УДК 519.85(075.8)

ОТКРЫТАЯ АГЕНТНАЯ СИСТЕМА РЕШЕНИЯ ЗАДАЧ ВЫБОРА НА ВЫЧИСЛИТЕЛЬНЫХ СЕТЯХ

М.П. Ревотюк
Белорусский государственный университет информатики и радиоэлектроники
Республика Беларусь, 220013, Минск, ул. П.Бровки, 6
E-mail: rmp@bsuir.unibel.by

Н.В. Кузнецова
Белорусский государственный университет информатики и радиоэлектроники
Республика Беларусь, 220013, Минск, ул. П.Бровки, 6
E-mail: kafitas@bsuir.unibel.by

Ключевые слова: задачи поиска решений, вычислительные сети, агентные системы, кооперативные схемы
Key words: search optimal variant problems, computer networks, agents systems, cooperative schemata

Объект рассмотрения – технология решения разовых или эпизодически возникающих задач повышенной вычислительной сложности по кооперативной схеме использования ресурсов вычислительной сети. Кооперация доступных ресурсов сети часто позволяет достичь нелинейного сокращения времени решения задачи даже без конструирования специальных вычислительных схем. Цель исследования – разработка шаблона платформенно-независимой открытой агентной системы решения задач выбора на конечных множествах вариантов и его реализация в рамках объектно-ориентированных технологий.
OPENED AGENT'S SYSTEM FOR CONCURRENT VARIANT SELECTION ON COMPUTER NETWORKS / M.P. Revotyuk (Belarusian State University of Informatiks and Radioelectronics, 6 Brovki, Minsk 220013, Republic of Belarus, E-mail: rmp@bsuir.unibel.by), N.V. Kuznetsova (Belarusian State University of Informatics and Radioelectronics, 6 Brovki, Minsk 220013, Republic of Belarus, E-mail: kafitas@bsuir.unibel.by). Technology of programming of real time slow flow intensity erased problems with high computation complexity, based on cooperative schemata using of computer networks resources, is presented. Cooperation of available networks resources often give effect of nonlinear decreasing of computation time without constructing of special algorithms. Goal of investigation – creating independent of platform opened agent's system template for select optimal variant from finite set one and it presentation in object based style.

1. Введение

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 решаемую задачу Z можно характеризовать тройкой

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

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*.

Построение в приемлемые сроки строгих и, вместе с тем, изящных в математическом отношении схем решения практических задач дискретной оптимизации в ряде случаев не представляется возможным.

Типичной является следующая ситуация:

- имеется алгоритм и программа *P* оценки отдельного варианта из *V*;
- существует некоторый вариант схемы *S*, гарантирующий решение задачи *Z*;
- реализация схемы *S* возможна на параллельно функционирующих ЭВМ вычислительной сети путём тривиальной декомпозиции разбиением множества *V* на подмножества.

Попытки разработки специальной схемы *S*, детально учитывающей особенности задачи *Z*, в ряде случаев не предпринимаются по формальным либо технологическим соображениям. Принцип приоритета действия перед планом привлекателен как для потребителя результатов, особенно при решении разовых задач, так и для разработчика, использующего объектно-ориентированные технологии для быстрого создания рабочей версии системы.

Очевидно, что прямолинейная реализация схемы *S* на *N* параллельно функционирующих ЭВМ приведёт к сокращению времени решения *Z* приблизительно в *N* раз. Однако реальные задачи часто позволяют использовать идею накопления опыта, возникающего при параллельном рассмотрении отдельных вариантов. Это приведёт к сокращению времени решения *Z* более, чем в *N* раз.

Предметом обсуждения является представление (1) в рамках объектно-ориентированных технологий на системе взаимодействующих агентов.

2. Оценка эффекта распределения задач выбора

2.1. Предпосылки сокращения времени решения

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

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

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

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

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_i , когда выполняется условие $F_{i,k(i)} = L_{i,k(i)} = F(i),\ i = \overline{1,n}$ (рис. 1).

