ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Июль—декабрь 2002. Серия 2. Том 9, № 2, 13–30

УДК 519.874

ВЕРОЯТНОСТНЫЙ ПОИСК С ЗАПРЕТАМИ ДЛЯ ДИСКРЕТНЫХ ЗАДАЧ БЕЗУСЛОВНОЙ ОПТИМИЗАЦИИ*)

Е. Н. Гончаров, Ю. А. Кочетов

Предложен новый вариант алгоритма поиска с запретами для решения дискретных задач безусловной оптимизации. Найдены условия на параметры алгоритма, при которых вероятность получения точного решения задачи стремится к единице с ростом числа шагов алгоритма. Проведены экспериментальные исследования поведения алгоритма при решении многостадийной задачи размещения. Показано влияние параметров алгоритма на качество получаемых решений.

Введение

Алгоритм поиска с запретами относится к классу алгоритмов локального поиска [10] и является одним из наиболее эффективных средств для решения задач дискретной оптимизации. Он был предложен Ф. Гловером [15] и хорошо показал себя при решении задач размещения, теории расписаний, календарного планирования и др. Его несомненными достоинствами являются легкая адаптация к сложным моделям, простота и возможность гибридизации с другими методами, например методом ветвей и границ, Лагранжевых релаксаций и т. д. К сожалению, пока мало известно об асимптотических свойствах этого алгоритма, его достоинствах и недостатках с точки зрения вероятности нахождения точного решения задачи.

В настоящей статье предлагается новый вариант алгоритма поиска с запретами, который порождает однородную непериодическую цепь Маркова с конечным множеством состояний. Установлены условия на параметры алгоритма, при которых эта цепь является неразложимой. Данные свойства гарантируют сходимость по вероятности наилучшего найденного алгоритмом решения к глобальному оптимуму. Для исследования реального поведения алгоритма проведены численные

*) Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований (проекты 01–07–90212, 02–01–01153).

© 2002 Гончаров Е. Н., Кочетов Ю. А.

эксперименты на трудных в вычислительном отношении примерах многостадийной задачи размещения. Показано влияние на качество получаемых решений таких параметров алгоритма, как степень рандомизации окрестности, длина списка запретов и стратегий выбора новых областей для повышения эффективности поиска.

Работа организована следующим образом. В § 1 приводится общая схема алгоритма. В § 2 устанавливаются асимптотические свойства. В § 3 приводятся варианты вероятностного алгоритма и их сравнение с детерминированным алгоритмом. В § 4 рассматривается многостадийная задача размещения, показано сведение к ней двухуровневой многостадийной задачи размещения. В § 5 приводятся результаты численных экспериментов. В последнем параграфе обсуждаются направления дальнейших исследований.

§ 1. Вероятностный поиск с запретами

Рассмотрим задачу минимизации целевой функции $F(x)$ на гиперкубе $E^n = \{0,1\}^n$: найти

$$\min\{F(x) \mid x \in E^n\}.$$

Обозначим через $d(x,y)$ расстояние Хемминга между x и y. Через $N_k(x)$ обозначим окрестность точки x радиуса k, т. е.

$$N_k(x) = \{y \in E^n \mid d(x,y) \leqslant k\}, \quad k = 1,2,\ldots,n.$$

При $k = n$ множество $N_k(x)$ любой точки x совпадает с E^n и нахождение в окрестности точки с минимальным значением целевой функции равносильно решению исходной задачи. Поэтому далее будут рассматриваться только малые значения $k = 1,2$ как наиболее удобные для локального поиска, хотя все последующие рассуждения можно обобщить и на случай произвольного $k < n$. Стандартный алгоритм локального спуска начинает свою работу с произвольно выбранной точки x^0. На t-м шаге алгоритма осуществляется переход из текущей точки в соседнюю точку, которая имеет минимальное значение целевой функции в окрестности данной точки, т. е.

$$F(x^{t+1}) = \min\{F(y) \mid y \in N_k(x^t)\}.$$

Алгоритм заканчивает работу в точке локального оптимума, когда $F(x^{t+1}) = F(x^t)$. При решении задач дискретной оптимизации типична ситуация, когда число локальных оптимумов велико и только один из них является глобальным оптимумом

$$F_{\text{opt}} = \min\{F(y) \mid y \in E^n\}.$$

Для того чтобы алгоритм не останавливался в точке локального оптимума, а *путешествовал* от одного локального оптимума к другому, из окрестности удаляется центральная точка и при поиске минимума применяется следующее правило. Пусть $k = 2$ и при переходе от x^{t-1} к x^t менялись значения в координатах (i_t, j_t). Алгоритм хранит такие пары за последние l шагов и на следующем шаге запрещает движение в данных направлениях. Упорядоченный список таких пар

$$\varphi^t = \{(i_t, j_t), (i_{t-1}, j_{t-1}), \ldots, (i_{t-l+1}, j_{t-l+1})\}$$

называется списком запретов. По построению в списке запретов все пары разные, и пара (i, j), $i \neq j$, не запрещает движение по парам $(i, i), (j, j)$ и наоборот. Для удобства будем считать, что $i_t \leqslant j_t$. Если на t-м шаге менялось значение только одной координаты, то $i_t = j_t$. При $k > 2$ аналогичным образом строятся тройки координат, четверки и т. д. Список запретов удаляет из окрестности $N_k(x^t)$ ровно l точек. Множество незапрещенных точек обозначим через $N_k(x^t, \varphi^t)$. Для того чтобы поиск был эффективным, целесообразно использовать малые значения l и управлять данным параметром в ходе работы алгоритма. Ниже будут приведены границы для l, при которых вероятностный алгоритм поиска с запретами находит глобальный оптимум за достаточно большое число шагов.

