

УДК 519.854

Л.Ф. Гуляницкий, Ю.С. Гложик

Институт кибернетики им. В.М. Глушкова НАН Украины, г. Киев
Национальный университет им. Т.Г. Шевченко, г. Киев, Украина
lh_dar@hotmail.com, pgt@ukr.net

Параллельный метод деформируемых многогранников для решения задач комбинаторной оптимизации

Описывается алгоритм комбинаторной оптимизации (КО), основанный на идеях метода Нелдера – Мида, названный гибридным методом деформируемых многогранников (ГМДМ). Этот алгоритм обладает глобальным характером поиска в пространстве решений решаемой задачи КО, что позволяет находить варианты решений задач из разных классов с повышенной точностью. В работе предлагается параллельный алгоритм ГМДМ и обсуждаются вопросы его реализации на многопроцессорном суперкомпьютере СКИТ-1. Исследование эффективности предложенного алгоритма осуществлено на основе анализа результатов проведенного вычислительного эксперимента по решению известной оптимизационной модели – квадратичной задачи о назначении (КЗН).

Введение

Используемые во многих сферах применения математических методов и моделей задачи КО принадлежат к числу труднорешаемых задач. Этим обуславливается необходимость как разработки новых эффективных приближенных алгоритмов решения, так и использования многопроцессорных вычислительных компьютеров.

Поскольку теоретическое исследование сходимости и точности алгоритмов КО редко приводит к результатам, которые можно использовать в практическом применении, исследователи осуществляют анализ таких алгоритмов путем проведения вычислительного эксперимента. Широко распространенной практикой является решение «типовых» задач КО, в качестве которых в подавляющем большинстве выступают задачи коммивояжера или КЗН.

КЗН находит широкое применение в различных сферах – разработка радиоэлектронной аппаратуры, экономика, планирование и размещение производства и др. [1] В то же время она служит признанным полигоном для разработки и исследования методов дискретной оптимизации. Несмотря на то, что эта задача исследуется давно [2], [3], точное решение даже при использовании суперЭВМ вызывает затруднение в общем случае уже для $n > 15 - 20$ [4], причем поиск ε -приближенного решения является также *NP*-полной проблемой. Этим объясняется интерес к приближенным и параллельным алгоритмам решения.

Формализация задачи

Пусть задано множество из n объектов, между каждой парой i, j которых указана степень их взаимосвязи c_{ij} . Кроме того, известны m позиций (мест), в

которые могут помещаться объекты (в каждую позицию – не более одного), причем заданы расстояния d_{ij} , являющиеся мерой удаления мест друг от друга, $i, j = \overline{1, m}$. Таким образом, заданы две матрицы $C = (c_{ij})_{n \times n}$ и $D = (d_{ij})_{m \times m}$, как правило, симметричные. Требуется так назначить заданные объекты в выделенные позиции, чтобы величина суммарного расстояния с учетом степени взаимосвязей была бы минимальной. Обычно полагают $n = m$, поскольку в противном случае можно считать объекты $n + 1, \dots, m$ фиктивными, т.е. для них $c_{ij} = 0$.

Известны постановки КЗН в виде моделей комбинаторной оптимизации и целочисленного программирования, в частности с булевыми переменными [1].

Комбинаторная постановка задачи. Предполагается, что объекты и позиции пронумерованы числами от 1 до n , а произвольный вариант назначения объектов в позиции описывается перестановкой $x = (x_1, \dots, x_n)$ элементов множества $\{1, \dots, n\}$ по следующему правилу: объект с номером i помещается в позицию с номером x_i . Обозначим множество всех таких перестановок P_n , т.е. $P_n = \{x \neq (x_1, \dots, x_n)\}$. Тогда КЗН можно сформулировать так: найти перестановку $x^* \in P_n$, для которой достигается

$$\min \left[f(x) = \sum_{i=1}^n \sum_{j=1}^n c_{ij} d_{x_i x_j} \right], \quad (1)$$

где $x = (x_1, \dots, x_n) \in P_n$.

