического растяжения анизотропных листов и цилиндрических оболочек. - Известия АН СССР; Механика твердого тела. М.: 1971, № 2.

5. Куршин Л.М. Обзор работ по расчету трехслойных пластин и оболочек. - В кн.: Расчет пространственных конструкций. М.: Стройиздат, 1962, вып.7.

НЕКОТОРЫЕ ВАРИАНТЫ ЗАДАЧИ КОМПОНОВКИ ПРОГРАММНЫХ МОДУЛЕЙ

М.Ш.Левин

Производительность вычислительных систем существенно зависит от числа обменов между оперативной памятью (ОП) и внешней памятью (ВП) [1,2]. Современные программные комплексы представляют собой иерархические системы, состоящие из программных модулей [3]. В качестве моделей таких систем рассматриваются направленные графы, вершины которых взаимно-однозначно соответствуют программным модулям, а дуги указывают связи между вызывающими и вызываемыми модулями, при этом часто предполагается, что передача по управлению осуществляется с возвратом в вызываемый модуль [3,4]. В процессе функционирования программных комплексов имеют место ситуации, когда одни модули, находящиеся в ОП, вызывают для выполнения из ВП другие. За счет объединения программных модулей возможно снижение среднего числа вызовов из ВП.

В работе рассматривается задача компоновки программных модулей, входящих в некоторую систему, с целью минимизации среднего числа вызовов при ограничении на общий объем ОП, предоставляемый для работы программного комплекса. Модель системы предполагается древовидной.

Пусть задано направленное дерево $G=(A,\Gamma)$ с ориентацией дуг от корня к висячим вершинам, где A - множество вершин, взаимно-однозначно соответствующее множеству программных модулей системы, Γ - многозначное отображение A в A, характеризующее отношение вызова между программными модулями. Каждая вершина $a \in A$ имеет

вес $V(a) \geq 0$ (требуемый объем ОП). Дугу из a в $b (a, b \in A)$ будем обозначать (a, b), направление дуги указывает вершину, соответствующую вызываемому модулю. Обозначим корень дерева через a_0.

Определение I. Назовем упорядоченное множество вершин $L(a_i, a_\ell) = \langle a_i, \ldots, a_j, \ldots, a_\ell \rangle$ путем от a_i к a_ℓ, $a_i \in A (i = \overline{1, \ell})$, если $a_{j+1} \subset \Gamma a_j$ $(j = \overline{i, \ell-1})$. Величину $V(L(a_i, a_\ell)) = \sum_{i=1}^{\ell} V(a_i)$ назовем весом пути $L(a_i, a_\ell)$.

Будем говорить, что вершина b достижима из a или a предшествует b, если существует путь $L(a, b)$.

Определение 2. Назовем критическим путем дерева G путь $L^*(G) = L(a_0, a)$, где $a \in \{b \in A (\Gamma b = \phi)$, если $V(L(a_0, a)) = \max_{b} V(L(a_0, b))$. Величину $V(G) = V(L^*(G))$ назовем весом дерева G, $V(G) = 0$ при $A = \phi$.

Пусть $G_a = (A_a, \Gamma)$ – поддерево с корнем в $a \in A$ (в A_a кроме a входят все вершины из A, достижимые из a).

Определение 3. Назовем "хвостом" вершины $a \in A$ граф $(\{A_a / a\}, \Gamma)$, а весом "хвоста" вершины a – величину $V(a) = V(G_a) - V(a)$. Очевидно, что $\hat{V}(a) = \max_{b \in \Gamma a} V(G_b)$.

В зависимости от исходных данных при работе модуля $a \in A$ возможны n_a вариантов, каждый из которых характеризуется набором значений средних чисел вызовов модулей из множества Γa. Обозначим через a_i^b среднее число вызовов модуля $b \in \Gamma a$ модулем a при варианте i $(i = \overline{1, n_a})$. Пусть каждый вариант характеризуется вероятностью P_i $(\sum_{i=1}^{n_a} P_i = 1)$. Тогда среднее число вызовов модуля b модулем a равно $a^b = \sum_{i=1}^{n_a} P_i a_i^b$. Определим среднее число вызовов программного модуля a при функционировании системы следующим рекурентным соотношением: $d(a) = \sum_{c \in \Gamma^{-1} a} d_a(c) d(c)$, $d(a_0) = 1$. Тогда среднее число вызовов модуля b модулем a при функционировании системы (вес дуги от a к b) равно $W(a, b) = d(a) \cdot d_b(a)$.

