第三章 图的匹配问题

§3.1 匹配与最大匹配

定义3.1.1 设 G 是一个图, $M \subseteq E(G)$，若 M 中的任意两条边在G 中不相邻，则称M 是G 的一个匹配。

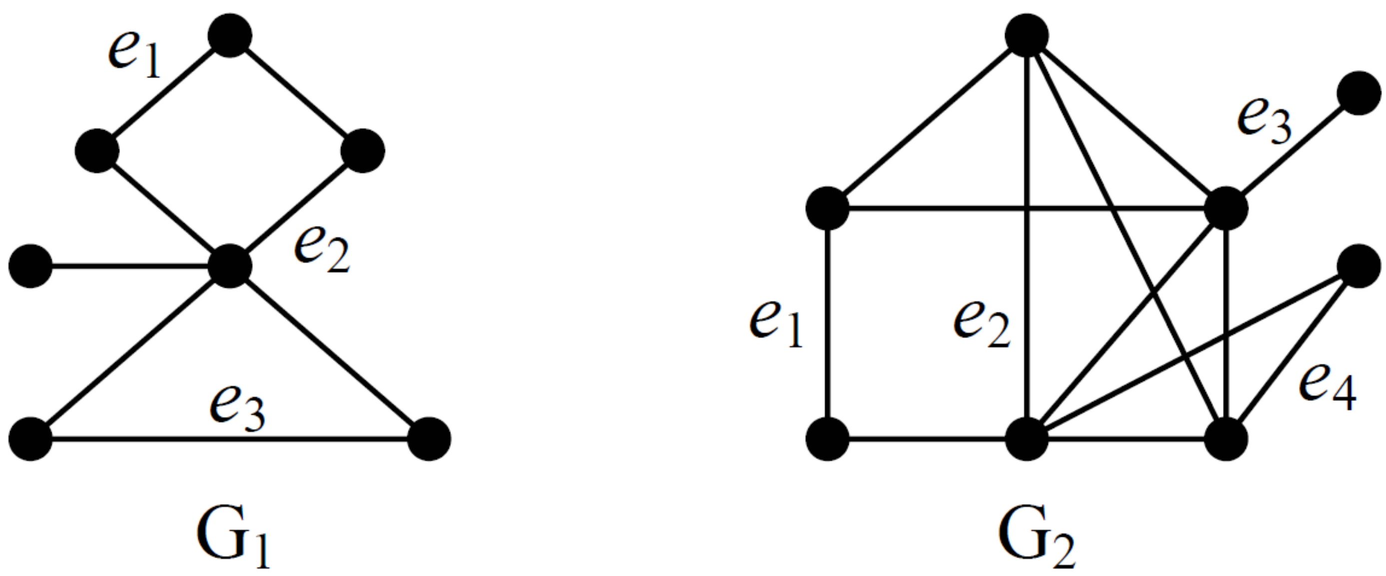
对匹配M 中每条边$e = uv$，其两端点 u 和 v 称为是M 饱和的(saturated vertex)。

注: 每个顶点要么未被M 饱和, 要么仅被M 中一条边饱和。

定义3.1.2 设M 是G 的一个匹配, 若G 中无匹配M'，使得$|M'|>|M|$, 则称M 是G 的一个**最大匹配**；如果G 中每个点都是M 饱和的，则称M 是G 的**完美匹配**(Perfect matching).

完美匹配一定是最大匹配，反之不然。

例如，在下图G_1中，边集$\{e_1\}$、$\{e_1,e_2\}$、$\{e_1,e_2,e_3\}$都构成匹配，$\{e_1,e_2,e_3\}$是G_1的一个最大匹配。在 G_2中，边集$\{e_1,e_2,e_3,e_4\}$是一个完美匹配，也是一个最大匹配。