Одним из главных ключей к классификации КЗН являются условия, которым удовлетворяют матрицы C и D (симметричность, неотрицательность, планарность соответствующего графа и т.д.). В ряде случаев возникает необходимость учитывать некоторые ограничения на размещение объектов: запрещенные позиции, фиксированное назначения ряда элементов в заранее указанные позиции, взаимное положение некоторых объектов, их удаленность и др.

КЗН в терминах целочисленного программирования. Произвольную перестановку из P_n можно описать при помощи так называемой перестановочной матрицы $X = (x_{ij})_{n \times n}$ – булевой матрицы, удовлетворяющей условиям

$$\begin{aligned} \sum_{i=1}^n x_{ij} &= 1, & i &= \overline{1, n}; \\ \sum_{j=1}^n x_{ij} &= 1, & j &= \overline{1, n}; \\ x_{ij} &\in \{0, 1\}, & i, j &= \overline{1, n}. \end{aligned} \quad (2)$$

Поэтому (1) можно сформулировать так: ищется по всем перестановочным матрицам

$$\min \left[F(X) \equiv \sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^n \sum_{l=1}^n c_{ik} d_{jl} x_{ij} x_{kl} \right]. \quad (3)$$

Вид функции (3) объясняет смысл термина «квадратичная задача о назначениях», который был введен в [2], а постановка (1) или (3) называется формой Купмена – Бэкмана. Заменой $a_{ijkl} = c_{ik} d_{jl}$ можно из (3) получить форму Лоулера:

$$F(X) \equiv \sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^n \sum_{l=1}^n a_{ijkl} x_{ij} x_{kl}.$$

Дальнейшая вариация постановок КЗН связана с ее линеаризацией, хотя этот путь приводит к существенному росту размерности возникающих задач.

Методы локального поиска

В алгоритмах этого типа вместо полного перебора применяется направленный локальный перебор в подмножествах вариантов, которые называются окрестностями – этим объясняется их название: *локальный поиск* (ЛП). Общая схема локального поиска в задачах минимизации такая: начиная с некоторого допустимого решения задачи, новое решение с лучшим значением целевой функции ищется в его окрестности. Если такое решение найдено, оно принимается и поиск улучшения решения дальше осуществляется уже в его окрестности, и т.д. Алгоритм заканчивается, когда достигнут локальный оптимум, то есть когда в окрестности текущего решения нет никакого другого варианта с меньшим значением целевой функции.

Преимущество алгоритмов ЛП заключается в том, что в большинстве случаев пространство вариантов многих задач может быть исследовано очень эффективно: вместо того чтобы вычислять значение целевой функции для $y \in N(x)$, достаточно вычислить только разницу $\Delta = f(x) - f(y)$ – это свойство используется в рассматриваемых алгоритмах.

Недостатком ЛП является то, что полученные решения являются только локальными оптимумами. Поэтому, чтобы позволить отыскивать более точные решения, на основе этой схемы были разработаны алгоритмы имитационного отжига (АИО), вероятностного моделирования и другие алгоритмы, основанные на поиске в окрестности [5].

Алгоритм имитационного отжига

В АИО поиск глобального решения имитируется процессом релаксации механической системы со многими степенями свободы, в качестве значения обобщенной энергии (гамильтониана) которой выступает целевая функция задачи. Из статистической механики известно, что такие системы при некоторой конечной температуре стремятся к состоянию равновесия. Флуктуации, возникающие при высоких температурах, имеют вероятностный характер и могут приводить к временному увеличению энергии системы, однако в общем при постепенном понижении температуры термодинамическая система приходит в состояние равновесия, соответствующее минимальному значению обобщенной энергии системы. Вероятность p перехода из одного состояния в некоторое «близкое» ему состояние (флуктуация) описывается из распределения Больцмана – Гиббса так:

$$p = e^{-\frac{\Delta E}{kT}}, \quad (4)$$

где ΔE – приращение энергии при таком переходе, T – температура, k – постоянная Больцмана [6]. Итак, предполагается, что каждый вариант решения задачи комбинаторной оптимизации $x = (x_1, \dots, x_n)$ задает некоторую конфигурацию атомов

стохастической системы, а значение целевой функции $f(x)$ определяет величину обобщенной энергии.