Предполагается, что при вызове модуля b модулем a последний находится в ОП, а модуль b загружается в ОП из ВП (адрес загрузки $\forall b \in A : \gamma(b) = \max_{a \in \Gamma^{-1} b} (\gamma(a) + V(a)), \gamma(a_0) = \gamma_0$).

Вес дерева G определяет требуемый для функционирования программного комплекса объем ОП, а среднее число вызовов программных модулей из ВП равно

$$W(G) = \sum_{\substack{a, b \in A \\ b \in \Gamma a}} W(a, b) \qquad (I)$$

Заметим, что вершина $a \in A$, в свою очередь, может представлять собой совокупность взаимосвязанных модулей, т.е. взвешенный граф, вес a равен весу этого графа.

Определение 4. Назовем дерево G допустимым, если $V(G) \leq V$, где V - ограничение на объем ОП, предоставляемый системе.

Объединение (склеивание) программных модулей a и v позволяет заменить вызов модуля b модулем a из ВП на переход к модулю b внутри ОП. Это снижает значение (I) на $w(a,v)$, но увеличивает вес графа на величину $\Delta(a,b) \in [0, V(b)]$.

Пусть $|A| = n$. Введем множество дуг $\bar{V} = \{u_1, \ldots, u_i, \ldots, u_{n-1}\}$. Обозначим через $\bar{G} = (\bar{A}, \bar{\Gamma})$ граф, полученный из G склеиванием вершин $a, b, b \in \Gamma a$, т.е. в G вместо вершин a и v входит вершина $\bar{J} = \bar{J}(a,b)$ с весом $V(\bar{J}(a,b)) = V(a) + V(b)$, $\bar{\Gamma}\bar{J} = \{\Gamma a \cup \Gamma b\} / b$. При этом множество дуг в $\bar{G}$ $\bar{V} = V / (a,b)$. Очевидно, что операция преобразования графа при склеивании вершин ассоциативна, т.е. порядок склеивания (стягивания дуг) несущественен. Введем переменную $x_i (i = \overline{1, n-1})$, которая равна I, если дуга u_i стягивается, и 0 - в противном случае. Тогда множество пар склеиваемых вершин можно задать булевым вектором $\lambda = (\lambda_1, \ldots, \lambda_i, \ldots, \lambda_{n-1})$. Перейдем от (I) к критерию

$$W(x) = \sum_{i=1}^{n-1} x_i W(u_i). \qquad (2)$$

С учетом стягивания дуг для любого $a \in \hat{A}$ вес вершины и вес "хвоста" можно рассматривать как функцию вектора x : $V(a,x), \hat{V}(a,x)$. Таким образом, задача состоит в определении вектора x, который максимизирует (2) при выполнении условия допустимости графа

$$V(a_0, x) + \hat{V}(a_0, x) \leq V. \qquad (3)$$

В случае когда система состоит из основного сегмента и вызываемых подпрограмм, задача компоновки не проще задачи о рюкзаке, которая принадлежит к классу универсальных переборных проблем [5]. Для таких постановок целесообразно искать эффективные (с полиномиальной трудоемкостью) приближенные алгоритмы, гарантирующие заданную относительную погрешность целевой функции [6]. Для простейших случаев сформулированной задачи существуют эффективные точные методы решения.

Для некоторых вариантов задачи компоновки программных модулей может не существовать эффективного приближенного алгоритма, гарантирующего заданную относительную погрешность целевой функции, и в этом случае необходима дополнительная плата за быстроту решения. Определим приближенное решение задачи следующим образом. Пусть $\varepsilon, \delta \in [0,1]$, x^* - точное решение задачи (2)-(3). Введем для некоторого x функци

относительной ошибки целевой функции $\alpha(x)$ и ресурса $\beta(x)$:

$$\alpha(x)=\begin{cases}0, W(x)\geq W(x^*),\\ \dfrac{W(x^*)-W(x)}{W(x^*)}, W(x)<W(x^*),\end{cases} \tag{4}$$

$$\beta(x)=\begin{cases}0, V(x)<V,\\ \dfrac{V(x)-V}{V}, V(x)>V\end{cases} \tag{5}$$

где $V(x)=V(a_0,x)+V(a_0,x), W(x^*)\neq 0$.