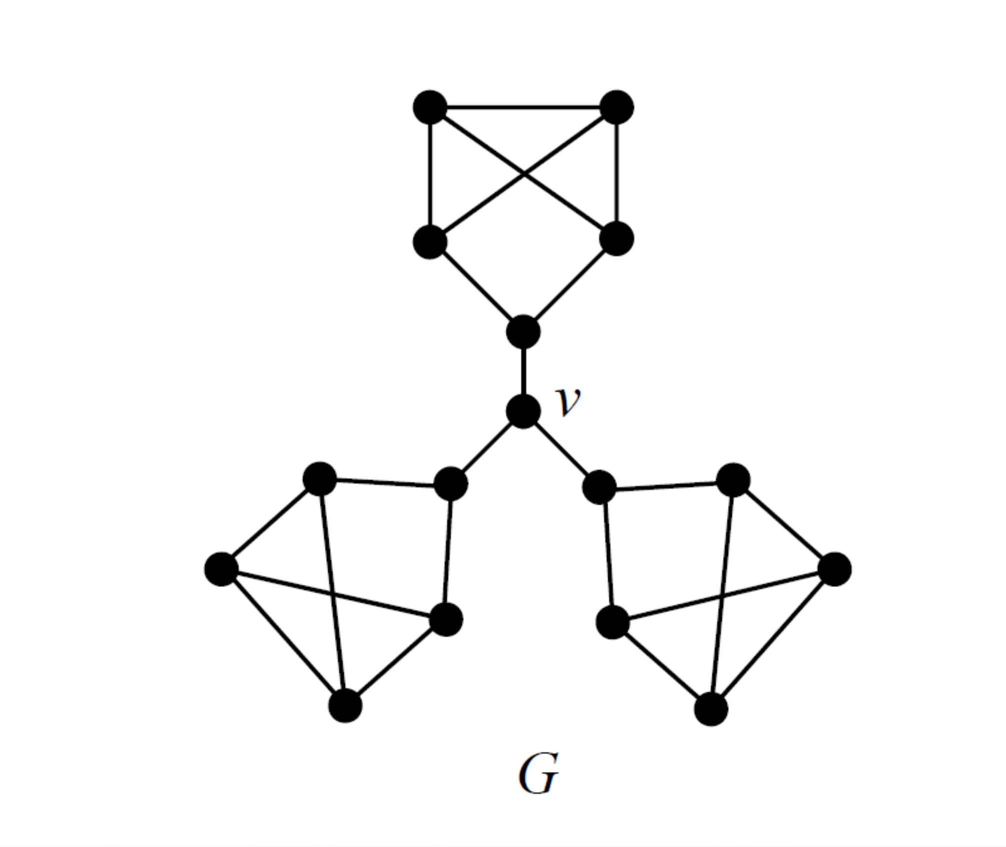


定义3.1.3 设M 是G 的一个匹配，G 的M 交错路(alternating path)是指其边在M 和$E(G)\setminus M$ 中交替出现的路。

如果G 的一条M 交错路的起点和终点都是M 非饱和的，则称其为一条M 可扩展路或M 增广路(augmenting path)。

定理 3.1.1(Berge,1957) 图G 的匹配M 是最大匹配的充要条件是G中不存在M 可扩展路。

§ 3.2 完美匹配



这个图有完美匹配吗？

完美匹配的判断条件是什么？

§3.2 完美匹配

定义 3.2.1 图G 的奇分支：G 的含有奇数个顶点的连通分支。用$O(G)$表示G 的奇分支的个数。

定理 3.2.1 (Tutte,1947) 图G 有完美匹配的充要条件是对$\forall S \subset V(G)$，$O(G \setminus S) \le |S|$。

推论 3.2.1 偶数个顶点的 $(k-1)$边连通k正则图有完美匹配。

推论 3.2.2 偶数个顶点的完全图 K_{2n} 有 $2n-1$个边不重的完美匹配。

思考： 如何找出这个2n-1个完美匹配？

举例

某工厂用六种颜色的纱生产双色布.已知每种颜色的纱至少与其它三种颜色的纱搭配，证明可以选出三种不同的双色布，它们包含了所有的六种颜色.

[证] 用表示六种纱的六个顶点$v_1,v_2,\ldots,v_6$为顶点集 *V*, 作图 $G=(V,E)$，使$v_iv_j\in E$当且仅当第 *i* 和第 *j* 种纱被搭配成一种双色布. 于是，问题转化为已知 *G* 的每个顶点的度至少为 3，要证 *G* 有 3 条边，其中任意两条没有公共端点.

不失一般性, 可设$v_1v_2, v_3v_4\in E$,若$v_5v_6\in E$, 则已得所求证的结论. 故不妨设$v_5v_6\notin E$, 由$d(v_5)\geq 3$, 不妨设$v_1v_5, v_2v_5, v_3v_5\in E$, 于是可分下列三种情形讨论: 情形 1 $v_1v_6\in E$, 则v_1v_6, v_2v_5, v_3v_4为所求.

情形 2 $v_4v_6\in E$， 则v_4v_6, v_1v_2, v_3v_5为所求.

情形 3: v_1、$v_4\notin N(v_6)$. 由$v_5\notin N(v_6)$, 有$d(v_6)\leq 2$, 矛盾.