Обозначим через $N_k(x^t, \varphi^t, p)$ вероятностную окрестность, которая получается из детерминированной окрестности $N_k(x^t, \varphi^t)$ следующим способом. Каждая точка $y \in N_k(x^t, \varphi^t)$ с вероятностью p включается в окрестность $N_k(x^t, \varphi^t, p)$ независимо от других точек. Заметим, что с ненулевой вероятностью это множество может оказаться пустым или содержать только одну точку. Теперь общая схема алгоритма поиска с запретами может быть представлена следующим образом.

Вероятностный алгоритм поиска с запретами

1. Выбирается $x^0 \in E^n$ и полагается $F_A := F(x^0)$, $\varphi^0 := \varnothing$, $t := 0$.
2. Выполняется, пока не сработал критерий остановки.
   2.1. Формируется окрестность $N_k(x^t, \varphi^t, p)$.
   2.2. Если $N_k(x^t, \varphi^t, p) = \varnothing$, то полагается $x^{t+1} := x^t$, иначе находится такая точка x^{t+1}, что $F(x^{t+1}) = \min\{F(y) \mid y \in N_k(x^t, \varphi^t, p)\}$.
   2.3. Если $F(x^{t+1}) < F_A$, то меняется значение $F_A := F(x^{t+1})$.
   2.4. Обновляется список запретов φ^t и счетчик $t := t + 1$.

В качестве критерия остановки может использоваться требуемая точность по отношению к заданной нижней или верхней оценке глобального оптимума, остановка по общему числу шагов либо числу шагов, в ходе которых не меняется значение F_A. Величины p и l являются управляющими параметрами алгоритма. Их выбор зависит от специфики

оптимизационной задачи и ее размерности. Варианты этой схемы и их адаптацию к различным NP-трудным задачам можно найти, например, в [15, 16].

§ 2. Асимптотические свойства алгоритма

Пусть $l = 0$ и $p < 1$. В этом случае за счет рандомизации окрестности алгоритм имеет шанс *покинуть* локальной оптимум даже без списка запретов. Выбор точки x^{t+1} зависит только от текущей точки x^t и не зависит от предыстории, т. е. последовательность $\{x^t\}$ является цепью Маркова с конечным множеством состояний E^n. Для любых двух точек x, y существует положительная вероятность перехода из x в y за конечное число шагов. Следовательно, цепь Маркова является неразложимой и

$$\Pr\{F_A = F_{\text{opt}}\} \longrightarrow 1 \text{ при } t \to \infty.$$

Пусть $l > 0$. Обозначим через Ω множество всех пар (x, φ) таких, что $x \in E^n, \varphi$ — список запретов. Это множество конечно, и переход от одной пары к другой также не зависит от предыстории, т. е. последовательность $\{(x^t, \varphi^t)\}$ также является конечной цепью Маркова.

Теорема 1. *При любых l и p таких, что $0 < l < n(n-1)/4$ и $0 < p < 1$, вероятностный алгоритм поиска с запретами порождает неразложимую цепь Маркова.*

Доказательство. Пусть $(x, \xi) \in \Omega$ и $(y, \phi) \in \Omega$. Построим путь от (x, ξ) до (y, ϕ) такой, чтобы на каждом шаге вдоль этого пути не нарушались ограничения из списка запретов. Такой путь можно построить, например, следующим образом (рис. 1).

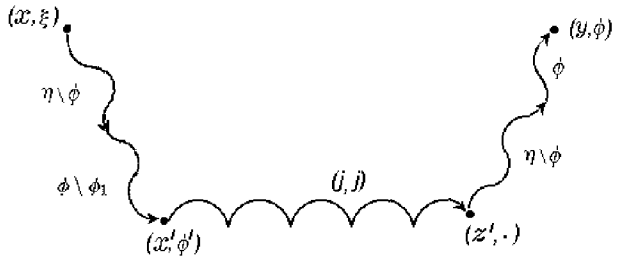

Рис. 1. Путь от (x, ξ) до (y, ϕ)

Обозначим через η множество всех пар (i, j), $i < j$, и положим $\phi_1 = \phi \setminus \eta$. На первом этапе осуществим переход из точки x в точку x' со списком запретов ϕ', удовлетворяющим условию $\eta \supseteq \phi' \supseteq \phi \setminus \phi_1$. Для этого сначала достаточно сделать l шагов по парам из $\eta \setminus \phi$, а затем не более l шагов по $\phi \setminus \phi_1$. На втором этапе, меняя на каждом шаге

только одну координату, переходим в точку z, которая при четном n отличается от точки $1-y$ (при нечетном n от точки y) по координатам из списка ϕ_1. Наконец, на последнем этапе осуществляем переход из точки z в заданную точку y, получив в конце пути список ϕ. Для этого достаточно выполнить все шаги из множества $\eta \setminus \phi$, а затем все шаги в соответствии со списком ϕ, используя его в обратном порядке. Так как при выполнении всех шагов из η переходим из z в z при нечетном n и в $1-z$ при четном n, то в итоге получаем требуемый путь. Теорема 1 доказана.

