第16卷第2期 数学研究与评论 Vol. 16 No. 2
1996年5月 JOURNAL OF MATHEMATICAL RESEARCH AND EXPOSITION May 1996

# 最优消除顺序的B&B算法*

黄有度
(合肥工业大学数力系，合肥 230009)

**摘 要** 讨论了最优节点消除顺序的性质，并给出了计算最优消除顺序的B&B算法.

**关键词** 图，最优节点消除顺序，B&B算法
**分类号** AMS(1991) 05C85/CCL O157.5

## 一 引 言

[1]从一个离散最优化问题的算法复杂性出发，提出了图的节点最优消除顺序问题. 这个问题具有重要的实用价值和理论意义. [1]研究了最优消除顺序的一些性质并提出了计算最优消除顺序的最小次算法和最小欠数算法. 但是这些算法对有些图是无效的. 本文作了进一步讨论并给出了有效的B&B算法.

## 二 对节点最优消除顺序的进一步讨论

给定一个具有 $n$ 个节点的无向图 $G$，设 $y$ 为 $G$ 的一个节点，用 $N_G(y)$ 表示在 $G$ 中与 $y$ 相邻的顶点集合. 图 $G$ 的节点消除运算为：先在 $G$ 中去掉一个节点 $y_1$ 及其关联边，并把 $N_G(y_1)$ 中所有不相邻的点连接起来，这样得到的图记为 $G*y_1$，并把此过程称为在 $G$ 中消除 $y_1$. 在消除 $y_1$ 时，消除的边数 $|N_G(y_1)|$ 记为 $n_1$. 然后，再从 $G*y_1$ 中消除一个节点 $y_2$，消除的边数 $|N_{G*y_1}(y_2)|$ 记为 $n_2$. 这样一直做下去，直到全部节点都被消除. 记消除顺序为 $\alpha=(y_1,y_2,\cdots,y_n)$，相应于 $y_i$ 消除的边数为 $n_i, i=1,2,\cdots,n$. 记 $T_1(\alpha)=\max_i\{n_i\}$. 若把 $n_i$ 从大到小排列，设为 $(d_1,d_2,\cdots,d_n)$，记 $T_2(\alpha)=(d_1,d_2,\cdots,d_n)$. 以 $\mathscr{D}$ 表示 $n$ 个节点的所有排列的集合，则图 $G$ 的节点最优消除顺序问题是寻求一个消除顺序 $\alpha$，使

$$T_1(\alpha)=\min_{\beta\in\mathscr{D}}T_1(\beta)$$

或

$$T_2(\alpha)=\mathrm{L}\min_{\beta\in\mathscr{D}}T_2(\beta),$$

其中L min表示字典序最小. 用 $T(G)$ 表示 $\min_{\beta\in\mathscr{D}}T_1(\beta)$，显然有 $T(G)\leqslant n-1$. 而且，最优消除顺

* 1993年10月2日收到. 95年4月收到修改稿.

序一般并不唯一.

**命题 1** 设图 $G$ 有 $n$ 个节点，则 $T(G)=n-1$ 的充要条件是 $G$ 为完全图(结果显然，证略).

**命题 2** 设具有 $n$ 个节点的图 $G$ 不是完全图，则 $T(G)=n-2$ 的充要条件是 $A=\{y:y\in G,|N_G(y)|\leqslant n-3\}$ 在 $G$ 中的导出子图 $G(A)$ 是完全图.

**证明** 必要性 设 $G(A)$ 不是完全图，则有 $y_i,y_j\in A$，但 $y_j\notin N_G(y_i)$. 先消除 $y_i,y_j$，因为 $|N_G(y_i)|\leqslant n-3$，$|N_{G*y_i}(y_j)|\leqslant n-3$，在 $G*y_i*y_j$ 中还有 $n-2$ 个节点，$T(G*y_i*y_j)\leqslant n-3$，所以有 $T(G)\leqslant n-3$.