问题：如何判断一棵树 T 有完美匹配？

定理3.2.1对树能否改进？

§3.3 k-因子

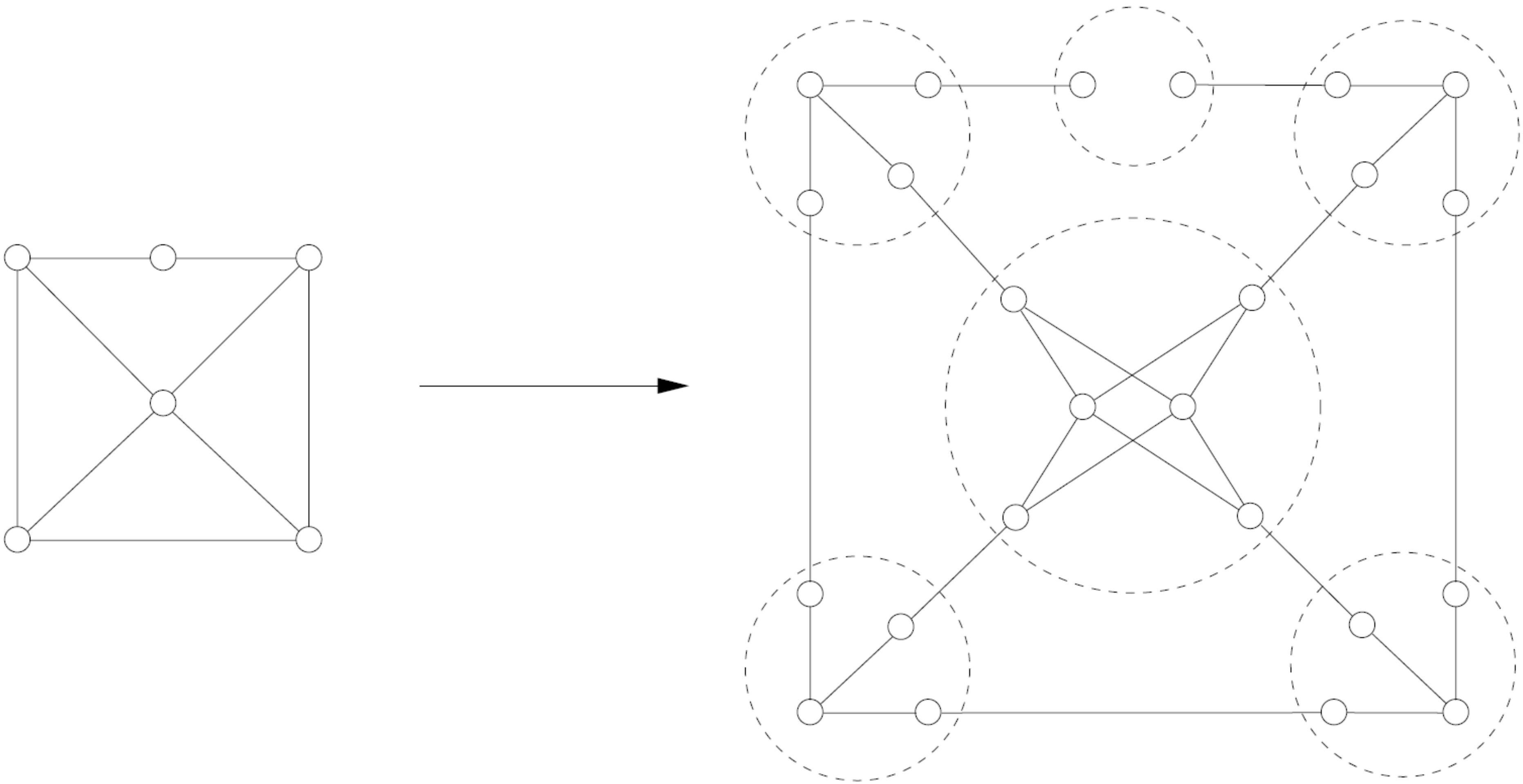
A *k-factor* of *G* is a spanning subgraph F of G with $d_F(v) = k$ for all $v \in V$.