Первая вычислительная процедура, имитирующая поведение стохастической системы, была предложена Метрополисом. Дальнейшим обобщением этого метода стал алгоритм имитационного отжига, на основе которого в последнее время интенсивно развиваются алгоритмы решения различного рода задач дискретной оптимизации (рис. 1).

Анализ алгоритмов отжига позволяет сделать вывод, что при разработке конкретного алгоритма необходимо определить:

- 1) множество всех вариантов решения задачи;
- 2) понятие соседних к текущему варианту решений и способ их генерации;
- 3) вероятность перехода от текущего варианта к соседним;
- 4) понятие равновесия при данной температуре;
- 5) температурное расписание, т.е. правило изменения значений параметра T ;
- 6) правило останова.

```

begin
   $x :=$  некоторый начальный вариант решения;
  { COMMENT:  $x$  - текущий вариант; }
   $T :=$  начальная температура;
  while (критерий останова не выполнен) do
    begin
      while ( не достигнуто равновесие ) do
        begin
           $y :=$  некоторый соседний с  $x$  вариант;
           $D := f(y) - f(x)$  ;
           $p := \min \left\{ 1, e^{-\frac{\Delta}{kT}} \right\}$ ;
           $o := \text{random} [0,1]$  ;
          if ( $o < p$ ) then  $x := y$ ;
        end;
        изменение  $T$  ;
      end;
    end;
  вывод рекорда;
end;

```

Рисунок 1 – Метод имитационного отжига

Первые два пункта традиционны для методов локального поиска, где, как правило, понятие соседства коррелирует с той или иной метрикой на множестве решений задачи. Порождение соседних решений может осуществляться как на основе их случайного выбора, так и детерминированно. Вероятность перехода в большинстве случаев определяется, как и в алгоритме Метрополиса. Поиск условий равновесия может основываться на физических аналогах с использованием энтропии, что позволяет надеяться на оценку среднего значения энергии при данной температуре [6].

Равновесие может определяться и так: задаются некоторый (натуральный) параметр ν и вещественное число $E > 0$, а выполнение ν переходов называют прогоном [7]. Если при данной температуре выполнено k прогонов и получены значения f_1, \dots, f_k , то считают, что достигнуто равновесие, если в $(k+1)$ -м прогоне

$$\frac{|f_{k+1} - f_1|}{f_{k+1}} < E \quad (5)$$

для некоторого $i \in \{1, \dots, k\}$. Величина f_i для прогона может быть, например, средним значением целевой функции или наилучшим. В [6] считается достигнутым положение равновесия, если было осуществлено либо заданное априори число переходов, либо число попыток превысило также заданную величину.

Температурное расписание определяет значения, принимаемые параметром T , и может быть:

- равномерным [6];
- медленным в начале, быстрым в конце;
- быстрым везде.

В [6], например, предложено полагать $T_0 = 10$, а $T_h = (T_1 / T_0)^h T_0$, где $T_1 / T_0 = 0.9$.

Ускоренный алгоритм вероятностного моделирования

Поскольку многократное вычисление экспоненты требует довольно значительных затрат машинного времени, ниже описывается алгоритм, в котором вероятность перехода от текущего варианта решения к соседнему вычисляется по упрощенной формуле, а сам процесс вероятностного моделирования организован иначе, чем в методах отжига (рис. 2).

Пусть $G: [0, 1] \rightarrow [0, 1]$ – некоторая строго монотонная функция, зависящая от параметра $H > 0$, M – натуральное, а γ, H, ε – действительные числа. Все эти величины выступают в качестве параметров алгоритмов, причем различным выбором функции G можно породить целое семейство таких алгоритмов, которые назовем G -алгоритмами [7].