Следствие. *При любом $x^0 \in E^n$ и любых l и p таких, что $0 < l < n(n-1)/4$ и $0 < p < 1$, справедливы утверждения*:

1) $F_A \to F_{\text{opt}}$ *почти наверное при* $t \to \infty$;

2) *существуют такие константы b и c, $b > 1$ и $0 < c < 1$, что* $\Pr\{F_A \neq F_{\text{opt}}\} \leqslant bc^t$;

3) *цепь Маркова $\{(x^t, \varphi^t)\}$ имеет единственное стационарное распределение $\pi(x, \varphi) > 0$.*

Доказательство. Первое и второе утверждения непосредственно следуют из свойства неразложимости цепи и того факта, что величина F_A монотонно не возрастает с ростом числа шагов алгоритма. Чтобы проверить третье утверждение, достаточно заметить (см. [3, гл. 12]), что цепь (x^t, φ^t) является непериодической, так как при $N_k(x^t, \varphi^t, p) = \varnothing$ имеем $x^{t+1} = x^t$. Следствие доказано.

Первое утверждение следствия фактически означает, что алгоритм найдет оптимальное решение за достаточно большое число шагов. Нельзя утверждать, что последовательность $\{x^t\}$ сходится к оптимальному решению, как это имеет место для алгоритма имитации отжига [9]:

$$\lim_{t \longrightarrow \infty} \Pr\{x^t \in X_{\text{opt}}\} = 1,$$

где X_{opt} — множество оптимальных решений. Однако для практических целей это не имеет большого значения, так как всегда можно взять лучшее решение F_A из последовательности $\{F(x^\tau)\}$, $\tau \leqslant t$. Вариант поиска с запретами, обладающий указанным свойством, можно найти в [14]. Следует отметить, что этот вариант использует другой способ рандомизации и концептуально близок к схеме имитации отжига.

Второе утверждение гарантирует сходимость величины F_A по вероятности к оптимальному решению задачи со скоростью геометрической прогрессии. По-видимому, в общем случае нельзя получить нетривиальные оценки на константу c, не делая никаких предположений относительно целевой функции [19]. В частных случаях, учитывая специфику задачи, это удается сделать, правда, для более простых алгоритмов случайного блуждания [18].

Третье свойство гарантирует, что алгоритм порождает эргодическую цепь Маркова с положительным стационарным распределением $\pi(x, \varphi)$. Отметим, что этим свойством обладают многие вероятностные алгоритмы. Например, алгоритм, на каждом шаге которого случайным образом с равномерным распределением выбирается очередная точка в E^n. Для него $\pi(x) = 1/|E^n| = 2^{-n}$, $x \in E^n$. Поэтому сам факт существования положительного стационарного распределения не является новым. Удивительно то, насколько сильно эта величина для вероятностного поиска с запретами отличается в точке глобального оптимума от 2^{-n}. Численные эксперименты на дискретных задачах размещения дают величины порядка $10^{-3} - 10^{-4}$ при $n = 100$ на сложных классах данных. Для метрических задач той же размерности эта величина имеет порядок 10^{-2}. Таким образом, вероятностный поиск с запретами действительно имеет большие шансы найти глобальный оптимум.

§ 3. Варианты алгоритма

Выше предполагалось, что величина l является постоянной и не меняется в ходе работы алгоритма. Это создает определенные трудности при реализации схемы, так как заранее неизвестно, какой длины следует брать список запретов.

При малых l алгоритм может начать «*циклить*». При больших l поиск становится малоэффективным. Одно из простых правил регулирования длины списка запретов состоит в следующем. Если на очередном шаге алгоритма окрестность $N_k(x^t, \varphi^t, p)$ оказалась пустой, то в список запретов добавляется фиктивная пара $(0, 0)$. Эта пара ничего не запрещает, но уменьшает число запретов на единицу. Такой вариант алгоритма с саморегуляцией l также обладает указанными асимптотическими свойствами, но уже при любых $l \geqslant 0$.

Другое правило управления списком запретов состоит в следующем [11]. На первом шаге алгоритма полагается $l = 1$. Затем длина списка меняется в интервале от 1 до $l_{\max}$ в зависимости от того, встречалась ли точка x^{t+1} ранее на последних, скажем, L_1 шагах алгоритма. Если да, то величина l увеличивается:

$$l := \min\{l_{\max}, \max\{l + 1; 1, 1l\}\}.$$

Если же за L_2 шагов алгоритма все точки были новыми, то величина l уменьшается:

$$l := \max\{1, \min\{l - 1; 0, 9l\}\}.$$

Хранение и поиск решений осуществляются при помощи хэш-функций [7], которые позволяют выполнять эти операции в среднем за $O(1)$

действий. Проверку списка запретов и его изменение на каждом шаге алгоритма тоже можно осуществлять за $O(1)$ действий. Для этого достаточно хранить список запретов одновременно в двух видах: в виде булевой матрицы и вектора, организованного в виде очереди. Проверку принадлежности списку запретов легко проводить по матрице, а изменение списка — по вектору. Если максимальная длина списка $l_{\max}$ не превышает $n(n-1)/4$ или в список запретов добавляются фиктивные пары при пустой окрестности, то такой вариант алгоритма тоже обладает желаемыми асимптотическими свойствами.

Рассмотрим вариант алгоритма, который использует так называемый *критерий ускорения* (aspiration criterion [15]). Эта модификация состоит в том, чтобы отменять запреты и осуществлять переход в *запрещенную* точку y, если $F(y) < F_A$. Очевидно, что точка y ранее не встречалась и запрет только мешает работе алгоритма. Теперь множество состояний описывается не парой (x^t, φ^t), а тройкой (x^t, φ^t, x_A), где x_A — наилучшее решение, найденное алгоритмом к шагу t. Выбор очередной тройки снова не зависит от предыстории, и алгоритм порождает цепь Маркова.

