# Высокоуровневое распараллеливание задач выбора взаимосвязанных вариантов

М.П.Ревотюк, Н.В.Хаджинова

*Учреждение образования "Белорусский государственный университет информатики и радиоэлектроники", ул. П.Бровки, 6, Минск, Беларусь*

Задачи выбора на конечных множествах вариантов можно выделить практически во всех случаях дискретной оптимизации независимо от применяемой схемы поискового алгоритма (простого перебора, метода динамического программирования, ветвей и границ и др.). Конкретную задачу $Z$ можно характеризовать приемлемой для объектного представления тройкой

$$Z = \langle V, P, S \rangle, \quad (1)$$

где $V$ – множество вариантов, подлежащих оценке по заданному критерию; $P$ – процедура получения оценки качества для отдельного варианта из множества $V$; $S$ – процедура реализации вычислительной схемы решения $Z$, определяющая порядок применения $P$ к элементам множества $V$.

Цель исследования – анализ потенциальной эффективности распараллеливания процесса решения задач вида $Z$ на сети ЭВМ или многопроцессорной ЭВМ в условиях дефицита времени.

Пусть решается задача $Z$ минимизации целевой функции $F$, а используемая для оценки отдельного варианта процедура $P$ обладает возможностью получения в процессе работы нижних оценок значения $F$.

Процедура $S$ в (1), по существу, определяет порядок перечисления вариантов в последовательности $V = \{v_i\}, i = \overline{1,n}$. Если время анализа каждого отдельного варианта $v_i$ процедурой $P$ будет $w_i$, то время $W$ выбора рационального варианта при условии их независимого последовательного анализа на одиночной ЭВМ составит

$$W = \sum_{i=1}^{n} w_i. \quad (2)$$

Предположения о свойствах задачи $Z$:

- процедура $S$ допускает разделение вариантов $V$ на подмножества;
- время работы процедуры $P$ характеризуется аддитивной зависимостью от количества итераций оценки отдельного варианта;
- значения оценок целевой функции $F$ связаны на множестве номеров итераций зависимостями

$$\langle F_{i,j} \le F_{i,j+k} \rangle \vee \langle L_{i,j} \ge L_{i,j+k} \rangle, k > 0, \; i = \overline{1,n}. \quad (3)$$

Здесь $F_{i,j}$ - текущая оценка целевой функции $F$, $L_{i,j}$ – нижняя граница оценки такой функции для варианта $v_i$ на итерации $j$, $j = \overline{1,k(i)}$; $k(i)$ – количество итераций исчерпывающего анализа варианта $v$, когда выполняется условие $F_{i,k(i)} = L_{i,k(i)} = F(i), \; i = \overline{1,n}$ (рис. 1).

Характерный для многих задач минимизации вид изменения оценок варианта $v_i$ позволяет организовать рациональную схему вычислений, используя значение рекордной оценки $R$ лучшего из ранее просмотренных вариантов,

$$R = \min_j \{F_j, j = \overline{1, i-1}\}. \quad (4)$$

Итерации анализа варианта $v_i$ можно прервать на шаге $m(i)$, если установлена истинность условия $L_{i,m(i)} < R$. Так как $m(i) \le k(i), i = \overline{1,n}$ (рис.1), то полагая $w_i \approx k(i)$, можно заметить, что время последовательного анализа всех вариантов с учетом ранее полученных рекордных оценок составит

$$W^* = \sum_{i=1}^{n} m(i) = \sum_{i=1}^{n} \min_j \{k(j), j \le i\}. \qquad (5)$$

Сравнивая (2) и (5), легко обнаружить, что $W^* \le W$, причем эффект сокращения времени решения в конечном счёте зависит от порядка рассмотрения вариантов, т.е. от того, насколько рано выбираются хорошие варианты.