В качестве функции G обычно выбираются функции

$$G_k(x) = (x^{1/k} + H)^k, \text{ для } x > 0. \quad (6)$$

Метод деформаций в комбинаторной оптимизации

Простой метод деформируемых многогранников. При решении задач недифференцируемой нелинейной оптимизации успешно применяется поиск методом деформируемых многогранников Нелдера – Мида [8]. Используя основные идеи этого метода, в работе [9] был предложен алгоритм дискретной оптимизации, предназначенный для использования как на традиционных, так и на многопроцессорных ЭВМ, который также был назван *методом деформируемых многогранников* (МДМ).

```

begin
     $x :=$  некоторый начальный вариант решения;
     $m := 0$ ;  $l := 0$ ;  $h := 0$ ;
    while (окрестность не просмотрена полностью) do
        begin
            while ( $1 \neq M$ ) do
                begin
                     $y :=$  очередная точка окрестности  $x$ ;
                     $? := ?(i, j)$ ;
                     $p := \min \left\{ 1, 1 - \frac{\Delta 100}{\gamma f(x)} \right\}$ ;
                     $o := m_h + \text{random} [0, 1] (1 - m_h)$ ;
                    if ( $p > o$ ) then begin  $x := y$ ;  $l := l + 1$ ; end;
                end;
                 $m_{h+1} := G(m_h)$ ;
                 $h := h + 1$ ;
                 $l := 0$ ;
            end;
        end;
    вывод рекорда;
end;
    
```

Рисунок 2 – G-алгоритм

В алгоритмах МДМ существенную роль играет понятие луча, которое вводится на основе понятия *d-отрезка* [10]. Если в заданном комбинаторном метрическом пространстве X с метрикой $d(x, y)$ положить $h = \inf d(x, y)$ для $x \neq y$, то *d-отрезком* $\langle x, y \rangle$, что соединяет любые две точки $x, y \in X$, называется упорядоченная совокупность точек $x_i \in X$, $i = \overline{1, k}$, которые удовлетворяют условиям $d(x, x_i) + d(x_i, y) = d(x, y)$, $i = \overline{1, k}$, причем $x_1 = x$, $x_k = y$, а $d(x, x_i) < d(x, x_{i+1})$, $i = \overline{1, k-1}$, и не существует такой точки $z \in X$, что $d(x_i, z) + d(z, x_{i+1}) = d(x_i, x_{i+1})$, $z \neq x_i$, $z \neq x_{i+1}$, $i = \overline{1, k-1}$. При этом рассматриваются такие пространства, для которых, если $d(x, y) > h$, то интервал $\langle x, y \rangle \equiv \langle x, y \rangle \setminus \{x, y\} \neq \emptyset$. В дальнейшем будет использоваться понятие полуинтервала, соединяющего две точки $x, y \in X$, под которым понимается *d-отрезок* без одной крайней точки: $\langle x, y \rangle \equiv \langle x, y \rangle \setminus \{x\}$.

В приведенных обозначениях алгоритм простого МДМ представлен на рис. 3. Здесь Q – множество перспективных на каждом шаге точек. Вершина многогранника теряет статус перспективной, если в множестве S субоптимальных решений на полуинтервалах, проведенных из этой вершины, не найдено улучшения решения.

Гибридные алгоритмы метода деформаций. В ряде развитых алгоритмов комбинаторной оптимизации для повышения точности получаемых решений используются процедуры возмущения (повторяющийся ЛП и *G*-алгоритмы). Осуществляемое построение полуинтервалов дает возможность соединять поиск в окрестности с глобальным сканированием пространства X , причем процедура сканирования, в отличие от общих операторов возмущения в большинстве других методов, определена конкретно.

```

procedure MDM( $x$ );
begin
   $Q := \Pi$ 
  for  $i := 1$  to  $m$  do
     $x :=$  некоторый начальный вариант решения;
     $P := P \cup x$ ;
     $Q := Q \cup x$ ;
  endfor; {сформирован начальный многогранник, все точки перспективны}
repeat
  выбор  $x \in P$ ;
  if  $x \in Q$  then
    begin
       $S := \Pi$ 
      for  $y \in P \ \& \ y \neq x$  do
        строим полуинтервал  $\langle x, x^? \rangle$  такой, что  $y \in \langle x, x^? \rangle$ ;
         $z := \arg \min \{f(u) : u \in \langle x, x^? \rangle \ \& \ u \neq y, y \in \langle x, x^? \rangle\}$ ;
         $S := S \cup z$ ;
      endfor
       $z^* := \arg \min \{f(u) : u \in S\}$ ;
      if  $f(x) > f(z^*)$  then  $x := z^*$ ;  $Q := Q \cup x$  else  $Q := Q \setminus x$ ;
    end
  else {  $x$  – неперспективная точка }
    begin
       $S := \Pi$ 
      for  $y \in Q$  do
        строим полуинтервал  $\langle x, x^? \rangle$  такой, что  $y \in \langle x, x^? \rangle$ ;
         $z := \arg \min \{f(u) : u \in \langle x, x^? \rangle \ \& \ u \neq y, y \in \langle x, x^? \rangle\}$ ;
         $S := S \cup z$ ;
      endfor
       $z^* := \arg \min \{f(u) : u \in S\}$ ;
      if  $f(x) > f(z^*)$  then
         $x := z^*$ ;  $Q := Q \cup x$ ;
      endif
    end
  until  $Q := \Pi$ 
   $x := \arg \min \{f(u) : u \in P\}$ ;
return  $x$ ;
end