Отметим, что при других способах рандомизации окрестности указанные асимптотические свойства могут и не иметь место. Например, если брать из окрестности r случайных точек [15]. В частности, для $r = |N_k(x)| - l$ получаем детерминированный алгоритм поиска с запретами. Если список запретов слишком мал, то такой алгоритм находит локальный оптимум и не в состоянии его покинуть. Поэтому требуется достаточно большой список запретов.

Ниже приведен пример задачи минимизации, для которой детерминированный алгоритм поиска с запретами не способен найти оптимальное решение при больших списках запретов.

Пусть $F(x) = -|4 - x_1 - \ldots - x_n|$, где $n \geqslant 10$ и четно, а $x_i \in \{0, 1\}$. Глобальный минимум равен $4-n$. Положим $x^0 = (0, \ldots, 0)$ и рассмотрим поведение детерминированного алгоритма для этого примера. На первом шаге список запретов пуст, т. е. $\varphi^0 = \varnothing$, и требуется найти точку x^1 такую, что $F(x^1) = \min\{F(y) \mid y \in N_2(x^0, \varphi^0)\}$. Таких точек много. Выбираем $x^1 = (1, 0, \ldots, 0)$ как лексикографически максимальную. Пара $(1, 1)$ добавляется в список запретов, $\varphi^1 = \{(1, 1)\}$, так как изменилась только первая координата, $F(x^1) = -3$. На втором шаге получаем $x^2 = (0, 1, 0, \ldots, 0)$, $\varphi^2 = \{(1, 2), (1, 1)\}$, $F(x^2) = -3$. На третьем шаге возвращаемся к начальной точке $x^3 = x^0, \varphi^3 = \{(2, 2), (1, 2), (1, 1)\}$, $F(x^3) = -4$. Алгоритм сделал первый цикл. В дальнейшем он будет возвращаться к x^0 много раз через циклы длины 3 или 4 (табл. 1).

Таблица 1

Поведение детерминированного алгоритма

Циклы	Целевая функция
$(1,1),(1,2),(2,2)$	$-3,-3,-4$
$(3,3),(1,3),(1,4),(4,4)$	$-3,-3,-3,-4$
$\ldots$	$-3,-3,-3,-4$
$(n-1,n-1),(1,n-1),(1,n),(n,n)$	$-3,-3,-3,-4$
$(2,3),(3,4),(2,4)$	$-2,-2,-4$
$(2,5),(3,5),(3,6),(2,6)$	$-2-2,-2,-4$
$\ldots$	$-2-2,-2,-4$
$(2,n-1),(3,n-1),(3,n),(2,n)$	$-2-2,-2,-4$
$\ldots$	$\ldots$
$(n-2,n-1),(n-1,n),(n-2,n)$	$-2,-2,-4$

Легко проверить, что при $l \geqslant |N_2(x^0)|$ алгоритм поместит все пары $(i,j), i \leqslant j$, в список запретов и остановится в точке $x^{n(n+1)/2} = x^0$. Если при пустой окрестности $N_2(x^t,\varphi^t)$ алгоритм добавляет в список φ^t фиктивную пару $(0,0)$, то через $l+1-|N_2(x^t,\varphi^t)|$ шагов первая пара $(1,1)$ окажется свободной и все циклы будут повторяться снова. В противном случае $x^{t+1} = x^t$, где $t \geqslant n(n+1)/2$.

При $l < |N_2(x^0)|$ алгоритм тоже делает циклы при $l = 2+4s$, $s = 0,\ldots,n/2-1$; $l = 2n+1+4s$, $s = 0,\ldots,n/2-2$; и др. (см. табл. 1). Таким образом, для любого l найдется l', $|l-l'| \leqslant 2$, такое, что при длине списка запретов l' детерминированный алгоритм не сможет найти оптимальное решение.

§ 4. Многостадийная задача размещения

Рассмотрим другой менее очевидный пример задачи безусловной оптимизации, который представляет самостоятельный интерес в дискретной теории размещений [1, 4–6, 20]. Обозначим через I множество возможных пунктов размещения предприятий, а через J множество потребителей, которым необходима их продукция. Будем предполагать, что имеется определенный набор типов предприятий и для производства продукции требуется участие предприятий разного типа. Без ограничения общности будем считать, что в каждом пункте $i \in I$ можно разместить только одно предприятие заданного типа. Продукция, прежде чем попасть к потребителю, должна пройти несколько стадий обработки на предприятиях разного типа [5, 20], т. е. заданы технологические цепочки вида $p = \{i_1,\ldots,i_{s(p)}\}$, возможно, разной длины, согласно которым продукция последовательно проходит обработку на предприятиях i_1, $i_2,\ldots,i_{s(p)}$. В одной цепочке предприятия

могут повторяться. Множество различных цепочек обозначим через P и положим $P_i = \{p \in P \mid i \in p\}$, $i \in I$.

Для каждого $i \in I$ считаем известными следующие величины:

$f_i \geqslant 0$ — стоимость открытия предприятия в пункте i;

$c_i \geqslant 0$ — удельная стоимость производства;

$d_{ii'} \geqslant 0$ — удельная стоимость транспортировки продукции из пункта i в пункт i'.

Через $D_{pj} \geqslant 0$ обозначим суммарную удельную стоимость транспортировки к потребителю j готовой продукции, изготовленной по технологической цепочке p:

$$D_{pj} = d_{i_1 i_2} + \cdots + d_{i_{s(p)-1} i_{s(p)}} + d_{i_{s(p)} j}.$$

Величина $d_{i_{s(p)}j}$ задает удельные затраты на доставку продукции из последнего пункта $i_{s(p)}$ к потребителю j. Аналогично определим суммарную удельную стоимость производства

$$C_p = c_{i_1} + \cdots + c_{i_{s(p)}}.$$

Объем продукции, необходимый потребителю j, обозначим через φ_j.