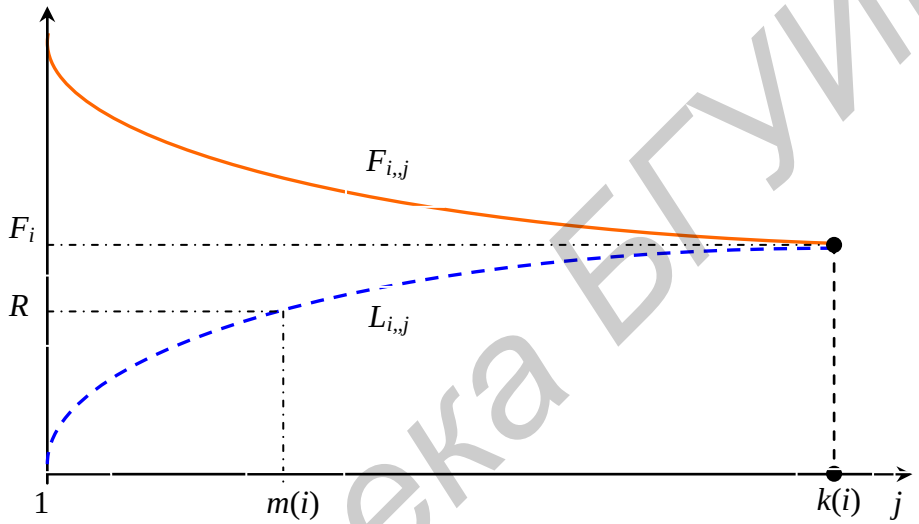


**Рис. 1.** Изменение оценки целевой функции при анализе отдельного варианта

Можно выделить следующие случаи законов упорядочения вариантов с однозначно прогнозируемой характеристикой качества:

- наилучший случай – $k(i) < k(i+1),\ \overline{i = 1, n-1}$;
- наихудший случай – $k(i) > k(i+1),\ \overline{i = 1, n-1}$;
- случай неприменимости накопления опыта – $k(i) = k(i+1),\ \overline{i = 1, n-1}$.

Закон упорядочения вариантов априорно неизвестен, в противном случае можно было бы просто исключить заведомо неперспективные из рассмотрения. Очевидно, что для характеристики задачи интерес представляет интервал $[k_{\min}, k_{\max}]$, где

$$k_{\min} = \min_i \{k(i),\ i = \overline{1,n}\},\ k_{\max} = \max_i \{k(i),\ i = \overline{1,n}\}.$$

Практически значимым для характеристики приемов улучшения метода организации решения задачи $Z$ является ширина интервала. Время анализа отдельного варианта зависит от текущего значения рекордной оценки, а ее значение снижается с увеличением количества просмотренных вариантов. Однако выражение (5) не предписывает обязательность последовательного просмотра вариантов.

Если процедура $S$, определяющая порядок перечисления вариантов в последовательности $V$, допускает образование $m$ подмножеств вариантов, то ускорение моментов установления бесперспективности может быть достигнуто

путем распараллеливания процесса решения между $m$ отдельными ЭВМ при незначительной модификации процедуры $P$.

Модификации процедуры $P$ состоит в том, что в процессе анализа вариантов каждая ЭВМ имеет доступ к значению оценки $F$ наилучшего варианта из числа просмотренных на всех ЭВМ. Если какая-то из ЭВМ улучшает значение $F$, то она немедленно передаёт сообщение об этом остальным ЭВМ. Таким образом, итерации анализа вариантов выполняются при известном текущем значении рекордной оценки $R$ согласно (4), которое может неоднократно улучшено в любой момент времени.