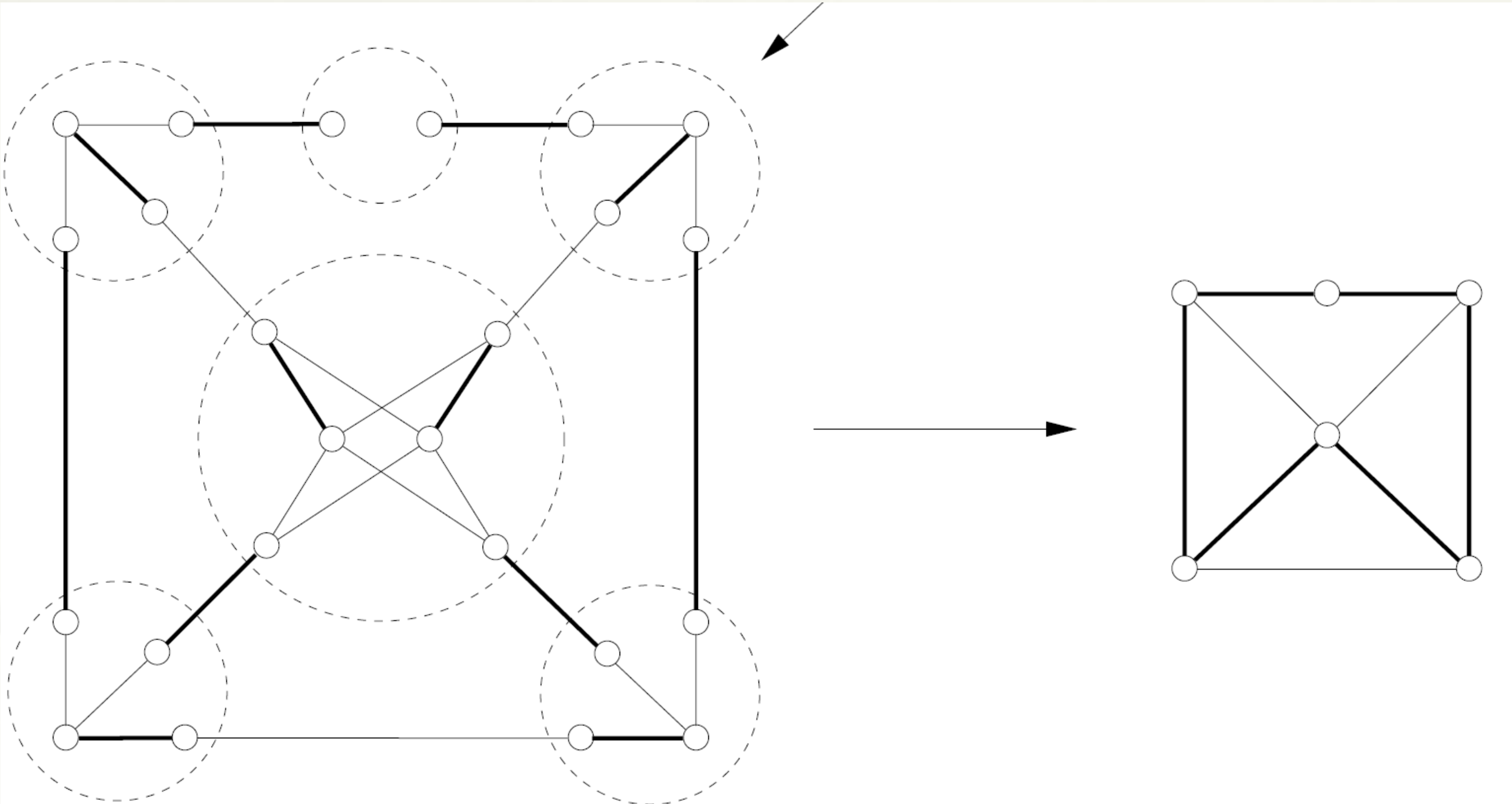
In particular, a 1-factor is a spanning subgraph whose edge set is a perfect matching and a 2-factor is a spanning subgraph whose components are cycles.

Many interesting graph-theoretical problems can be solved in polynomial time by reducing them to problems about 1-factors. One example is the question of deciding whether a given graph *G* has an k-factor.

Reduction of the k-factor problem to the 1-factor problem.