Введем следующие переменные:

$$z_i = \begin{cases} 1, & \text{если в пункте } i \text{ открывается предприятие,} \\ 0 & \text{в противном случае,} \end{cases}$$

$$y_{pj} = \begin{cases} 1, & \text{если потребитель } j \text{ получает продукцию } p\text{-й цепочки,} \\ 0 & \text{в противном случае.} \end{cases}$$

Многостадийная задача размещения состоит в отыскании такого варианта размещения предприятий и выбора на их основе подмножества технологических цепочек, чтобы с минимальными суммарными затратами удовлетворить запросы всех потребителей. С использованием введенных обозначений данная задача может быть записана следующим образом [1, 4, 5, 20]: найти

$$\min_{z,y} \left\{ \sum_{i \in I} f_i z_i + \sum_{j \in J} \varphi_j \sum_{p \in P} (C_p + D_{pj}) y_{pj} \right\}$$

при условиях

$$\sum_{p \in P} y_{pj} = 1, \quad j \in J,$$

$$\sum_{p \in P_i} y_{pj} \leqslant z_i, \quad i \in I, \quad j \in J,$$

$$y_{pj}, \quad z_i \in \{0, 1\}, \quad i \in I, \quad p \in P, \quad j \in J.$$

Целевая функция определяет суммарные затраты на открытие предприятий, производство продукции и ее доставку потребителям. Первое ограничение гарантирует удовлетворение спроса всех потребителей. Второе ограничение позволяет производить продукцию только на открытых предприятиях.

Сформулированная задача является NP-трудной в сильном смысле и тесно связана с задачей выбора оптимального набора строк пары матриц [1], задачей минимизации полиномов от булевых переменных [2] и двухуровневой задачей выбора номенклатуры изделий [6]. Если каждая цепочка $p \in P$ содержит только одно предприятие, то получаем простейшую задачу размещения [13].

Представим многостадийную задачу размещения в виде задачи безусловной оптимизации. Введем переменные

$$x_p = \begin{cases} 1, & \text{если выпускается продукция по технологической цепочке } p, \\ 0 & \text{в противном случае.} \end{cases}$$

Тогда исходная задача равносильна нахождению минимума функции

$$F(x) = \sum_{i \in I} f_i \max_{p \in P_i} \{x_p\} + \sum_{j \in J} \varphi_j \min_{p \in P} \{C_p + D_{pj} \mid x_p = 1\}.$$

Рассмотрим одно интересное обобщение многостадийной задачи размещения, в котором потребители выбирают продукцию, используя собственную целевую функцию. Пусть величины g_{pj} задают предпочтения потребителя j на множестве P. Будем считать, что $g_{p_1 j} \neq g_{p_2 j}$ при $p_1, p_2 \in P$. Каждый потребитель стремится выбрать продукцию с минимальным значением g_{pj}. Процесс принятия решений проходит в два этапа. Сначала выбираются пункты размещения предприятий и организуется производство по некоторым технологическим цепочкам, а затем потребители исходя из своих предпочтений посылают заказы на производство и доставку продукции. Такая двухуровневая схема принятия решений приводит к следующей математической постановке: найти

$$\min_{z_i \in \{0,1\}} \left\{ \sum_{i \in I} f_i z_i + \sum_{j \in J} \varphi_j \sum_{p \in P} (C_p + D_{pj}) y_{pj}(z_i) \right\}$$

где $y_{pj}(z_i)$ — оптимальное решение задачи:

$$\min_{y_{pj} \in \{0,1\}} \sum_{j \in J} \sum_{p \in P} g_{pj} y_{pj}$$

при условиях

$$\sum_{p\in P} y_{pj} = 1, \quad j \in J,$$

$$\sum_{p\in P_i} y_{pj} \leqslant z_i, \quad i \in I, \quad j \in J.$$

Эту задачу назовем двухуровневой многостадийной задачей размещения.

Теорема 2. *Многостадийная задача размещения и двухуровневая многостадийная задача размещения полиномиально сводятся друг к другу.*

Доказательство. Сведение многостадийной задачи размещения к двухуровневой многостадийной задачи размещения очевидно. Покажем обратное. Следуя [6], для $j \in J$ введем такую перестановку $(p_1, p_2, \ldots, p_n)$, $n = |P|$, что $g_{p_1 j} < g_{p_2 j} < \cdots < g_{p_n j}$, и положим

$$h_{pj} = \varphi_j(C_p + D_{pj}), \quad p \in P,$$

$$a_{p_k j} = h_{p_1 j} + \sum_{l=1}^{k-1} \min\{0; h_{p_{l+1} j} - h_{p_l j}\}, \quad k = 1, \ldots, n,$$

$$b_{p_k j} = \sum_{l=1}^{k-1} \max\{0; h_{p_{l+1} j} - h_{p_l j}\}, \quad k = 1, \ldots, n.$$

Тогда для непустого множества $S \subseteq P$ и такого вектора $y^*_{pj} \in \{0; 1\}$, что

$$y^*_{pj} = \begin{cases} 1, & \text{если } g_{pj} = \min\limits_{p' \in S} g_{p'j}, \\ 0 & \text{в противном случае,} \end{cases} \qquad j \in J,$$

получаем

$$\max_{p\in S} a_{pj} + \min_{p\in S} b_{pj} = \sum_{p\in P} h_{pj} y^*_{pj}.$$

Полагая $f_{pi} = f_i$, если $p \in P_i$ и 0 в противном случае, получаем сведение двухуровневой многостадийной задачи размещения к задаче выбора оптимального набора строк пары матриц

$$\min_{S\subseteq P, S\neq\varnothing} \left\{ \sum_{i\in I} \max_{p\in S} f_{pi} + \sum_{j\in J} \max_{p\in S} a_{pj} + \sum_{j\in J} \min_{p\in S} b_{pj} \right\}.$$

Эквивалентность же задачи выбора оптимального набора строк пары матриц и многостадийной задачи размещения показана в [1, 2]. Теорема 2 доказана.