Пусть каждая ЭВМ получает для анализа некоторое подмножество вариантов $V_i$, причём

$$\bigcap_{i=1}^{m} V_i = \varnothing, \qquad \bigcup_{i=1}^{m} V_i = V. \tag{6}$$

Для оценки вычислительных затрат на анализ всех вариантов рассмотрим наихудший случай их перечисления, когда $k(i) > k(i+1),\ \overline{i=1,n-1}$. Если $k(1) = k_{\max}$, а $k(n) = k_{\min} = 0$, то (5) в этом случае можно представить в виде

$$W^* = \frac{k_{\max} \cdot n}{2}. \tag{7}$$

Пусть подмножества вариантов образованы так, что $|V_i| = n/m,\ i = \overline{1,m}$. Тогда при параллельной работе $m$ ЭВМ и синхронным рассмотрением отдельных вариантов время решения задачи $Z$ определяется продолжительностью рассмотрения последнего подмножества. Его оценка получается масштабированием (7) вида $k_{\max} \leftarrow k_{\max}/m,\ n \leftarrow n/m$. Отсюда следует, что время решения задачи на $m$ ЭВМ сокращается в $m^2$ раз.

Вследствие отсутствия информации о виде закона распределения оценок поступающих на анализ вариантов, предположения, лежащие в основе полученных выше результатов являются достаточно грубыми. Здесь предполагается однородность вычислительной среды на всём интервале решения задачи $Z$ и не учитывается возможность образования очередей сообщений.

Исследования эффективности кооперативных схем на основе более точных моделей вычислительных процессов и различных способов кооперирования вычислительных ресурсов [1,3] представляют скорее академический интерес. Важным является то, что пересечение интервалов работы ЭВМ во времени снижает зависимость среднего времени решения задачи от порядка просмотра вариантов. Параллельная работа ЭВМ исключает потребность тщательного конструирования процедуры $S$.

Имея в наличии методы или программные средства решения задачи $Z$, определяемой (1), часто представляет интерес сравнение альтернатив использования мощной ЭВМ или кооперации маломощных ЭВМ.

Пусть множество вариантов $V$ отображается на отрезок $x \in [0, x_{\max}]$, а значения вычислительной трудоемкости их анализа – на отрезок $y \in [0, y_{\max}]$.

Накопление опыта и время решения зависит от исходного упорядочения последовательности анализируемых вариантов. Наилучший случай, как отмечалось ранее, соответствует некоторой неубывающей, а наихудший – убывающей функции $y(x)$. Реальный вид такой функции априорно неизвестен. Однако, предполагая ее линейный вид, возможные случаи упорядочения вариантов можно рассматривать

относительно значения $dy/dx$ для прямой, проходящей через точку $(x_{\max}/2, y_{\max}/2)$. Фиксация точки отражает намерение нормировки времени решения.

Общее время анализа вариантов с учетом накопления опыта в системе из $m$ ЭВМ при рассматриваемых предположениях характеризуется зависимостью [2]:

$$t(m) = \begin{cases} xy_{\max}/m, & 0 < x \le x_{\max}/2; \\ (x_{\max} - x + (x - x_{\max})/2m)\, y_{\max}/m, & x_{\max}/2 < x \le x_{\max}. \end{cases}$$

Легко заметить, что $t(m) \in [\,\cdot\,, x_{\max} y_{\max}/2m]$. После усреднения по всем значениям $x$ для случая его равномерного распределения (для простоты формулировки выводов) на интервале $[0, x_{\max}]$ получим среднее время решения задачи $Z$ на $m$ ЭВМ:

$$\overline{t(m)} = x_{\max} y_{\max} (1 + 1/m)/8m.$$

Среднее время решения задачи $Z$ при независимом рассмотрении вариантов

$$\overline{t^*(m)} = x_{\max} y_{\max}/2m.$$

Из отношения $\overline{t^*(1)}/\overline{t(1)}$ следует, что накопление опыта на одиночной ЭВМ сокращает время решения в среднем в 4/3 раза.

Относительный эффект распределения процесса между $m$ агентами с обменом рекордными оценками

$$\frac{\overline{t(1)}}{\overline{t(m)}} = \frac{3m^2}{2m+1}. \qquad (8)$$

Очевидно, что при $m > 1$ время решения сокращается от 2.4 до ~1.5$m$ раз. Полученные оценки характеризуют полезность распараллеливания и могут быть использованы для обоснования вычислительной схемы решения задачи.

