УДК 519.85

И.В. Гребенник, Л.Ю. Юрченко

Харьковский национальный университет радиоэлектроники, Харьков

УПОРЯДОЧЕНИЕ ПЕРЕСТАНОВОК ПРИ РЕШЕНИИ ЗАДАЧ КОМБИНАТОРНОЙ ОПТИМИЗАЦИИ С ЛИНЕЙНОЙ ЦЕЛЕВОЙ ФУНКЦИЕЙ

Рассматривается задача оптимизации линейной функции с линейными ограничениями на множестве перестановок. Предлагается алгоритм генерации перестановок в порядке убывания значений линейной целевой функции. Проводится обоснование алгоритма, даются оценки его временной и емкостной сложности. Анализируются примеры.

комбинаторная оптимизация, множество перестановок, линейная функция, упорядочение перестановок, перестановочный многогранник

Введение

Задачи комбинаторной оптимизации возникают при математическом моделировании и решении многих научных и прикладных задач, возникающих в проектировании, управлении и многих других областях [1 – 3]. Широкое применение комбинаторные оптимизационные модели находят в геометрическом проектировании [4]. Несмотря на разнообразие методов комбинаторной оптимизации [1 – 4], не существует эффективного универсального метода решения задач оптимизации на комбинаторных множествах.

Комбинаторные оптимизационные задачи геометрического проектирования относятся к классу NP-полных задач. Применение для их решения известных методов локальной и глобальной оптимизации и разработка новых эффективных методов, основанных на использовании свойств комбинаторных оптимизационных моделей, является актуальной проблемой. Поэтому необходимо продолжение исследований в области модификации существующих и разработки новых эффективных методов комбинаторной оптимизации.

Один из подходов к созданию методов комбинаторной оптимизации основан на генерации упорядоченных комбинаторных конфигураций. Известны различные способы генерации элементов комбинаторных множеств в заданном порядке, например, лексикографическом [5, 6]. В [7] предлагается достаточно общий подход, позволяющий генерировать упорядоченные комбинаторные конфигурации, опираясь на свойство периодичности. Это позволяет использовать один и тот же подход для генерации элементов различных типов комбинаторных множеств. Альтернативой указанному способу упорядочения комбинаторных конфигураций является упорядочение по значениям целевой функции задачи комбинаторной оптимизации. Это даст возможность строить последовательности приближений к решению в методах комбинаторной оптимизации.

Целью настоящей работы является разработка и обоснование подхода к генерации перестановок в заданном порядке при решении задач комбинаторной оптимизации.

Постановка задачи

Рассмотрим задачу комбинаторной оптимизации на множестве перестановок, отображенном в евклидово пространство, в следующей постановке:

$$L(x) = \sum_{i=1}^{n} \alpha_i x_i \to \min ; \qquad (1)$$

$$Cx \le d ; \qquad (2)$$

$$x \in E_{nk} \subset R^n , \qquad (3)$$

где $x_i \ge 0$; $C = [C_{ij}]_{r \times n}$; $C_{ij} \in R$; $\alpha_i \in R$; $i \in J_n = \{1, 2, \ldots, n\}$; $j \in J_r$; $d \in R^r$; E_{nk} – множество векторов в R^n, координаты которых представляют собой перестановки из n элементов $a_1 \le a_2 \le \ldots \le a_n$, k из которых предполагаются различными.

Выпуклая оболочка множества E_{nk} называется перестановочным многогранником [3, 8]: $\operatorname{conv} E_{nk} = \Pi_{nk}$. При этом множество E_{nk} совпадает с множеством вершин многогранника Π_{nk}: $E_{nk} = \operatorname{vert} \Pi_{nk}$. Сформулированы критерий вершины и критерий смежности вершин перестановочного многогранника [3, 8].

Подходы к решению задач комбинаторной оптимизации вида (1) – (3) рассмотрены в ряде работ [9, 10]. Решением задачи (1) – (3) является такая перестановка $x^* \in E_{nk}$, которая доставляет минимальное значение линейной целевой функции при выполнении линейных ограничений.

Как известно [3], безусловный минимум линейной функции $L(x)$ на множестве E_{nk} достигается в точке $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n) \in E_{nk}$, такой что $\bar{x}_{l_j} = a_j$. При этом последовательность $\{l_1, l_2, \ldots, l_n\}$ такова, что $\alpha_{l_1} \geq \alpha_{l_2} \geq \ldots \geq \alpha_{l_n}$. Наличие ограничений (2) чаще всего делает точку $\bar{x}$ недопустимой.