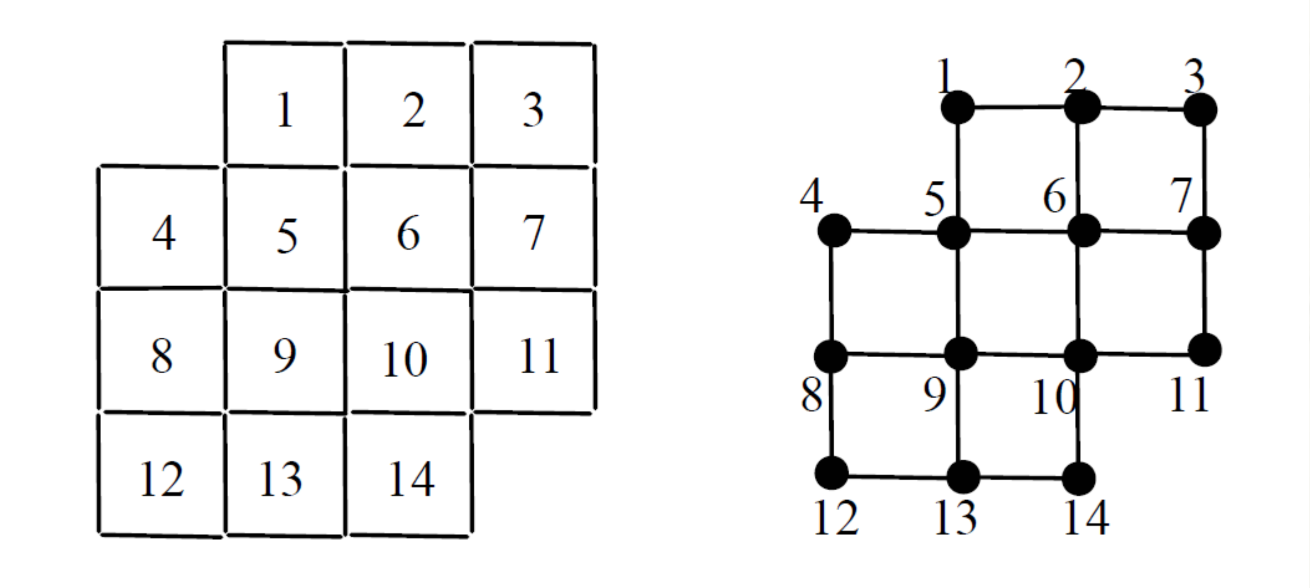
For each vertex v of G, first replace v by a set Yv of $d(v)$ vertices, each of degree one. Then add a set Xv of $d(v)-k$ vertices and form a complete bipartite graph Hv by joining each vertex of Xv to each vertex of Yv. In effect, the resulting graph H is obtained from G by replacing each vertex v by a complete bipartite graph $Hv[Xv, Yv]$ and joining each edge incident to v to a separate element of Yv.





§3.4 二部图的匹配

下图所示的是14个大小相同的正方形组成的图形。试证明：不论如何用剪刀沿着图形中所画的直线对它进行裁剪，总剪不出7个由相邻的两个小正方形组成的矩形来。



定理 3.3.1 (Hall, 1935) 设G 是具有二划分 (X,Y) 的二部图，则 G 有饱和 X 的匹配当且仅当对$\forall S \subseteq X$，$N(S) \geq S$，其中$N(S)$表示S 的所有邻点之集。

推论 3.3.2.(Frobenius,1917) 具有二划分 (X,Y) 的二部图 G 有完美匹配的充要条件是 $|X|=|Y|$ 且对 $\forall S \subseteq X$ (或 Y)，均有 $|N(S)| \geq |S|$。

推论3.3.3 (König,1916) 设 G 是 k 正则二部图($k > 0$)，则 G 有 k 个边不重的完美匹配。

思考：如何找出这个k 个完美匹配？

§3.5 二部图中最大匹配的算法

1.背景与问题

指派问题（assignment problem）：欲安排 n 个人员$x_1, x_2,\ldots,x_n$ 从事n 项工作 $y_1, y_2,\ldots, y_n$。已知每个人能胜任其中一项或几项工作。试问：能否给每个人分配一项他所胜任的工作？若能，如何求出这种安排？

§3.5 二部图中最大匹配的算法

1.背景与问题

图论描述： 对于一个二部图$G=(X, Y)$：$X=\{x_1, x_2,\ldots,x_n\}$，$Y=\{y_1, y_2, , y_n\}$，当且仅当 x_i胜任工作 y_j 时，$x_i y_j \in E(G)$。问：G 中是否有完美匹配？若有，如何求之？

2. 理论基础

Berge 定理： 图G 的匹配M 是最大匹配的充要条件是G 中不存在M 增广路。

Hall 定理： 设G 是具有二划分(X,Y)的二部图，则G 有饱和X 的匹配当且仅当对$\forall S \subseteq X$，$|N(S)| \geq |S|$，其中$N(S)$表示S 的所有邻点之集。

3.匈牙利算法

匈牙利算法由匈牙利数学家 Egerváry 首先提出，后来由Edmonds(1965)基于Berge 定理和Hall 定理进行了改进[1]。这种算法既能判定一个二部图中完美匹配是否存在，又能在存在时求出一个完美匹配。

[1] J. Edmonds, Path, tree and flowers, Canad. J. Math. 17(1965)449-467.