```

Рисунок 3 – Схема простого МДМ

На основе сочетания идей МДМ и G -алгоритмов предлагается схема гибридного алгоритма (рис. 4), который будем называть гибридным алгоритмом МДМ (ГМДМ) [11].

Здесь G_Search – процедура G -алгоритма.

Главные параметры алгоритма:

m – число вершин многогранника;

k – количество пар вершин, которые выбираются для исследования (проведения полуинтервала от текущей точки к максимально удаленной и поиска минимумов на этих прямых);

s – радиус окрестности $L_s(y)$, точки которой на построенном полуинтервале удаляются из рассмотрения ($s>0$);

x – точка, максимально удаленная в пространстве решений от начальной точки x .

```

procedure ГМДМ ( $x$ );
begin
   $h := 0$ ;  $P^0 := \emptyset$ ;
  for  $i:=1$  to  $m$  do
     $x :=$  некоторый начальный вариант решения;
     $x :=$  G_Search ( $x$ );
     $P := P \cup x$ ;
  endfor; { завершено формирование начального многогранника  $P$  }
  repeat
     $P := P^h$ ;
    for  $i:= 1$  to  $k$  do
      Отбор для вариации ( $x, y \in P$ );
      строим полуинтервал  $\langle x, x^{\circ} / \& f(x) > f(y) \rangle$ ;
       $z := \arg \min \{ f(u) : u \in \langle x, x^{\circ} / \setminus L_s(y), y \in \langle x, x^{\circ} / \rangle \}$ ;
       $z :=$  G_Search ( $z$ );
       $P := P \cup z$ ;
    endfor; { сформирован многогранник из  $(m+k)$  точек }
     $P^{h+1} :=$  ОтборПопуляции ( $P$ );
     $h := h+1$ ;
  until выполняется условие завершения
   $x := \arg \min \{ f(u) : u \in P \}$ ;
return  $x$ ;
end

```

Рисунок 4 – Схема гибридного алгоритма

Кроме числа точек в популяции m , остальные параметры могут изменяться от итерации к итерации, то есть определяться динамически с учетом хода процесса поиска решений.

Как и в алгоритмах МВС, АИО или G -алгоритмах, можно избежать повторного вычисления значений целевой функции: стоит вычислять лишь разницу $\Delta = f(x) - f(y)$, учитывая тот факт, что отрезок составляют лишь соседние точки – как уже отмечалось, во многих случаях вычисление разницы имеет существенно более низкую трудоемкость.

Для параллельного алгоритма ГМДМ после оператора отбора для вариации пары $(x, y) \in P$ операторы построения полуинтервала, нахождения на нем минимального значения и его оптимизация с помощью функции **G_Search** выполняются на k процессорах кластера.

Результаты вычислений

Изложенные алгоритмы были реализованы на языке C++ на многопроцессорном вычислительном комплексе СКИТ-1, разработанном в Институте кибернетики им. В.М. Глушкова НАН Украины. Основные характеристика кластера СКИТ-1: шестнадцатиузловой кластер, состоящий из 32 процессоров, типа 32-разрядный Intel Xeon с тактовой частотой 2,67 ГГц (кэш – 512 Кбайт), имеющих оперативную память 1 Гбайт.

Проверка эффективности описанных выше алгоритмов осуществлялась путем решения серии реальных задач КЗН, взятых из известной интернет-библиотеки [12].