С целью разработки метода решения задачи (1) – (3) построим процедуру генерации элементов множества E_{nk} в порядке возрастания значений линейной функции вида (1). Результатом применения такой процедуры должно стать множество элементов:

$$X = \{x^1, x^2, \ldots, x^N\}; \; L(x^i) \leq L(x^{i+1}), \qquad (4)$$

где $x^i \in E_{nk}$; $i \in J_{N-1}$; $N \in J_t$; $t = n!$; $x^1 = \bar{x}$.

Обоснование алгоритма упорядочения перестановок

Рассмотрим произвольно выбранную перестановку $x^i \in E_{nk}$ – вершину перестановочного многогранника Π_{nk}. Эта вершина может быть точкой максимума x^0 или точкой минимума $\bar{x}$ функции $L(x)$ на множестве E_{nk} или некоторой промежуточной вершиной. В случае, если $x^i \neq x^0$ и $x^i \neq \bar{x}$, то среди n – 1 вершин, смежных с x^i, найдется хотя бы одна вершина x^h, для которой $L(x^h) \geq L(x^i)$, и хотя бы одна x^l, для которой $L(x^l) \leq L(x^i)$. Справедливость этого утверждения следует из свойства монотонности линейной функции на многограннике.

Предположим, что первые m упорядоченных перестановок из множества X, удовлетворяющих условию (4), уже построены. Тогда $m+1$-я перестановка в множестве X принадлежит множеству вершин многогранника Π_{nk}, смежных с построенными m вершинами.

Докажем это. Предположим противное: существует вершина $\tilde{x} \in \Pi_{nk}$, не смежная ни с одной из первых m вершин множества X, и при этом она является $m+1$-м элементом множества X, удовлетворяющим (4). Другими словами, в вершине $\tilde{x} \in \Pi_{nk}$ достигается наименьшее значение функции $L(x)$ среди $t-m$ вершин, не содержащихся среди первых m элементов множества X.

Из сказанного выше следует, что если вершина $\tilde{x} \in \Pi_{nk}$ не является точкой минимума $\bar{x}$ функции $L(x)$ на множестве E_{nk}, то существует хотя бы одна смежная с ней вершина многогранника € $\in \Pi_{nk}$, в которой значение целевой функции будет меньше, то есть $L($€$) \leq L(\tilde{x})$. Но если вершина $\tilde{x}$ не смежна ни с одной из первых m вершин множества X, то, следовательно, вершина €, не входит в данные m и в ней значение целевой функции меньше, чем $L(\tilde{x})$. Но тогда в вершине $\tilde{x} \in \Pi_{nk}$ не достигается наименьшее значение функции $L(x)$ среди $t-m$ вершин, не содержащихся среди первых m элементов множества X. Мы пришли к противоречию. Следовательно, исходное утверждение верно, и $m+1$-я перестановка в множестве X принадлежит множеству вершин многогранника Π_{nk}, смежных с построенными m вершинами.

Приведенные рассуждения могут быть положены в основу алгоритма генерации перестановок в порядке возрастания значений линейной целевой функции задачи (2).

Алгоритм генерации упорядоченных перестановок

Построим алгоритм, который генерирует перестановки в порядке увеличения значения линейной функции, удовлетворяющие условию (4). Для работы алгоритма сформируем два списка, один из которых назовем списком результатов, а другой – списком претендентов. Алгоритм состоит из следующих шагов:

1. Генерация перестановки $\bar{x}$ с наименьшим значением линейной функции $L(x)$ и помещение ее в конец списка результатов.

2. Для перестановки, которая находится в конце списка результатов, то есть была сгенерирована последней, нахождение всех ее смежных вершин. Помещение тех из найденных вершин, которых еще нет в списке результатов и в списке претендентов, в список претендентов.

3. Выбор из списка претендентов наилучшей перестановки (с наименьшим значением функции $L(x)$), и добавление ее в конец списка результатов.

4. Если сгенерировано заданное количество перестановок N, то выход из алгоритма, иначе – переход к шагу (2).

Результат действия алгоритма – множество X – содержится в списке результатов, в котором перестановки упорядочены по возрастанию значения линейной функции. На каждой итерации алгоритма может быть получена следующая в таком списке вершина, тогда работу алгоритма можно прерывать не по достижению некоторого количества сгенерированных вариантов, а по какому-либо другому условию, например при получении первой перестановки, которая бы удовлетворяла ограничениям задачи (1) – (3).

Оценки вычислительной сложности алгоритма

Количество вычислительных ресурсов, необходимых для работы алгоритма, зависит от размерности задачи n и от количества N упорядоченных перестановок, которые следует сгенерировать.