算法思想： 从图G 的任何匹配M 开始。

若M 饱和X，则M 是G 的完美匹配。

若 M 不饱和X，在X 中选择一个M 不饱和点x。若存在以x为起点的M增广路P，则由Berge 定理知M 不是最大匹配，且$M' = M \oplus E(P)$是比M 更大的匹配。用M'替代M 重复上述过程；

算法思想:

若 G 中不存在以 x 为起点的 M 增广路，则可找到与 x 由 M 交错路相连的顶点集合A，而$S = A \cap X$ 满足$|N(S)| < |S|$。由Hall 定理，G不存在完美匹配。

算法步骤：

step1. 任取图G 的一个匹配M。

step2. 若M 饱和X，则停止， M 是G 的完美匹配。

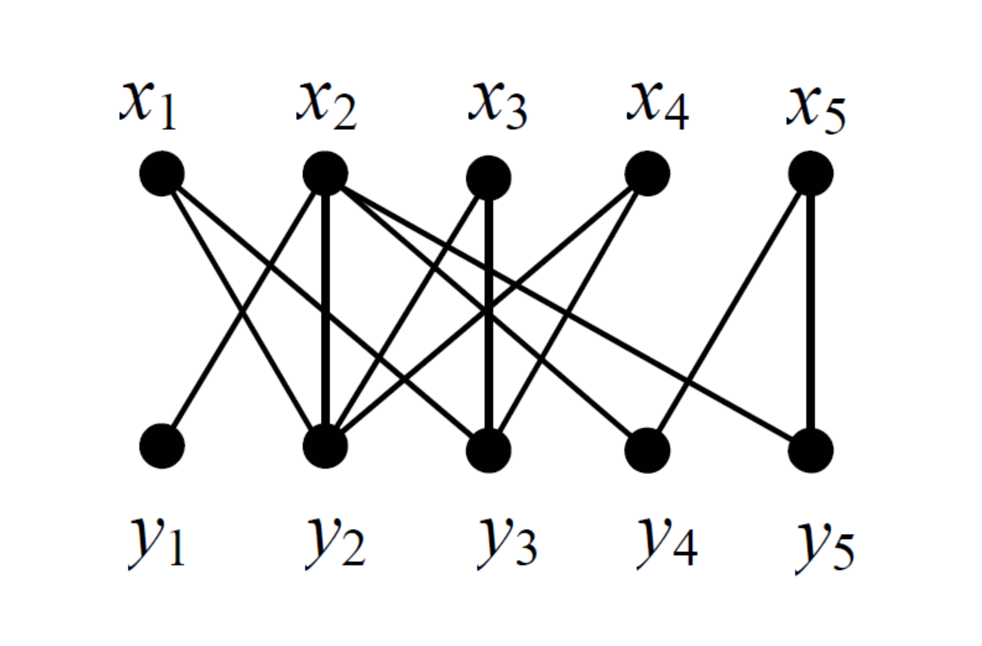
否则， 取 X中一个M 不饱和点x，记S := {x}, T :=∅。

step3. 若$N(S)=T$，则停止，G无完美匹配。(因

$|N(S)|=|T|=|S|-1<|S|$)。否则取 $y \in N(S) \setminus T$ 。

step4. 若y 是M 饱和的，设 $yz \in M$ ，令$S := S \cup \{z\}$, $T := T \cup \{y\}$，转step3。（此时仍保持$|T|=|S|-1$)。否则，取M 增广路$P(x, y)$，令$M:=M \oplus E(P)$，转step2.

例. 判断如下二部图是否存在完美匹配。若存在，求其出一个完美匹配；若不存在，给出满足$N(S) = T$的集合S和T。



匈牙利算法的计算复杂性：

设$n=|X|=|Y|$。算法每找到一条增广路更新一次匹配，匹配边增加一条，故最多需执行n次增广路的循环。

算法每找一条增广路需反复执行第 2 步和第3 步“生长”交错路，这实际上是反复扩充集合S和T的过程。算法每循环一次第2步和第3步，S和T的元素各增加1个。由于$|X|=|Y|=n$，故这种扩张不会超过n次。

而算法每执行一次第二步和第三步，除了需要做两次赋值（$S:=S\cup\{z\}$，$T:=T\cup\{y\}$）和一次判断（y是否M 饱和的）外，主要计算量在于判断$N(S)=T$。由于总有$N(S)\supseteq T$，故可转为判断是否$|N(S)|=|T|$。$|T|$的计数可通过每次循环加1 来实现，$|N(S)|$的计数可用上一循环的$|N(S)|$值加上本次循环新进入S集合的点z在$Y-N(S)$中的邻点数计算，这需要不超过$|Y|=n$次判断。因此算法总的计算复杂度约为$n\cdot n\cdot(n+6)=O(n^3)$。