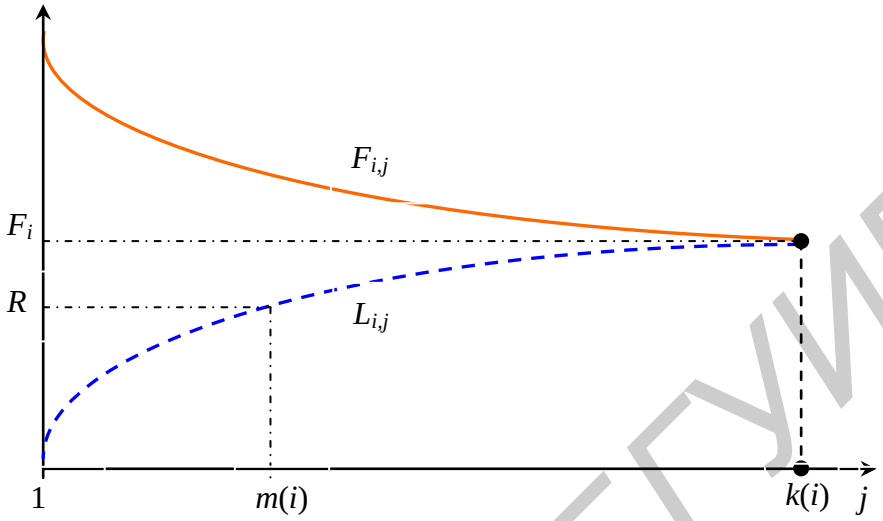

Рис. 1. Изменение оценки целевой функции при анализе отдельного варианта

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,

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

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

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

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.

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 – $k(i) = k(i+1),\ \overline{i = 1, n-1}$.

Однако упорядочение вариантов априорно неизвестно, иначе 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.

2.2. Параллельное накопления опыта

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

(6) $$\bigcap_{i=1}^{m} V_i = \varnothing, \quad \bigcup_{i=1}^{m} V_i = V .$$

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

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

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] 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.*

2.3. Потенциальные оценки кооперативных схем

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 $x \in (0, b]$, а значения вычислительной трудоемкости их анализа - на отрезок $y \in (0, a]$.

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 $(a/2,\ b/2)$. 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]:

$$t(m) = \begin{cases} bx/m, & 0 < x \le a/2; \\ (a - x + (x - a)/2m)\,b/m, & a/2 < x \le a. \end{cases}$$

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

$$\overline{t(m)} = ab(1 + 1/m)/8m.$$

Среднее время решения задачи *Z* при независимом рассмотрении вариантов

$$\overline{t^*(m)} = ab/2m.$$

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.

3. Схема агента анализа варианта

Анализ структуры задач выбора показывает, что 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 *Z*. В настоящее время существует ряд технологических возможностей реализации распределенных вычислений (RPC, CORBA, DCOM), однако вопрос построения алгоритма решения задачи и его реализации остается в общем случае открытым.

Технология построения открытых систем взаимодействующих агентов в современных средах, включая Internet, базирующаяся на оформлении агента в виде объекта класса, может быть основой решения задачи обеспечения коммуникации между узлами сети. Реализацию процедуры анализа варианта удобно представить классом на языках C++ или JAVA, где в случае необходимости может быть предусмотрен интерфейс вызова "родных" функций вычислительной среды исполнения агента. Такие функции должны представлять, в частности, процедуру *P*. Остальные функции не являются проблемно-зависимыми и могут с точностью до протокола обмена соответствовать виртуальным операторам присваивания.

Идея объектно-ориентированного представления функциональной части агента (для краткости представлена шаблоном класса на языке C++):

```
template <class Criterial, class Variant> class Agent {
 Criterial F;
 static volatile
  Criterial R; // Глобальный объект рекордной оценки
protected:
 virtual Criterial FirstStep();
 virtual bool NextStep(Criterial &);
public:
 virtual void P(Variant V) {
  F = V.FirstStep(); // Предварительная оценка  варианта
  if (R>F) { // Организация анализа варианта
   while (V.NextStep(F)) if (R<F) return;
   if (R>F) R = F; // Установление нового рекорда
  }
 }
};
```

Здесь предполагается решение задачи минимизации целевой функции *F*. Кроме того, используемая для оценки отдельного варианта процедура *P* обладает возможностью получения в процессе работы нижних оценок значения функции *F*, а *R* – текущее значение рекордной оценки, является общедоступным для всех агентов.

Полиморфизм функций сравнения позволяет считать класс Agent проблемно независимым. Система взаимодействующих агентов обеспечивает объединение ресурсов вычислительной сети с целью гарантированного решения исходной задачи Z. Ее поведение предсказуемо во времени, что обусловлено возможностью адаптивной идентификации наблюдаемых параметров модели вычислительных процессов. Агенты должны быть размещены во всех потенциально доступных узлах вычислительной сети.

Предположим, что доступные для кооперации ЭВМ оснащены агентами, ожидающими получения описания локальной задачи [3]. Получив задание, агент активизирует процесс ее решения, продолжая следить за общей обстановкой на сети. Слежение заключается в проверке условия бесперспективности анализируемого варианта посредством сравнения оценки его целевой функции с рекордным значением.

Агент решения задачи должен выполнять посылку широковещательного сообщения группе кооперируемых ЭВМ сети. Такая посылка легко реализуема посредством применения протокола UDP и группового обмена на уровне гнездовых соединений [4]. Групповой обмен на локальных вычислительных сетях исключает образование очередей сообщений.

4. Примеры использования кооперативных схем

4.1. Организация статистических экспериментов

Имитационное моделирование систем, характеризуясь универсальностью применения, базируется на экспериментальной оценке разнообразных сложных систем методом статистических испытаний [5].