Объем необходимой оперативной памяти должен включать память для хранения всего списка результатов и списка претендентов. Учитывая, что в список результатов все данные только перемещаются по мере необходимости из списка претендентов, достаточно просто определить количество данных, помещаемых в список претендентов. На каждой итерации выбирается новый элемент – вершина перестановочного многогранника Π_{nk}, который переносится в список результатов. При этом в список претендентов заносятся все вершины, смежные с выбранной. Их количество не превышает $n-1$ [3, 8], при этом одна из смежных вершин та, по которой мы «пришли» в данную. Следовательно, в список претендентов отправится не более $n-2$ вершин. Кроме того, уже на второй итерации среди смежных к данной попадаются вершины, которые уже есть в списке претендентов. Количество добавляемых вершин уменьшается с увеличением номера итерации алгоритма. Таким образом, верхняя оценка *емкостной сложности* алгоритма имеет вид:

$$R_{mem} = N \cdot (n-2) \cdot n \cdot s,$$

где N – количество сгенерированных перестановок; s – объем памяти, необходимый для хранения одного числового значения.

Например, для генерации 1000 лучших вариантов из множества перестановок 20-ти четырехбайтных значений потребуется около 1,4 Мб оперативной памяти.

В процессе выполнения алгоритма рассчитывается значение функции $L(x)$ для всех $N \cdot (n-2) \cdot n$ вариантов, а на каждое вычисление значения функции приходится минимум *n* операций умножения и $n-1$ операция сложения. Кроме этого на каждой итерации выполняются такие действия:

– выбор каждой смежной вершины перестановочного многогранника занимает 2 операции пересылки, а всего смежных вершин не более $(n-1)$;

– добавление каждой из смежных вершин (количество которых не превышает $(n-1)$) в вектор, отсортированный по значению функции $L(x)$. При этом перед добавлением выполняется проверка на вхождение в один сортированный вектор, и добавление в другой, также с проверкой вхождения. Эта операция на каждой i-й итерации имеет сложность $O(2\log_{10} i)$, $i \in J_N$. Для достижения такой стандартной для бинарного поиска трудоемкости массивы элементов должны содержать не только сами элементы, но и сохраненные значения целевой функции. Для грубой оценки сверху можно записать, что сложность этого этапа – $O(2\log_{10} N)$;

– перенос элемента с конца одного сортированного списка в конец другого сортированного списка – $n+1$ операций пересылки, так как элементом массива является перестановка из n элементов и значение целевой функции.

Таким образом, оценка *временной сложности* алгоритма определяется как:

$$R_{cpu} = n \cdot N \cdot (n-2) \cdot (2n-1) +$$
$$+ N[(n-1) \cdot (2 + 2\log_{10} N) + n + 1]$$

или

$$R_{cpu} = 2n^3 N - 5n^2 N + 5nN + (2n-2)\log_{10} N - 1.$$

Следовательно, временная сложность алгоритма полиномиально зависит от размерности задачи n и числа генерируемых вариантов N и может быть оценена как $O(n^3 N)$.

Примеры

Рассмотрим некоторые примеры результатов работы алгоритма.

1. Результат генерации перестановок из множества E_4, порожденного элементами $\{1,2,3,4\}$, в порядке возрастания значений линейной функции $L(x) = 9x_1 + 8{,}4x_2 + 7x_3 + 5{,}95x_4$ представлен в табл. 1.

Таблица 1

Результат работы алгоритма при $n = 4$, $N = 24$

Номер	Перестановка	Значение
1	(1;2;3;4)	70,6
2	(2;1;3;4)	71,2
3	(1;2;4;3)	71,65
4	(1;3;2;4)	72
5	(2;1;4;3)	72,25
6	(3;1;2;4)	73,2
7	(2;3;1;4)	74
8	(1;3;4;2)	74,1
9	(1;4;2;3)	74,45
10	(3;2;1;4)	74,6
11	(3;1;4;2)	75,3
12	(1;4;3;2)	75,5
13	(4;1;2;3)	76,25
14	(2;4;1;3)	76,45
15	(2;3;4;1)	77,15
16	(4;1;3;2)	77,3
17	(4;2;1;3)	77,65
18	(3;2;4;1)	77,75
19	(2;4;3;1)	78,55
20	(3;4;1;2)	79,5
21	(4;2;3;1)	79,75
22	(4;3;1;2)	80,1
23	(3;4;2;1)	80,55
24	(4;3;2;1)	81,15

2. Для перестановок из множества E_{22}, порожденного числами $\{1,2,...,22\}$, сгенерировать N элементов, упорядоченных по возрастанию линейной функции, так чтобы все они соответствовали условию (4). Для сгенерированных перестановок построить график зависимости значения целевой функции от номера перестановки. На рис. 1 представлен результат генерации 1000 перестановок из множества E_{22}, удовлетворяющих условию (4). Коэффициенты функции $L(x)$ являются элементами множества {56; 55,5; 55; 54,3; 51; 50,5; 47; 46; 40; 39,5; 39; 38; 36,23; 35,6; 30; 25,6; 23,2; 21; 19,5; 17,8; 15,4; 10,2}. Процесс генерации занял меньше минуты на процессоре *Pentium4M* с тактовой частотой 1,7Ghz.