§3.6 拓展知识

1. 求二部图最大匹配目前已知的最好算法是Hopcroft 和Karp 提出的一个O(n^2.5)算法。读者可参看文献[2]或[3]。

2. 求一般图的最大匹配也有多项式时间算法。第一个多项式时间算法是由Edmonds 给出的一个O(n^4)阶算法。关于该算法读者可参看文献[4]，也可在文献[2]或[5]~[7]中找到对该算法的描述。Ahuja、Magnanti 和Orlin 对这一算法进行了处理[8]，将其复杂度降低为O(n^3)。Even和Kariv提出的一个算法[9]将时间复杂度改进到O(n^2.5)。目前已知的最快算法是Micali和Vazirani 提出的O(mn^0.5)阶算法[10,11]，该算法对稀疏图运算时，比O(n^2.5)算法快。

[2] D.B. West, Introduction to Graph Theory (second edition), Prentice Hall, 2001。（中译本：李建中、骆吉周译，图论导引，机械工业出版社，2006）。
[3] J. Hopcroft, and R.M. Karp, An $O(n2.5)$ algorithm for maximum matching in bipartite graphs,SIAM J. Computing, 2(1973),225-231.
[4] J. Edmonds, Paths, trees, and flowers, Canad. J. Math., 17(1965), 449-467.
[5] C.H. Papadimitriou & K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity,Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.
[6] 田丰，马仲蕃，图与网络流理论，科学出版社，北京，1987。
[7] 蒋长浩，图论与网络流，中国林业出版社，2001。
[8] R.K. Ahuja, T.L. Magnanti, J. B. Orlin, Network Flows : Theory, Algorithms, and Applications,Prentice Hall, 1993.(影印版：网络流：理论算法与应用，机械工业出版社，2005 年)。
[9] S. Even, and O. Kariv, An $O(n2.5)$ algorithm for maximum matching in general graphs, in Proc.16th IEEE Symp. Found. Comp. Sci., 1975, 100-112.
[10] S. Micali, and V.V. Vazirani, An $O(|V|\cdot|E|)$ algorithm for finding maximum matching in general graphs. in Proc. 21th IEEE Symp. Found. Comp. Sci., ACM (1980), 17-27.
[11] V.V. Vazirani, A theory of alternating paths and blossoms for proving correctness of the $O(|V|\cdot|E|)$ general graph matching algorithm, Combinatorica, 14(1994),71-91.

§3.7 赋权二部图最大权匹配

1.背景与问题

最优指派问题（assignment problem）：欲安排n 个人员x_1, $x_2,\ldots,x_n$ 从事n 项工作 y_1, $y_2,\ldots$, y_n。已知每个人能胜任其中一项或几项工作，各人做不同工作的效率不同。求一种工作安排使得总的工作效率达到最大。

§3.7 赋权二部图最大权匹配

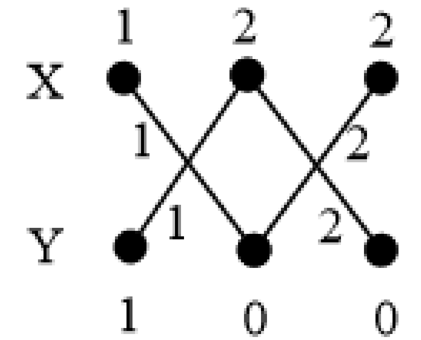
1.背景与问题

图论描述： 给定赋权完全二部图 $K_{n,n} = (X, Y)$：$X = \{x_1, x_2,\dots,x_n\}$，$Y = \{y_1, y_2, \dots, y_n\}$，边$x_iy_j$有权$w_{ij}$（权可以为0，这表示 x_i不胜任工作y_j）。求$K_{n,n}$ 的一个具有最大权的完美匹配。

顶点标号和**可行顶点标号**（feasible vertex labeling）

图 G 的顶点标号是从顶点集到正整数集的一个映射。对于赋权完全图$(K_{n,n}, w)$，若对每条边$e = xy$，均有$l(x) + l(y) \geq w(xy)$，则称这个标号为$(K_{n,n}, w)$的一个可行顶点标号。

下图显示了一个赋权二部图以及它的一个可行定点标号。



一个二部图及其可行顶点标号

注：可行顶点标号总是存在的。一种平凡的可行标号是：

$$l(v) = \begin{cases} \max\limits_{y \in Y} w(vy), v \in X \\ 0, v \in Y \end{cases}$$

相等子图

记$G=K_{n,n}$，设l是(G,w)的一个可行的顶点标号。令

$$E_l=\{xy\in E(G)|l(x)+l(y)=w(xy)\}$$

G 中以E_l为边集的**生成子图**称为 G 的 ***l* 相等子图**，记为G_l。注意G_l的顶点集与 G 的顶点集相同。

下面的例子显示了赋权图$G=K_{4,4}$的平凡顶点标号，以及在这种标号下的相等子图。

$$w=\begin{array}{cc} & \begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \end{array} \\ \begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} & \begin{pmatrix} 3 & 5 & 5 & 4 \\ 2 & 2 & 0 & 2 \\ 2 & 4 & 4 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