Задача состоит в определении при фиксированных ε и δ такого x^0, что

$$\alpha(x^0)\leq\varepsilon, \tag{6}$$

$$\beta(x^0)\leq\delta. \tag{7}$$

Рассмотрим простейшие случаи поставленной задачи. когда G представляет собой двухуровневое дерево. Пусть $\Gamma a_0=\{a_1,\dots,a_i,\dots a_m\}$. Обозначим $\forall a_i\in\Gamma a_0$ дугу (a_0,a_i) через u_i, $W(u_i)$ через W_i. В этом случае $V(a_0,x)=V(a_0)+\sum_{i=1}^{m}x_iV(a_i)$, $V(a_0,x)=\max_{1\leq i\leq m}[(1-x_i)V(a_i)]$ и задача компоновки будет иметь следующий вид:

максимизировать $W(x)=\sum x_iW_i$ при условиях

$$V(a_0)+\sum_{i=1}^{m}x_iV(a_i)+\max_{1\leq i\leq m}[(1-x_i)V(a_i)]\leq V. \tag{8}$$

Для этой задачи существует эффективный приближенный алгоритм, гарантирующий заданную относительную погрешность целевой функции в соответствии с (6). При этом требуется предварительное упорядочение вершин множества Γa_0 в порядке невозрастания весов. Алгоритм является несложной модификацией "алгоритма разбиения на интервалы" для решения задачи о рюкзаке [6] и имеет аналогичную оценку трудоемкости и памяти $O\left(\frac{m^2}{\varepsilon}\right)$.

При разработке реальных систем часто имеют место случаи, когда некоторые характеристики программных модулей практически равны. Пусть, например, в задаче (8) веса вершин из множества Γa_0 равны V. Тогда существует очевидный точный алгоритм с трудоемкостью $O(m\log m)$. Необходимо упорядочить элементы Γa_0 в порядке невозрастания весов соответствующих дуг и последовательно присоединять их к корневой вершине, проверяя допустимость получающегося графа. В случае когда равны веса дуг, направленных в вершины Γa_0, эффективный точный алгоритм аналогичен.

Рассмотрим постановку задачи компоновки программных модулей в случае трехуровневого дерева. Обозначим $\forall a \in A/a_0$: $W(a)$ – вес дуги, направленной в a, $x(a)$ – булева переменная, равная I, если вершина a объединяется с предшествующей, и 0 – в противном случае. Тогда задача имеет вид:

аксимизировать $\sum x(a)W(a)$ при условиях

$$V(a_0)+\sum_{a\in\Gamma a_0} x(a)[V(a)+\sum_{b\in\Gamma a} x(b)V(b)]+\max\{[1-x(a)][V(a)+ \tag{9}$$
$$+\sum_{b\in\Gamma a} x(b)V(b)]+\max_{b\in\Gamma a}\{(1-x(b))V(b)]\}\le \hat{b}.$$

В случае, когда объемы модулей множества A/a_0 равны V, ограничения в (9) значительно упрощаются:

$$\sum_{a\in\Gamma a_0} x(a)[y(a)+1]+\max_{a\in\Gamma a_0}\{[1-x(a)][y(a)+1]+$$
$$+\operatorname{sign}[m(a)-y(a)]\}\le \frac{V-V(a_0)}{V},$$

где $\forall a\in\Gamma a_0$ $y(a)=\sum_{b\in\Gamma a} x(b)$, $m(a)=|\Gamma a|$

От этой задачи можно перейти к эквивалентной. Введем $\forall a\in\Gamma a_0$ переменную $z(y(a))$, которая равна I, если a присоединяются последователей (в порядке невозрастания весов направленных в них дуг). Таким образом, $\forall a\in\Gamma a_0$ имеется $m(a)$ вариантов, каждый из которых характеризуется числом присоединяемых последователей $y(a)$, $0\le y(a)\le m(a)$, и величиной $W(y(a))$, равной сумме весов стянутых при этом дуг. Тогда эквивалентная задача имеет вид:

максимизировать $\sum_{a\in\Gamma a_0}[x(a)W(a)+\sum_{y(a)=0}^{m(a)} z(y(a))W(a)]$ при условиях

$$\sum_{a\in\Gamma a_0} x(a)\sum_{y(a)=0}^{m(a)} z(y(a))[y(a)+1]\le b, \max_{a\in\Gamma a_0}\{[1-x(a)]\cdot z(y(a))\cdot[y(a)+1]+$$
$$+\operatorname{sign}[m(a)-y(a)]\}\le\left[\frac{V-V(a_0)}{V}\right]-b,$$
$$\sum_{a\in\Gamma a_0} x(a)\le 1,\ \sum_{y(a)=0}^{m(a)} z(y(a))=1 \quad u\ x(a)\le z(y(a))\ \forall a\in\Gamma a_0. \tag{10}$$

Для решения задачи (10) при фиксированном целом b $(0\le b\le \left[\frac{V-V(a_0)}{V}\right])$ существует эффективный приближенный алгоритм, гарантирующий заданную относительную погрешность целевой функции в соответствии с (6). Этот алгоритм является модификацией алгоритма решения задачи "лестничного рюкзака" [6] и имеет аналогичную оценку трудоемкости и памяти $O\left(\frac{m(a_0)}{\varepsilon}\sum_{a\in\Gamma a_0} m(a)\right)\le O\left(\frac{n^2}{\varepsilon}\right)$. Для решения задачи (10) необходимо решить ее для всех значений b и выбрать решение с наибольшим значением целевой функции. При этом оценка трудоемкости и памяти $O\left(\frac{n^3}{\varepsilon}\right)$.

Введенная модель программного комплекса может быть распространена и на более общие случаи, например, на корневой ациклический граф. При этом дополнительно, в качестве возможных вариантов склеивания вершин, необходимо рассматривать множество доминаторных дуг, связывающих некоторую вершину и ее непосредственного доминатора [5].

Литература

1. Липаев В.В. Распределение ресурсов в вычислительных системах.- М.: Статистика, 1979. - 247 с.

2. Броштейн И.И. Выбор стратегий обмена информацией между двумя уровнями памяти. - В кн.:Методы реализации алгоритмов контроля и управления АСУ. М.: ИПУ АН СССР, 1974, вып.3, с.34.

3. Липаев В.В., Филиппович В.В. Принципы и правила модульного построения сложных комплексов программ АСУ. - Управляющие системы и машины. 1975, № 1, с.15-22.

4. Колганова Т.В., Орлов С.Г., Филиппович В.В. Исследование характеристик иерархической структуры программ АСУ. - Управляющие системы и машины, 1976, № 5, с.42-52.

5. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. - М.: Мир, 1979. - 535 с.

6. Левнер Е.В., Генс Г.В. Дискретные оптимизационные задачи и эффективные приближенные алгоритмы: Препринт. М.: ЦЭМИ АН СССР. 1978. - 55 с.

КОМПЛЕКС АЛГОРИТМОВ УПРАВЛЕНИЯ ИСПЫТАНИЯМИ КОМПЛЕКТУЮЩИХ ИЗДЕЛИЙ РЭА И ЕГО ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

В.Н.Фролов, В.Г.Юрасов

Технологический процесс входного контроля надежности ЭРЭ играет важную роль при обеспечении заданных требований к РЭА. Снижение себестоимости входного контроля, уменьшение временных затрат, повышение надежности отбраковки элементов на современном этапе достигается за счет его автоматизации. Проблема автоматизации рассматриваемой технологической операции состоит из отдельных взаимосвязанных подсистем:

а) автоматизация управления режимами граничных испытаний и выполнение измерений;

б) автоматизированный анализ результатов тестирования и принятия окончательных решений по отбраковке комплектующих РЭА.

Решение этих задач связано с применением вычислительной техники, что выдвигает проблему организации машинного информационно-измерительного комплекса.