Реализация кооперативной схемы возможна в рамках системы, где основным активным элементом выступает агент, выявляющий доступность и готовность к решению задачи узла вычислительной сети. Известно, что "жадный" алгоритм распределения загрузки узлов сети является оптимальным по критерию минимума времени решения задачи. В настоящее время существует ряд технологических возможностей реализации распределенных вычислений (RPC, CORBA, DCOM). Объектно-ориентированное оформление агента в виде объекта полиморфного класса на языках C++ или JAVA, обеспечивает абстрагирование от уровня коммуникации между узлами сети [2], а определение задачи необходимо задавать в виде (1) .

В качестве примера рассмотрим решение классической задачи коммивояжера:

задана матрица расстояний (стоимости переезда) $C = \|c_{ij}, i, j = \overline{1,n}\|$ между любым из $n$ городов, необходимо найти цикл минимальной длины однократного посещения каждого города.

Формальная модель задачи имеет вид [4]:

$$\begin{cases} Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot x_{ij} \Rightarrow \min; \\ \sum_{i=1}^{n} x_{ij} = 1; \quad \sum_{j=1}^{n} x_{ij} = 1; \\ x_{ij} \ge 0, i, j = \overline{1,n}; \\ u_i - u_j + n x_{ij} \le n - 1, \ i = \overline{2,n}, \ j = \overline{2,n}, \ i \ne j. \end{cases} \qquad (9)$$

Модель (9) не является конструктивной для построения эффективной вычислительной схемы решения задачи коммивояжера. Среди точных методов решения ее наиболее эффективным считается метод ветвей и границ. Экспериментально установленная оценка вычислительной сложности метода ветвей и границ для случайной матрицы расстояний размером $n \times n$ – $O(1.26^n)$ [4].

Алгоритм метода ветвей и границ допускает естественное распараллеливание. Процесс анализа совокупности вариантов может быть реализован разными способами, различающимися правилами порождения ветвей дерева. Среди наиболее успешных известен подход, базирующийся на решении задач о назначении, анализе получающихся замкнутых циклов и, если таких циклов более одного, последующем переборе вариантов разрыва циклов. Рекурсия обхода дерева строится на матрице расстояний, где разрывы циклов задаются назначением бесконечных значений длин запрещаемых дуг. Установка бесконечного значения – естественный прием запрета нежелательных для использования дуг, например, $c_{ii} = \infty,\ i = \overline{1,n}.$

Организация кооперативной схемы требуют решения двух задач:

- определение способа спецификации отдельного варианта;
- включение в итерационный процесс механизма установления бесперспективности анализируемого варианта.

Узел дерева вариантов при решении задачи коммивояжера соответствует некоторой локальной задаче о назначениях

$$\begin{cases} Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot x_{ij} \Rightarrow \min; \\ \sum_{i=1}^{n} x_{ij} = 1; \quad \sum_{j=1}^{n} x_{ij} = 1; \\ x_{ij} \geq 0, i, j = \overline{1, n}. \end{cases} \qquad (10)$$

Задача о назначениях (10) полностью определена значением матрицы *C*, а ее решение содержит точно *n* ненулевых элементов в матрице *X*. Легко заметить, что каждое решение целесообразно представить не матрицей *X*, а вектором

$$Y = \{y_1 = j \,|\, x_{ij} = 1, i = \overline{1,n}; y_k = j \,|\, x_{ij} = 1, i = y_{k-1}, k = \overline{2,n}\}, \qquad (11)$$

перечисляющего вершины цикла оптимального решения. Таким образом, размер сообщения линейно зависит от размерности задачи.