Начальное приближение находилось методом Монте-Карло при 10 бросаниях, затем все эти 10 вариантов передавались всем итерационным алгоритмам. Приводимые в табл. 1 данные касаются решения четырех реальных задач, размерности матриц которых равны 12, 25, 30 и 36.

В эксперименте использовались два варианта последовательного алгоритма ГМДМ: с выбором $m=10$ вершин и с $m=16$ – в последнем случае алгоритм обозначался ГМДМ(1). Используются обозначения: f^* – оптимальное значение целевой функции, t – время счета ГМДМ в с, t_l – время выполнения ГМДМ(1) на СКИТ-1 с использованием одного процессора, t_8 – время выполнения параллельного алгоритма, соответствующего ГМДМ(1) и обозначенного ГМДМ(8) для $m=16$ вершин, $\varepsilon = (f - f^*)/f^*$, где f – среднее значение, найденное данным алгоритмом, K_n – коэффициент ускорения.

Как видим, наивысшая точность при допустимых временных затратах была продемонстрирована на однопроцессорном ЭВМ гибридным МДМ. Использование параллельного алгоритма ГМДМ на 8 процессорах позволяет получить средний коэффициент ускорения, равный 6,65 (естественно, что здесь учтены затраты на диспетчеризацию). Повышение этого коэффициента с ростом размерности можно объяснить тем фактом, что между процессорами передаются всегда две перестановки длины n . Поскольку общий объем вычислений в алгоритме имеет более высокий порядок, то удельный вес таких затрат снижается с ростом n .

На рис. 5 представлена точность ε алгоритмов в процентах, т.е. $\varepsilon = \varepsilon * 100$ (%), где ε – величина из табл. 1.

Таблица 1

n	f^*	МВС		АИО		G-алгоритм		ГМДМ		ГМДМ(1)		ГМДМ(8)	
		ε	t	ε	t	ε	t	ε	t	ε	t_l	t_8	K_n
12	1652	0,014	0	0,015	0,4	0,003	0,9	0	5	0	25	4	6,25
25	3744	0,040	0	0,004	1,4	0,004	7	0,001	34	0,001	64	10	6,40
30	6124	0,033	0	0,005	2,1	0,007	7,4	0,005	14	0,001	169	24	7,04
36	9526	0,111	0,05	0,050	40,7	0,024	128,2	0,010	619	0,011	298	43	6,93

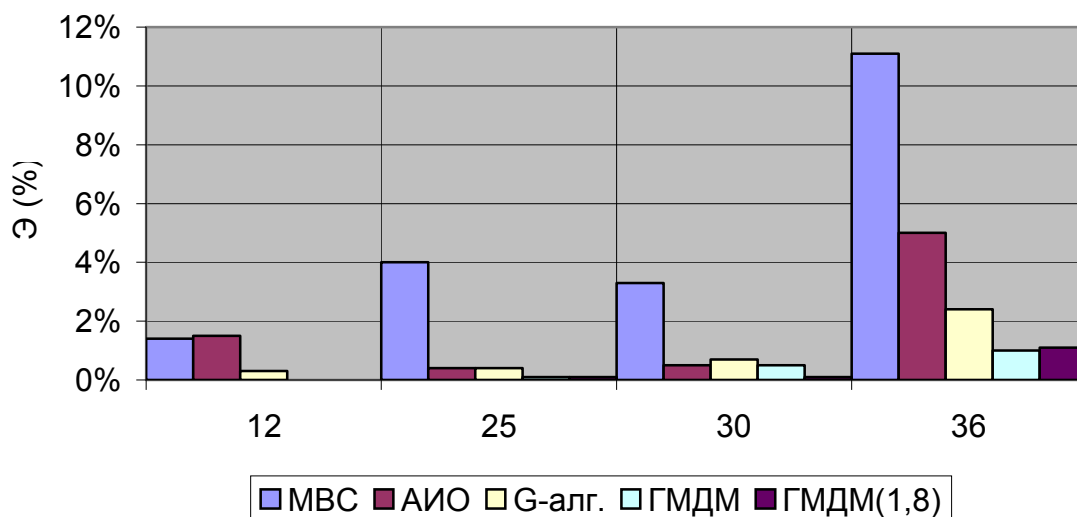


Рисунок 5 – Точность алгоритмов

Заключение

В работе рассмотрены алгоритмы комбинаторной оптимизации, которые могут применяться для решения широкого круга задач КО: метод вектора спада, алгоритм имитационного отжига, G -алгоритм и гибридный метод деформируемых многогранников.