Таким образом, двухуровневая многостадийная задача размещения также может быть представлена как задача безусловной оптимизации.

§ 5. Вычислительные эксперименты

Для проведения вычислительных экспериментов был сформирован класс задач со следующими параметрами:

$$|J| = 100,\ |I| = 50,\ |P| = 100,\ f_i = 3000,\ \varphi_j = 1,\ C_p = 0,\ s(p) = 4.$$

Матрица D_{pj} формировалась следующим образом. Для каждого $j \in J$ случайным образом определялось подмножество $P(j) \subset P$, $|P(j)| = 10$, технологических цепочек, доступных потребителю j. Затем с помощью псевдослучайной величины δ с равномерным распределением на множестве $\{0, 1, 2, 3, 4\}$ полагалось

$$D_{pj} = \begin{cases} \delta, & \text{если } p \in P(j), \\ \infty & \text{в противном случае.} \end{cases}$$

Для данного класса задач было сформировано 30 примеров (http://www.math.nsc.ru/AP/benchmarks/index.html). Методом ветвей и границ [4] в каждом примере найдено точное решение. Класс задач оказался достаточно сложным, так как в нем, в частности, содержится нетривиальная подзадача о покрытии. Среднее значение разрыва двойственности

$$\Delta = 100\%(F_{\text{opt}} - F_{LP})/F_{\text{opt}},$$

где F_{LP} — оптимальное значение линейной релаксации, для этих задач составило около 45%. Значительный разрыв двойственности предопределил большое число просматриваемых вершин в дереве ветвлений, в среднем около 10^6. Указанный класс значительно сложнее классов из [1, 4, 5] как для точных методов, так и для вероятностных жадных алгоритмов и итерационных методов локального поиска.

Для сокращения трудоемкости одного шага алгоритма рассматривалась более узкая окрестность

$$N(x) = N_1(x) \cup N'(x) \subset N_2(x),$$

где $N'(x) = \{y \in N_2(x) \mid d(y, 0) = d(x, 0)\}$. Все эксперименты проводились для алгоритма с этой окрестностью и добавлением в список запретов фиктивной пары $(0, 0)$, если рандомизированная окрестность оказывалась пустой.

Первый вычислительный эксперимент связан с поиском наилучших значений p для разных длин списка запретов. Обозначим через $\varepsilon(T)$ среднюю относительную погрешность

$$\varepsilon(T) = 100\%(F_A - F_{\text{opt}})/F_{\text{opt}}$$

после выполнения T шагов алгоритма. Изменение погрешности $\varepsilon(T)$ для одной из тестовых задач при $l = 0, 10, 50$ изображено на рис. 2.

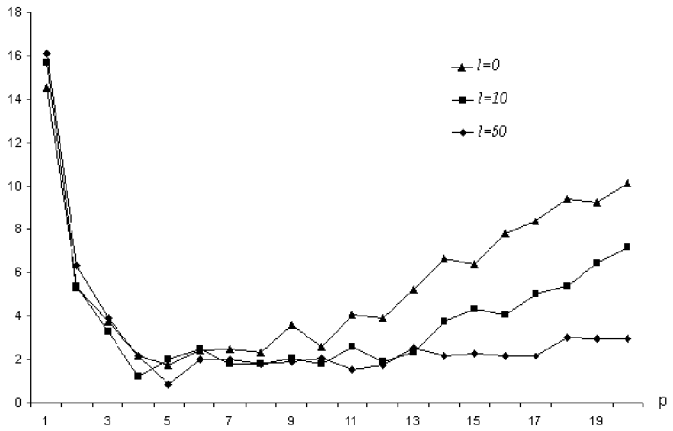

Рис. 2. Изменение $\varepsilon(T)$ при $T = 20000$

Каждая точка на графике показывает среднюю погрешность за 50 испытаний. Наилучшее значение p^* лежит в интервале $[3\%, 9\%]$. Для сравнения $\varepsilon(T) = 17,4\%$ при $p = 100\%$, $l = 50$. Время счета на PC Pentium II 800 MHz составило 152 секунды. При рандомизации $p = 10\%$ время счета составило 21 секунду. Следовательно, детерминированный алгоритм поиска с запретами дает в среднем худшее решение и затрачивает на его нахождение больше времени.

Аналогичная картина наблюдается и на остальных примерах данного класса. За 20 тысяч шагов детерминированный алгоритм в 95 случаях из 100 давал большую погрешность, чем вероятностный аналог. По-видимому, такая картина должна иметь место для сложных примеров, хотя интуитивно кажется, что чем сложнее задача, тем тщательнее надо просматривать окрестность и всегда лучше просматривать всю окрестность, чем ее часть. Тем не менее это не так, по крайней мере для многостадийной задачи размещения. Для простых задач, когда любой локальный оптимум является глобальным, детерминированный алгоритм будет скорее всего выигрывать у вероятностного алгоритма. На сложных же задачах, в которых много локальных оптимумов и они непредсказуемым образом разбросаны по допустимой области, преимущество, как подсказывает наш опыт, должно быть на стороне вероятностных алгоритмов.