Установление бесперспективности варианта для задачи о назначениях удобно проводить, воспользовавшись известным для транспортных задач приемом оценки нижней границы целевой функции в процессе их решения [5]. На основании теории двойственности, нижняя оценка целевой функции $\tilde{Z}^q$ на итерации $q$ решения транспортной задачи венгерским методом определяется следующим образом

$$\tilde{Z}^q = \sum_{i=1}^{n} u_i^q \cdot a_i + \sum_{j=1}^{n} v_j \cdot b_j, \qquad (12)$$

где

$$u_k^q = 0, k \in [1, n];$$

$$u_i^q = c_{i1} - c_{i1}^q - v_1^q, i \neq k;$$

$$v_j = c_{kj} - c_{kj}^q, j = \overline{1,n}.$$

Здесь $c_{..}^{q}$ - элемент вспомогательной матрицы оценок, построенной на итерации $q$ алгоритма венгерского метода [5]. Значения коэффициентов $a_{.}$ и $b_{.}$ в транспортной задаче представляют объемы поставки и потребления, но применительно к задаче о назначениях их следует полагать единичными:

$$a_i = 1,\ i = \overline{1,n};\ b_j = 1,\ j = \overline{1,n}.$$

Очевидно, что получение значений $\widetilde{Z}^{q}$ линейно зависит от размерности задачи, но требует сохранения исходной матрицы.

Таким образом, схема алгоритма работы агента имеет вид:

получив описание локальной задачи (10), решить задачу о назначениях, проверяя на каждой итерации перспективность варианта относительно глобального рекорда;

если получено законченное решение, то, во-первых, провести ветвление, помещая в стек локальных задач описание листьев текущего дерева и, во-вторых, скорректировать известное всем значение рекордной оценки;

если стек локальных задач не пуст, выбрать новый вариант и перейти к его анализу, в противном случае ожидать сообщения о новой задаче.

Представление исходной задачи (9) должно быть известно каждому агенту, но отдельные задачи в стеке можно задать в разностной форме (11).

Экспериментальная реализация рассмотренных агентов показала, что на системе из $m$ кооперируемых на локальной сети ЭВМ, $m$<12, сокращение времени решения задачи при $n < 1000$ достигает (0.97…1.74)·$m$ раз по сравнению с использованием одиночной ЭВМ. При этом существенным оказывается распределение значений матрицы стоимостей.

Представление задач выбора (1) иерархией полиморфных классов [2] в рамках объектно-ориентированных технологий не порождает специальных требований к вычислительной среде. Для обеспечения обмена информацией о рекордных оценках и анализируемых вариантов достаточным оказывается наличия механизма обмена сообщениями.

Высокоуровневое распараллеливание задач, реализованное на системе агентов, ожидающих появления описания задач в форме (1), характеризуется технологичностью реализации, надежностью вычислительного процесса и возможностью утилизации доступных ресурсов вычислительной сети.

**Литература**

1. Тихомирова Е.В. Характеристики потоков анализируемых вариантов в кооперативных схемах//Моделирование и информационные технологии проектирования: Сб. научн. тр., вып.4 - Мн.: Объединенный институт проблем информатики НАН Беларуси, 2002. - С.110-116.

2. Ревотюк М.П., Кузнецова Н.В. Агентная система кооперации ресурсов вычислительной среды для решения задач выбора//Известия Белорусской инженерной академии, № 1(15)/1, 2003. – С. 265-268.

3. Ревотюк М.П., Кузнецова Н.В. Оценка эффекта распараллеливания жадных алгоритмов//Известия Белорусской инженерной академии, № 1(17)/3, 2004. – С. 206-209

4. Рейнгольд Э., Нивергельд Ю., Део Н. Комбинаторные алгоритмы. Теория и практика/Пер. с англ. – М.: Мир. 1980. – 476 с.

5. Юдин Д.Б., Гольштейн Е.Н. Задачи и методы линейного программирования. – М.: Советское радио, 1964. – 736 с.