充分性 设 $G(A)$ 是完全图，$(y_1,y_2,\cdots,y_n)$ 为一个最优消除顺序. 先证 $T(G)\geqslant n-2$. 若 $y_1\notin A$，即 $|N_G(y_1)|\geqslant n-2$，则已有 $T(G)\geqslant n-2$. 若 $y_1\in A$，设 $y_i$ 与 $y_j$ 为 $G$ 的另外任意两个节点，易证 $y_i$ 与 $y_j$ 在 $G*y_1$ 中必有相连，于是 $G*y_1$ 为完全图. 由命题 1，$T(G*y_1)=n-2$，所以也有 $T(G)\geqslant n-2$. 由于 $G$ 非完全图，由命题 1 有 $T(G)<n-1$. 故 $T(G)=n-2$. 这就证明了命题 2.

类似可得下一个命题：

**命题 3** 给定具有 $n$ 个节点的图 $G$，$G(A)=G(\{y:y\in G,|N_G(y)|\leqslant n-3)$ 非完全图，则 $T(G)=n-3$ 的充要条件是：对 $G$ 中任意满足 $|N_G(y)|\leqslant n-4$ 的节点 $y$，$G(\{y_i:y_i\in G*y,|N_{G*y}(y_i)|\leqslant n-4\})$ 是完全图.

[1] 中提出的最小次算法是依据其命题 4：

设 $\alpha=(x_1,x_2,\cdots,x_{j-1},x_j,x_{j+1},\cdots,x_n)$ 是图 $G$ 的节点消除顺序，记 $Y=x_1,x_2,\cdots,x_{j-1}$，如果在 $G*Y$ 中 $x_j$ 的次不大于 $x_{j+1}$ 的次，那么节点消除顺序 $\beta=(x_1,x_2,\cdots,x_{j-1},x_{j+1},x_j,x_{j+2},\cdots,x_n)$ 不比 $\alpha$ 好，即 $T_2(\alpha)\leqslant T_2(\beta)$，这里“$\leqslant$”是字典序的小于等于.

但这不足以说明每次消除当时的最小次节点可得最优消除顺序，请看下面例子.

**例 1** 设图 $G$ 如图 1，$|N_G(y_7)|=3$，$|N_G(y_1)|=|N_G(y_3)|=|N_G(y_5)|=4$，$|N_G(y_2)|=|N_G(y_4)|=|N_G(y_6)|=5$. 如果用最小次算法先消除 $y_7$，则 $G*y_7$ 为完全图，$T(G*y_7)=5$. 因此，如果 $\alpha$ 为以 $y_7$ 开头的任一节点消除顺序，有 $T_1(\alpha)=5$. 但由命题 2 有 $T(G)<n-2=5$，而由命题 3 有 $T(G)=4$. 实际上，以 $y_1,y_3,y_5$ 中任两个开头的节点消除顺序都是最优的. 此例说明最小次算法并不总是有效的.

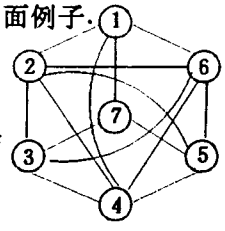

图1

图 $G$ 的一个节点 $y$ 的欠数为 $y$ 的邻点集在 $G$ 中的导出子图变为完全图时所需增加的边数. 图 1 中 7 个节点的欠数都是 3，故最小欠数算法对此图也无效.

## 三 最优消除顺序的 B & B 算法

这里给出一种 B & B(Bound and Branch) 算法[2] 可有效地找出最优消除顺序，此过程表现为构造一棵以 $G$ 为树根的树，树上每一节点表示 $G$ 的节点消除过程中的某一中间步骤所代表的图(如 $G$，$G*y_1$，$G*y_1*y_2$ 等)，树的节点值表示经过此节点的所有消除顺序的值的一个下界. 例如，若节点 $S$ 代表 $G*y_1*y_2$，则 $S$ 的节点值 $T$ 表示所有以 $y_1y_2$ 开头的消除顺序 $T(\alpha)$ 的一个下界.