Характерные черты метода статистических испытаний, существенные для организации корпоративной схемы вычислительного процесса:

- последовательная схема анализа вариантов системы, допускающая произвольный порядок их перечислений;
- итерационная схема анализа отдельного варианта, обусловленная необходимостью накопления статистики для достижения требуемой точности и достоверности результатов моделирования [5].

В общем случае отказ от анализа варианта из-за установления его бесперспективности может потребовать нетривиальной реализации оператора сравнения R и F в представленном выше классе Agent. Например, пусть методом статистических испытаний решается задача выбора оптимального по критерию F варианта системы из конечного множества конкурирующих вариантов.

Практически всегда можно считать, что значение F для любого варианта системы распределено по нормальному закону с неизвестной дисперсией. Правомочность такого предположения следует из закона больших чисел и предельных теорем теории вероятности, а также из-за отсутствия обоснованных гипотез о значении дисперсии оценки F.

Доверительный интервал $\varepsilon(n)$ оценки F после n итераций статистических испытаний определяется при указанных предположениях выражением:

$$\varepsilon(n) = t_{\alpha/2,n-1}\sqrt{d/n}\,,$$

где $t_{\alpha/2,n-1}$ – квантиль распределения Стьюдента с n-1 степенью свободы, соответствующий доверительной вероятности 1-α ; d – оценка дисперсии величины F по результатам n испытаний.

Истинное значение F с вероятностью 1-α находится в интервале $[\widehat{F}-\varepsilon(n),\widehat{F}+\varepsilon(n)]$, где $\widehat{F}$ – оценка математического ожидания величины F по результатам n испытаний.

Пусть R – текущее значение оценки лучшего из проанализированных вариантов. Очевидно, что при решении задачи минимизации F условием прекращения анализа текущего варианта из-за его бесперспективности является

$$\widehat{F}-\varepsilon(n) > R\,.$$

При выявлении ситуации $\widehat{F} > R$ при некотором значении n можно определить требуемое число испытаний из уравнения

$$\widehat{F}-\varepsilon(m) = R\,,$$

где m – прогнозируемый номер испытания для вынесения, возможно, окончательного суждения об анализируемом варианте.

Можно заметить, что значение

$$m = \frac{\widehat{F}-R}{t_{\alpha/2,n-1}\cdot d}\,.$$

пригодно и для случая решения задачи максимизации значения F.

4.2. Решение задачи коммивояжера методом ветвей и границ

Задача коммивояжера в классической постановке формулируется следующим образом: задана матрица расстояний (стоимости переезда) $C = \left\| c_{ij}, i, j = \overline{1,n} \right\|$ между любым из n городов, необходимо найти цикл минимальной длины однократного посещения каждого города. Если представить решение задачи коммивояжера матрицей булевых переменных $X = \left\| x_{ij}, i, j = \overline{1,n} \right\|$, где единица означает включение в оптимальный цикл дуги $i \to j$, то формальная модель оптимизации имеет вид:

$$
\begin{aligned}
& Z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} \cdot x_{ij} \Rightarrow \min; \\
& \sum_{i=1}^{n} x_{ij} = 1; \\
& \sum_{j=1}^{n} x_{ij} = 1; \\
& x_{ij} \geq 0, i, j = \overline{1,n}; \\
& u_i - u_j + n x_{ij} \leq n - 1, \ i = \overline{2,n}, \ j = \overline{2,n}, \ i \neq j.
\end{aligned}
\qquad (9)
$$

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 $n \times n$ – $O(1.26^n)$ [6]. Экспоненциальный вид закона объясняется древовидной структурой процесса поиска оптимального цикла.

Алгоритм метода ветвей и границ допускает естественное распараллеливание процесса анализа совокупности вариантов. Древовидный характер порождения вариантов позволяет разделить процесс генерации и обработки любой вершины дерева между параллельно функционирующими ЭВМ на сети.

Схема алгоритма метода ветвей и границ может быть реализована разными способами, различающимися правилами порождения ветвей дерева. Наиболее успешным считается 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, как например, $c_{ii} = \infty, \ i = \overline{1,n}.$

Организация кооперативной схемы требуют решения двух задач:

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

Рассмотрим содержание пересылаемых сообщений. Узел дерева вариантов при решении задачи коммивояжера соответствует некоторой локальной задаче о назначениях

$$Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} \cdot x_{ij} \Rightarrow \min;$$

$$\sum_{i=1}^{n} x_{ij} = 1; \tag{10}$$

$$\sum_{j=1}^{n} x_{ij} = 1;$$

$$x_{ij} \geq 0, i, j = \overline{1, n}.$$

Задача о назначениях 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

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

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 [7]. На основании теории двойственности, нижняя оценка целевой функции $\tilde{Z}^q$ на итерации q решения транспортной задачи венгерским методом определяется следующим образом

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

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_{ij}^q - элемент вспомогательной матрицы оценок, построенной на итерации q алгоритма венгерского метода [7]. Значения коэффициентов a_i и b_j 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 $\tilde{Z}^q$ 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 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.