Заметим, что относительная погрешность невелика при $l = 0$. Другими словами, вероятностный алгоритм локального поиска без списка запретов позволяет находить достаточно хорошие решения при малых значениях p. Заметим, что влияние списка запретов уменьшается при уменьшении p. Таким образом, рандомизация окрестности сокращает относительную погрешность, время счета и влияние списка запретов.

Второй вычислительный эксперимент связан с критерием ускорения (aspiration criterion) и его влиянием на относительную погрешность. Напомним, что согласно этому критерию разрешается игнорировать список запретов, если в окрестности текущей точки найдено лучшее значение целевой функции, чем F_A. Интуитивно понятно, что такой вариант алгоритма может иметь определенные преимущества, однако численные эксперименты на указанном классе исходных данных не подтверждают это предположение. По-видимому, такое значение оказывается запрещенным крайне редко либо алгоритм многократно возвращается к нему с разными списками запретов и в итоге все равно находит эту точку или равноценную ей.

Третий вычислительный эксперимент связан с гипотезой *большой долины* [8, 12]. Согласно этому предположению в среднем локальные оптимумы значительно ближе к глобальному оптимуму, чем случайно выбранная точка. Распределение локальных оптимумов не является равномерным. Существует определенная концентрация локальных оптимумов в малой части допустимой области, которую образно называют *горным массивом* (massive central) для задач на максимум или *большой долиной* (big valley) для задач на минимум. Если это предположение верно, то при решении задачи целесообразно запоминать разные точки с небольшим отклонением от значения F_A, т. е. формировать *элиту*, и время от времени выбирать новую стартовую точку между точками элиты.

Это правило лежит в основе оператора кроссовера (crossover) для генетических алгоритмов [17]. Точнее, новая стратегия выбора стартовой точки заключается в следующем. В ходе работы алгоритма формируется *популяция* решений, для которых значение целевой функции незначительно отличается от значения F_A. Это множество строится так, чтобы все решения были различны и расстояние Хемминга между ними было достаточно велико. В ходе численных экспериментов это расстояние было не меньше 5, размер популяции 10. При смене стартовой точки из этого множества равновероятно выбирались два решения (родители). На их основе строилось новое решение, каждая компонента которого с равной вероятностью принимала значение соответствующей компоненты одного из родителей (uniform crossover). Затем к полученному решению применялся вероятностный оператор *мутации*, который с заданной вероятностью (0,05) менял значение каждой компоненты на противоположное $x_i \to 1 - x_i$. В дальнейшем такую стратегию выбора стартовой точки будем называть генетическим стартом.

На рис. 3 изображена частота нахождения оптимального решения для следующих трех стратегий:

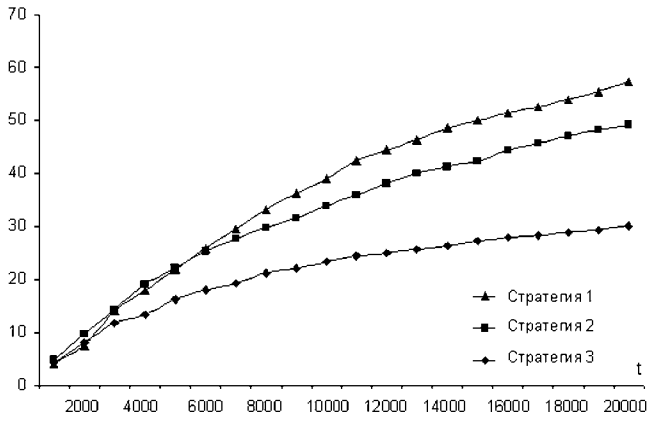

Рис. 3. Частота получения точного решения для $l = 50$

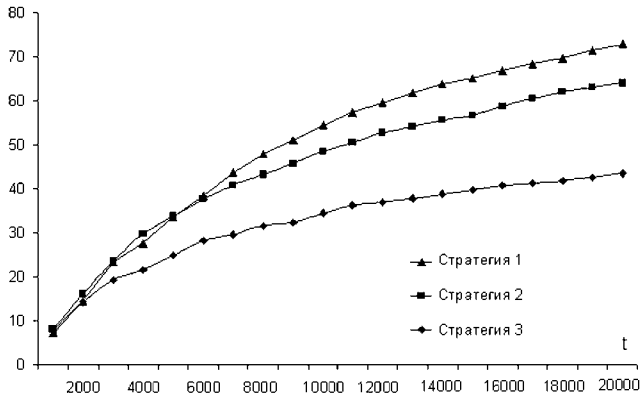

Рис. 4. Частота получения погрешности $\varepsilon \leqslant 1\%$ для $l = 50$

1) генетический старт;

2) выбор случайной стартовой точки, если рекорд F_A не меняется в течение 500 шагов;

3) стандартная стратегия без смены стартовой точки.

Частота получения с помощью указанных стратегий приближенного решения с погрешностью не более 1% изображена на рис. 4. Каждая точка на этих графиках показывает среднюю величину по 30 испытаниям в 30 тестовых примерах.

В табл. 2 приведены доверительные интервалы для вероятности получения за 20 тысяч шагов алгоритма точного решения и приближенного решения с погрешностью не более 1%. Коэффициент доверия 0,95.

Таблица 2

Доверительные интервалы

Стратегия	$\varepsilon = 0$	$\varepsilon \leqslant 1\%$
1	$0,541 - 0,605$	$0,701 - 0,759$
2	$0,460 - 0,524$	$0,610 - 0,672$
3	$0,272 - 0,330$	$0,403 - 0,467$

Рис. 3 и 4 свидетельствуют о преимуществе стратегии, основанной на гипотезе большой долины. По-видимому, данный эксперимент можно рассматривать как косвенное доказательство концентрации локальных оптимумов для исследуемого класса исходных данных.