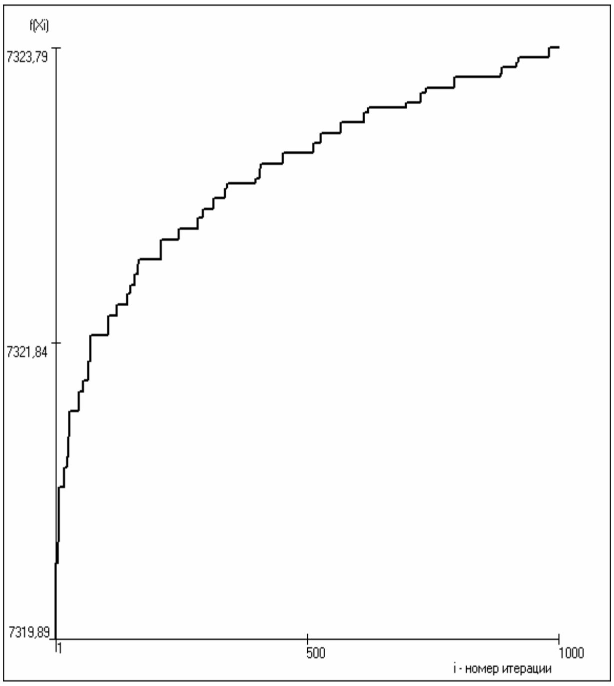

Рис. 1. Результат генерации 1000 перестановок из 22-х элементов

Выводы

Применение предложенного в работе алгоритма позволяет генерировать упорядоченные по значению линейной функции перестановки в количествах, определяемых наличными временными ресурсами.

Проведенные вычислительные эксперименты демонстрируют, что на больших размерностях можно достаточно быстро получить 100-1000 вариантов для рассмотрения.

Генерацию перестановок в порядке возрастания значения целевой функции задачи комбинаторной оптимизации можно положить в основу методов комбинаторной оптимизации. При этом необходимо дальнейшее развитие предложенного алгоритма.

В первую очередь, устранение его главного недостатка – этот алгоритм может генерировать последовательность перестановок с возрастающим значением целевой функции, только начиная с перестановки с наименьшим значением линейной целевой функции.

Более полезным был бы алгоритм, который позволяет генерировать для данной перестановки следующую или предыдущую перестановку в порядке возрастания значения линейной функции.

Список литературы

1. Сергиенко И.В. Математические модели и методы решения задач дискретной оптимизации. – К.: Наук. думка, 1988. – 472 с.

2. Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация: Алгоритмы и сложность. – М.: Мир, 1985. – 512 с.

3. Стоян Ю.Г., Ємець О.О. Теорія і методи евклідової комбінаторної оптимізації. – К.: Інститут системних досліджень освіти, 1993 – 188 с.

4. Стоян Ю.Г., Яковлев С.В. Математические модели и оптимизационные методы геометрического проектирования. – К.: Наук. думка, 1986. – 268 с.

5. Липский В. Комбинаторика для программистов. – М.: Мир, 1988. – 213 с.

6. Ruskey F. Combinatorial generation. – Dept. of Computer Science Univ. of Victoria, Canada, 1j-CSC 425/520, 2003.– 289 p.

7. Тимофеева Н.К. Упорядочение множества значений аргумента целевой функции в комбинаторной оптимизации // Кибернетика и системный анализ. – 1998. – № 6. – С. 78-87.

8. Емеличев В.А., Ковалёв М.М., Кравцов М.К. Многогранники, графы, оптимизация. – М:. Наука, 1981. – 344 с.

9. Яковлев С.В., Валуйская О.А. Оптимизация линейных функций на вершинах перестановочного многогранника с дополнительными линейными ограничениями // Український матем. журн. – 2001. – № 9. – С. 1278-1280.

10. Ємець О.О., Ємець Є.М. Відсікання в лінійних частково комбінаторних задачах евклідової оптимізації // Доповіді НАН України. – 2000.– № 9. – С. 105-109.

Поступила в редколлегию 17.07.2007

Рецензент: д-р техн. наук, проф. Т.Е. Романова, Институт проблем машиностроения НАНУ, Харьков.