4.3. Решение задачи размещения методом перебора

Рассмотрим одну из разновидностей задачи размещения производственных объектов, относящихся к классу транспортных.

Пусть имеется n пунктов потребления некоторой однородной продукции, характеризуемых вектором потребления $B = b_j, j = \overline{1,n}$. Для удовлетворения потребностей продукции требуется разместить m однородных объектов с объёмом производства $A = a_i, \ i = \overline{1,m}$.

Предполагается выполнение условия баланса:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j .$$

Предположим, что производственные объекты могут быть размещены в любом из пунктов, пронумерованных от 1 до k, а затраты на транспортировку продукции характеризуются матрицей $C = \left\| c_{ij}, i = \overline{1,m}, \ j = \overline{1,n} \right\|$.

Обозначим R – множество мест размещения объектов производства , $|R|$=k, а r - подмножество, представляющее конкретный вариант, $|r|=m$.

При фиксации варианта $r \subset R$, величина транспортных расходов на перевозку продукции составит

$$C(r) = \sum_{i \in r} \sum_{j=1}^{n} x_{ij} \cdot c_{ij},$$

где x_{ij} - объём продукции, поставляемой из пункта i в пункт j.

Полагается, что объемы поставки и потребления при этом сбалансированы:

$$\sum_{i \in r} x_{ij} = b_j, j = \overline{1,n};$$

$$\sum_{j=1}^{n} x_{ij} = a_i, \ i = \overline{1,m}.$$

Задача минимизации $C(r)$ при последних условиях – классическая транспортная задача. Цель решения обсуждаемой здесь задачи – найти

$$r^* = \arg\min_{r \subset R} C(r),$$

т.е. сочетания m номеров из k.

Возможен подход к решению такой задачи в рамках целочисленного линейного программирования. Постановка задачи существенно определяется схемой порождения подмножеств $r \subset R$. Если описание такой схемы связано с трудно формализуемыми правилами, то метод перебора остается вполне приемлемым.

Структуризация рассматриваемой задачи, как задачи выбора Z очевидна. Вариант v представлен классической транспортной задачей, а множество вариантов V определяется набором всех сочетаний из k элементов множества R по m элементов. Процедура P оценки отдельного варианта – одна из известных схем решения транспортной задачи. Процедура S порождения вариантов может представлять перебор всех допустимых комбинаций образования $r \subset R$.

Среди известных алгоритмов решения транспортных задач здесь удобно выбрать алгоритм венгерского метода. Являясь итерационным, он допускает получение на каждой итерации нижней оценки целевой функции (12). Получение значения такой оценки характеризуется вычислительной сложностью $O(m+n)$. Вычислительная сложность алгоритма венгерского метода $O(m \cdot n)$, поэтому увеличение трудоёмкости модифицированной схемы незначительно.

5. Заключение

Распараллеливание может быть привлекательным приемом утилизации доступных вычислительных ресурсов в сетях с неравномерной загрузкой. Например, в системах управления группами оборудования в роботизированных производствах задачи оперативного планирования возникают редко, но требуют единовременно больших вычислительных ресурсов.

Практическая реализации кооперативных схем возможна после оснащения потенциально доступных ЭВМ агентами, ожидающими получения описания подлежащей решению задачи. Предложенный объектный стиль построения агента и общего представления задач выбора из дискретных множеств вариантов допускает реализацию оптимального по критерию времени жадного алгоритма управления процессом решения задач. Очевидна реализуемость агентов отдельными потоками в мультипроцессорных системах.

Проведенные эксперименты решения ряда комбинаторных задач выбора подтверждают, что кооперация M ЭВМ время решения более, чем в M раз за счет повышения скорости обнаружения бесперспективных вариантов.

Список литературы

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. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно–ориентированного проектирования. Паттерны проектирования/Пер. с англ. – СПб.: Питер, 2001. – 386 с.
4. Родли Д. Создание JAVA – апплетов/Пер. с англ. – К.: НИПФ “ДиаСофт Лтд.”, 1996. – 386.
5. Шеннон Р. Имитационное моделирование систем – искусство и наука/Пер. с англ. – М.: Мир, 1978. – 418 с.
6. Рейнгольд Э., Нивергельд Ю., Део Н. Комбинаторные алгоритмы. Теория и практика/Пер. с англ. – М.: Мир. 1980. – 476 с.
7. Юдин Д.Б., Гольштейн Е.Н. Задачи и методы линейного программирования. – М.: Советское радио, 1964. – 736 с.