## §7.5 完美对集

奇分支和偶分支：奇数个顶点的分支称为奇分支；偶数个顶点的分支称为偶分支。用表示G的奇分支的个数。

定理7.4(Tutte定理)：图G有完美对集当且仅当

证：显然只要对简单图证明这个定理就行了。

首先假设G有完美对集M。设S是V的一个真子集，并设

是的奇分支，因为是奇分支，所以的某一顶点一定在M下和S的一个顶点配对(见图7.6)。所以，由于

，因而有

反之，假设G满足(7.9)式，但没有完美对集，则G是没有完美对集的极大图的生成子图。由于是的生成子图，所以

。因而由(7.9)式有

特别地，置，因而。

用U表示中度为的顶点集。由于当时，显然有完美对集，因此可以假定我们将证明：是完全图的不相交的并图。假若不然，的某一分支不是完全图，则在此分支里存在顶点，使得。此外，由于，在中存在顶点w，使得

表示在图7.7中。

由于是不包含完美对集的极大图，所以对于所有，都有完美对集。设分别是的完美对集， 并且用H表示由导出的的子图。由于H的每个顶点的度均为2，所以H是圈的不相交的并图。进而，由于沿着这些圈的的边和的边是交错的，所以所有这些圈都是偶圈。我们分两种情形讨论：（见图7.8）

情形1：xz和yw在H的不同分支中(图7.8 (a))。于是，若yw在H的圈C中，则在C中的边连同不在C中的边一起组成的一个完美对集，这和的定义矛盾。

情形2：xz和yw在H的同一分支C中。根据x和z的对称性，可以假定顶点依次出现在C中(图7.8 (b))。于是在C的yw节中的边连同yz以及不在C的节中的边一起组成的一个完美对集，再次与的定义矛盾。

因此不论情形1或情形2都导致矛盾，由此可知确是完全图的不相交的并图。

现在，根据(7.10)式有。因此的奇分支个数最多是，但这样一来，显然就有一个完美对集：的各奇分支中的一个顶点和U的一个顶点配对，U的余下的顶点以及的各分支中余下的顶点，则也相互配对，正如图7.9所表示的那样。

由于假定是没有完美对集的，因而得到了希望出现的矛盾。于是G确实有完美对集。

## §7.6 一般图最大对集的算法

1.与偶图类似，从任一对集M出发；若M未饱和G中所有顶点，从任一M非饱和顶点v出发，构造M交错树T。(见图7.10)

x

u

v

图7.10

2.奇点与偶点：

若T中的一个偶点u与T外一个M非饱和顶点x相邻，则中存在一条M可扩路。但奇点不具有这个性质。（见图7.11）

e e e

o o o

e e e e e

o e e o o

e e e

o o o

v e

图7.11 奇点与偶点

3.花：

若T中两个偶点x和y相邻，则在中出现一个奇圈，称为花。

(见图7.11)

\*收缩花(见图7.12，7.13)收缩前如图7.12，收缩后见图7.13。

e e e

o o e

e e e e e y

o e e e e

x

e e e

o o o

v e

图7.12 收缩前

e e

o o

e e e y

o e e x

e e e 伪点

o o o

v e

图7.13 收缩后

4.算法：

算法的基本步骤：

1. 从任一对集M开始；
2. 若M饱和G中所有顶点，则算法停止；
3. WHILE G中还有未处理过的M非饱和点 DO

BEGIN

1. 任取一个未处理过的M非饱和顶点v，将v置为偶点；
2. ；
3. ；
4. WHILE T中有偶点未处理完 DO

BEGIN

1. 取T中一个未处理完的偶点u；
2. IF 且x是T外的M非饱和点 THEN
3. 找到一条M-可扩路，扩展对集得，用代替M,

回到第2步；

1. IF 且x是T外的M饱和点且 THEN
2. 将x标记为奇点，将y标记为偶点；
3. IF 且x是T中的偶点 THEN
4. 找到一个花，将花收缩为一个伪点；
5. IF 且x是T中的奇点 THEN 不做任何动作；

END;

1. 算法找到一棵匈牙利树T，将T保存起来；
2. ；

END;

1. 算法找到最大对集M, 算法终止。

\*匈牙利树的例子：

算法1：MAPS(G) M可扩路查找过程

1. 选一个outer点和一条以前没处理过的边(x,y)。现在把(x,y)

看作已处理了。如果不存在这样的边，转H；

1. 如果y是没标记过的顶点并且是M非饱和的，将(x,y)加入T中，

转A；

1. 如果y是outer点，将(x,y)加入T中，转B；
2. 如果y是inner点，转第1步；
3. 设(y,z)是M中的边，将(x,y), (y,z)加入T中，标记y inner, z outer，

转第1步。

算法2：求最大(基数)对集算法

2. 存在可扩路；
3. WHILE 存在可扩路 DO

BEGIN

1. 确定一个M非饱和点；
2. FOR 每个 DO

BEGIN

1. 将栈STACK置空；
2. 将每一条边看作是未处理过；
3. ，标记 outer；

L: 9. MAPS(G)；

B: 10. 将找到的花压入栈中，在G中收缩它，收缩后的该点

标记为outer，如果它包含树根，标记它M非饱和，

转L；

H: 11. 将找到的匈牙利树压入栈，将它从G中删除。

END；

12. 输出M；

13. 存在可扩路false；

14. 转S;

A: 15. 确定T中M可扩路P，将STACK中弹出的每一个元素在G

中扩展；如果该元素是一个花，对应P中一个伪点，将花中相应的偶长度路(M交错路)插入P中；

16. 通过交换P中的对集边和非对集边，扩展M;

S: 17. END;

例子：见图7.14。