Особое внимание уделено разработке и исследованию параллельного алгоритма ГМДМ. Приведены результаты вычислительного эксперимента по решению КЗН, которые подтверждают эффективность предложенного способа распараллеливания.

Перспективным является исследование теоретических условий сходимости предложенного алгоритма и путей повышения его эффективности в зависимости от конфигурации многопроцессорных вычислительных комплексов.

Литература

1. Burkard R.E. Quadratic assignment problems // Eur. J. of Operation Research. – 1984. – Vol. 15. – P. 283-289.
2. Koopmans T.C., Beckmann M.J. Assignment Problems and the Location of Economic Activities // Econometrica. – 1957. – Vol. 25. – P. 53-76.
3. Gilmore P.C. Optimal and Suboptimal Algorithms for Quadratic Assignment Problem // SIAM J. on Applied Mathematics. – 1962. – Vol. 10. – P. 305-313.
4. Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. Алгоритмы и сложность. – М.: Мир, 1985. – 512 с.
5. Hoos Н.Н., Stützle T. Stochastic Local Search: Foundations and Applications. – San Francisco: Morgan Kaufmann Publ., 2005. – 658 p.
6. Kirkpatrick S., Gelatt C.D., Vecchi M.P. Optimization by Simulated Annealing // Science. – 1983. – Vol. 220, № 4598. – P. 671-680.
7. Гуляницкий Л.Ф. Решение задач комбинаторной оптимизации алгоритмами ускоренного вероятностного моделирования // Компьютерная математика. – 2004. – № 1. – С. 64-72.
8. Химмельблау Д. Прикладное нелинейное программирование. – М.: Мир, 1974. – 534 с.
9. Гуляницкий Л.Ф. Метод деформаций в дискретной оптимизации // Исследование операций и АСУ. – 1989. – Вып. 34. – С. 30-33.
10. Гуляницкий Л.Ф. Один гибридный алгоритм комбинаторной оптимизации // Abs. Int. Ukrainian – Polish Workshop «Problems of Stochastic and Discrete Optimization» (May 10-15, 2005, Kaniv, Ukraine). – Kaniv, 2005. – P. 63-65.
11. Сергиенко И.В., Гуляницкий Л.Ф. Фронтальные алгоритмы для многопроцессорных ЭВМ // Кибернетика. – 1981. – № 6. – С. 1-4.
12. www.opt.math.tu-graz.ac.at/qaplib/inst.html.
13. Roucairol C. A parallel branch-and-bound algorithm for the quadratic assignment problem // Discrete Applied Mathematics. – 1987. – Vol. 18. – P. 211-225.
14. Golden B.L., Skiscim C.C. Using simulated annealing to solve routing and location problem // Naval Research Logistics Quarterly. – 1986. – Vol. 33, № 6. – P. 273-277.

Л.Ф. Гуляницкий, Ю.С. Гложик

Параллельный метод деформируемых многогранников для решения задач комбинаторной оптимизации

Описується алгоритм комбинаторної оптимізації (КО), обґрунтований ідеями методу Нелдера – Міда, названий гібридним методом деформируемых многогранников (ГМДБ). Цей алгоритм носить глобальний характер пошуку у просторі розв'язків задач, що дозволяє знаходити варіанти розв'язків задач із різних класів з підвищеною точністю. У роботі пропонується паралельний алгоритм ГМДБ і обговорюються питання його реалізації на багатопроцесорному суперкомп'ютері СКІТ-1. Дослідження ефективності запропонованого алгоритму здійснено на основі аналізу результатів проведеного обчислювального експерименту розв'язку відомої оптимізаційної моделі – квадратичної задачі про призначення (КЗП).

Статья поступила в редакцию 19.07.2005.