B & B 算法：

步骤 A（延伸）：已知 $T(G)$ 的一个上界 $T_0$，并考虑某一选定的树上节点 $S$ 及其节点值 $T$.

1° 若 $S$ 的节点数 $|V(S)| \leqslant T+1$，或 $S$ 为完全图，则把节点 $S$ 改为树叶（用方块表示树叶），并且

$$\max\{|V(S)|-1, T\} \Rightarrow T,\ \min\{T, T_0\} \Rightarrow T_0,$$

树叶值取为 $T$，转 5°.

2° 若对 $S$ 中任一节点 $y$，都有 $|N_S(y)| \geqslant T_0$，则把 $S$ 改为树叶，树叶值取为 $T_0+1$，转 5°.

3° 设 $S$ 中 $|N_S(y)| < \min\{T_0, |V(S)|-1\}$ 的节点为 $y_1, \cdots, y_k$，从 $S$ 向下引出 $k$ 条树枝，连到节点 $y_1, \cdots, y_k$，分别代表图 $S*y_1, \cdots, S*y_k$. 在每条树枝旁分别注上树枝值 $|N_S(y_i)|$，节点 $y_i$ 旁注上节点值 $\max\{T, |N_S(y_i)|\}$.

4° 在节点 $y_1, \cdots, y_k$ 中任选一个具有最小节点值者作为 $S$，其节点值作为 $T$，转 1°.

5° 转入步骤 B.

步骤 B（修剪）

1° 剪去所有不连树叶的、树枝值大于等于 $T_0$ 的树枝.

2° 从最新得到的树叶出发，

3° 向树根方向找出第一个节点值小于 $T_0$ 的节点记为 $R$，如找不到，转 6°.

4° 如果 $R$ 下方没有不带树叶的树枝，转 3°.

5° 从 $R$ 下方不带树叶的节点中任选一个节点值最小的节点作为 $S$，其节点值作为 $T$，转入步骤 $A$.

6° 算法结束，这时的 $T_0$ 即为 $T(G)$，树叶值为 $T_0$ 的树叶所代表的一组消除顺序即为最优消除顺序.

开始时，取图 $G$（树根）作为 $S$，$|V(G)|-1 \Rightarrow T_0$，$1 \Rightarrow T$，然后交替使用步骤 A 和 B，直到算法结束.

这个算法可找出 $T(G)$ 及至少一个最优消除顺序. 如果把步骤 A 的 2° 中“$\geqslant$”改为“$>$”，3° 中的“$<$”改为“$\leqslant$”，步骤 B 的 1° 中“$\geqslant$”改为“$>$”，则可找出全部最优消除顺序.

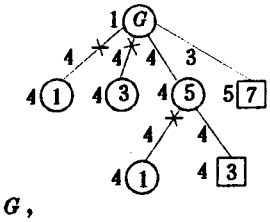

图2

例 2 求例 1 中图 $G$ 的 $T(G)$ 和一个最优消除顺序.

开始，取 $T_0$ 为 6，树根 $G$ 的节点值取为 1，并按照上述算法，依次考虑 $G$，$G*y_7$，$G*y_5$，$G*y_5*y_3$，最后得到 $T(G)=4$，以 $y_5y_3$ 开头的所有消除序列都是最优的.

运算过程由图 2 表示，带“×”的树枝被剪去.

本文的 B & B 算法易于用计算机编程，且由于在计算过程中剪去大量树枝，故计算量不会很大.

## 参 考 文 献

[1] 刘振宏、王常藩，最优消除顺序，系统科学与数学，12(4)(1992)，307—316.

[2] G. L. Nemgauser, L. A. Wolsey, *Interger and Combinatorial Optimization*, John Wiley & Sons, Inc., New York, 1988.

# The B & B Algorithm for Optimal Node-Delecting Orders

*Huang Youdu*

(Heifei University of Technology, Hefei 230009)

**Abstract**

In this paper, the characters of the optimal node-delecting order are further investigated, and a B & B algorithm is presented for finding an optimal node-delecting order.

**Keywords** graph, optimal node-delecting order, B & B algorithm.