§ 6. Направления дальнейших исследований

Выше рассматривались варианты алгоритма, когда порог рандомизации окрестности был фиксирован. Увеличение порога ведет к более тщательному исследованию области решений, близких к x^t. При уменьшении порога алгоритм стремится сменить область поиска. Правильное управление этим параметром может повысить эффективность алгоритма и решить проблему выбора оптимальной рандомизации поиска. Разработка адаптивной стратегии управления этим параметром представляется важным направлением на пути совершенствования данного подхода.

Другим интересным направлением исследований является поиск наиболее трудных классов задач для локального поиска. Если число локальных оптимумов растет экспоненциально с ростом размерности задачи [21], то для локального поиска задача становится достаточно трудной. Интересно найти другие трудные случаи и дать их классификацию, например, с точки зрения распределения локальных оптимумов. Генерация трудных примеров и объяснение, почему эти примеры оказываются трудными для разных подходов, способствуют лучшему пониманию природы NP-трудных задач и совершенствованию алгоритмов их решения.

ЛИТЕРАТУРА

1. **Береснев В. Л., Гимади Э. Х., Дементьев В. Т.** Экстремальные задачи стандартизации. Новосибирск: Наука, 1978.
2. **Береснев В. Л., Гончаров Е. Н.** Приближенный алгоритм для задачи минимизации полиномов от булевых переменных // Дискрет. анализ и исслед. операций. Сер. 2. 1998. Т. 5, № 2. С. 3–19.

3. **Боровков А. А.** Теория вероятностей. М.: Наука, 1986.

4. **Гончаров Е. Н.** Метод ветвей и границ для простейшей двухуровневой задачи размещения предприятий // Дискрет. анализ и исслед. операций. Сер. 2. 1998. Т. 5, № 1. С. 19–39.

5. **Гончаров Е. Н., Кочетов Ю. А.** Поведение вероятностных жадных алгоритмов для многостадийной задачи размещения // Дискрет. анализ и исслед. операций. Сер. 2. 1999. Т. 6, № 1. С. 12–32.

6. **Горбачевская Л. Е., Дементьев В. Т, Шамардин Ю. В.** Двухуровневая задача стандартизации с условием единственности оптимального потребительского выбора // Дискрет. анализ и исслед. операций. Сер. 2. 1999. Т. 6, № 2. С. 3–11.

7. **Кормен Т., Лейзерсон Ч., Ривест Р.** Алгоритмы: Построение и анализ. М.: МЦНМО, 1999.

8. **Кочетов Ю. А.** Вероятностные методы локального поиска для задач дискретной оптимизации // Дискретная математика и ее приложения: Сб. лекций молодежных и научных школ по дискретной математике и ее приложениям. М.: Изд-во центра прикл. исслед. при мех.-мат. фак. МГУ, 2001. С. 84–117.

9. **Aarts E. H. L., Korst J. H. M., Laarhoven P. J. M.** Simulated annealing // Local search in combinatorial optimization. Chichester: John Wiley & Sons, 1997. P. 91–120.

10. **Aarts E. H. L., Lenstra J. K.** Local search in combinatorial optimization. Chichester: John Wiley & Sons, 1997.

11. **Battiti R., Protasi M.** Reactive local search for maximum clique // Proc. of Workshop on Algorithm Engineering, 1997. P. 74-82. (http://www.dsi.unive.it/wae97/proceedings/contents.html)

12. **Boese K. D., Kahng A. B., Muddu S.** A new adaptive multi-start technique for combinatorial global optimizations // Oper. Res. Lett. 1994. V. 16, N 2. P. 101–113.

13. **Krarup J., Pruzan P.** The simple plant location problem: survey and synthesis // European J. Oper. Res. 1983. V. 12, N 1. P. 36–81.

14. **Faigle U., Kern W.** Some convergence results for probabilistic tabu search // ORSA J. Comput. 1992. V. 4, N 4. P. 32–37.

15. **Glover F., Laguna M.** Tabu search. Boston: Kluwer Acad. Publ., 1997.

16. **Hertz A., Taillard E., de Werra D.** Tabu search // Local search in combinatorial optimization. Chichester: John Wiley & Sons, 1997. P. 121–136.

17. **Reeves C. R.** Genetic algorithms for the operations researcher // INFORMS J. Comput. 1997. V. 9, N 3. P. 231–250.

18. Schuler R., Schöning U., Watanabe O. An improved randomized algorithm for 3-SAT // Techn. Rep., Dept. of Math. and Comput. Sci. Tokyo Inst. of Technology, 2001.

19. Schumacher C. Black box search — framework and methods. Ph. D. Thesis. The Univ. of Tennessee, Knoxville, USA, 2000.

20. Tcha D. W., Lee B.-I. A branch-and-bound algorithm for the multilevel uncapacitated facility location problem // European J. Oper. Res. 1984. V. 18, N 1. P. 35–43.

21. Tovey C. A. Hill climbing with multiple local optima // SIAM J. Algebraic and Discrete. Methods. 1985. V. 6, N 3. P. 384–393.

22. Wolpert D. H., Macready W. G. No free lunch theorem for search // Techn. Rep. SFI-TR-95-02-010. Santa Fe Inst., 1995.

Адрес авторов:

Институт математики
им. С. Л. Соболева СО РАН,
пр. Академика Коптюга, 4,
630090 Новосибирск,
Россия

Статья поступила
29 апреля 2002 г.