图 G 中各边的权

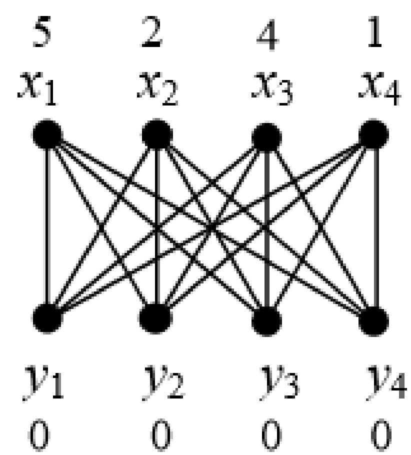
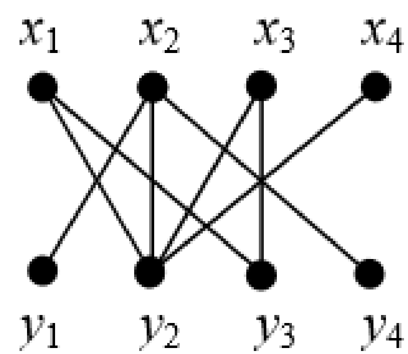


图 G 的平凡标号



对应的相等子图

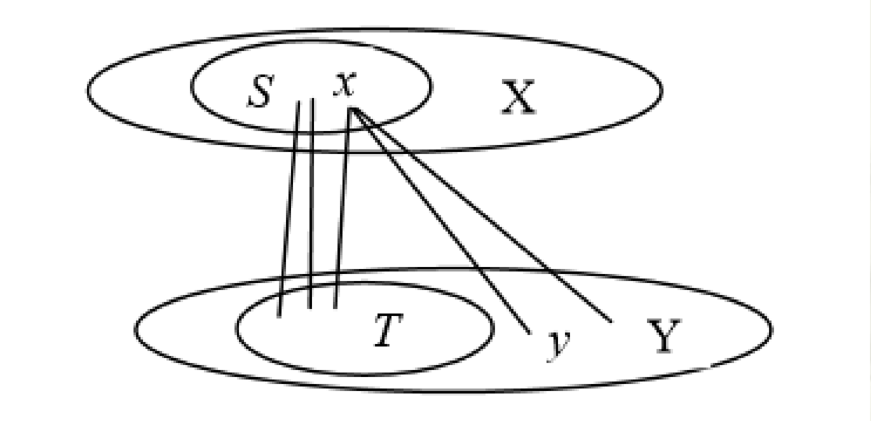
定理 3.7.1 设 l 是图 G 的一个可行顶点标号。若相等子图 G_l 有完美匹配 M^*，则 M^* 是 G 的最大权完美匹配。

算法思想： 首先给出$K_{n,n}$的任意一个可行顶点标号（如平凡标号），然后决定相等子图G_l，在G_l中执行匈牙利算法。若在G_l中找到完美匹配，则由定理 3.7.1，它就是 G 的大权匹配。否则，匈牙利算法终止于$S\subset X$，$T\subset Y$，且$N_{G_l}(S)=T$。

令$\alpha_l=\min\{l(x)+l(y)-w(xy)|x\epsilon S, y\epsilon Y\backslash T\}$， (*)

对每个顶点，修改标号如下

$$l'(u)=\begin{cases} l(u)-\alpha_1 & u\epsilon S \\ l(u)+\alpha_1 & u\epsilon T \\ l(u) & \text{其它} \end{cases} \qquad (**)$$



可以检验l'仍是一个可行标号。用l'替代 l 。$G_{l'}$的边数比G_l的多。继续上述过程，直到获得 一个相等子图含有完全美匹配为止。

Kuhn-Munkres 算法：求赋权完全二部图中的最大权完美匹配。

Step 1. 从任一可行的顶点标号 l 开始，决定G_l.
Step 2. 在G_l中执行匈牙利算法，如果求得完美匹配 M，则 M 即为 G 的最大权匹配，算法停止；否则，匈牙利算法必终止于某个$S \subset X$和$T \subset Y$使得$N_{G_l}(S) = T$，此时转下步。
Step 3. 按(*)式计算α_l，按(**)计算新的可行顶点标号l', 以l'替代 l ， 以$G_{l'}$替代G_l，转第２步。

Kuhn-Munkres 算法的计算复杂度为O(n^3)。