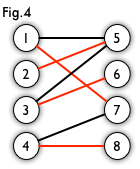
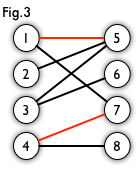
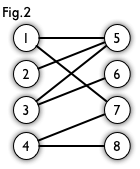
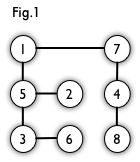
这篇文章讲无权二分图（unweighted bipartite graph）的 大匹配（maximum matching）和完美匹配（perfect matching），以及用于求解匹配的匈牙利算法（Hungarian Algorithm）；不讲带权二分图的 佳匹配。

二分图：简单来说，如果图中点可以被分为两组，并且使得所有边都跨越组的边界，则这就是一个二分图。准确地说：把一个图的顶点划分为两个不相交集 U U 和V V ，使得每一条边都分别连接U U、V V中的顶点。如果存在这样的划分，则此图为一个二分图。二分图的一个等价定义是：不含有「含奇数条边的环」的图。图 1 是一个二分图。为了清晰，我们以后都把它画成图 2 的形式。

匹配：在图论中，一个「匹配」（matching）是一个边的集合，其中任意两条边都没有公共顶点。例如，图 3、图 4 中红色的边就是图 2 的匹配。



我们定义匹配点、匹配边、未匹配点、非匹配边，它们的含义非常显然。例如图 3 中 1、

4、5、7 为匹配点，其他顶点为未匹配点；1-5、4-7为匹配边，其他边为非匹配边。

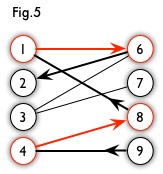
大匹配：一个图所有匹配中，所含匹配边数 多的匹配，称为这个图的 大匹配。图 4 是一个 大匹配，它包含 4 条匹配边。

完美匹配：如果一个图的某个匹配中，所有的顶点都是匹配点，那么它就是一个完美匹配。图 4 是一个完美匹配。显然，完美匹配一定是 大匹配（完美匹配的任何一个点都已经匹配，添加一条新的匹配边一定会与已有的匹配边冲突）。但并非每个图都存在完美匹配。

举例来说：如下图所示，如果在某一对男孩和女孩之间存在相连的边，就意味着他们彼此喜欢。是否可能让所有男孩和女孩两两配对，使得每对儿都互相喜欢呢？图论中，这就是完美匹配问题。如果换一个说法： 多有多少互相喜欢的男孩/女孩可以配对儿？这就是大匹配问题。

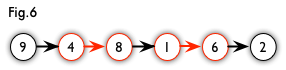


基本概念讲完了。求解 大匹配问题的一个算法是匈牙利算法，下面讲的概念都为这个算法服务。



交替路：从一个未匹配点出发，依次经过非匹配边、匹配边、非匹配边…形成的路径叫交替路。

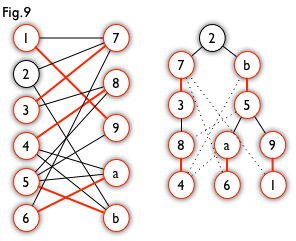
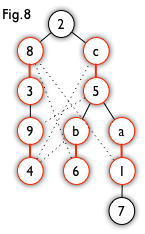
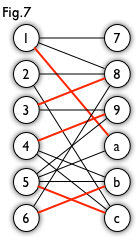
增广路：从一个未匹配点出发，走交替路，如果途径另一个未匹配点（出发的点不算），则这条交替路称为增广路（agumenting path）。例如，图 5 中的一条增广路如图 6 所示（图中的匹配点均用红色标出）：



增广路有一个重要特点：非匹配边比匹配边多一条。因此，研究增广路的意义是改进匹配。只要把增广路中的匹配边和非匹配边的身份交换即可。由于中间的匹配节点不存在其他相连的匹配边，所以这样做不会破坏匹配的性质。交换后，图中的匹配边数目比原来多了 1 条。

我们可以通过不停地找增广路来增加匹配中的匹配边和匹配点。找不到增广路时，达到 大匹配（这是增广路定理）。匈牙利算法正是这么做的。在给出匈牙利算法 DFS 和 BFS 版本的代码之前，先讲一下匈牙利树。

匈牙利树一般由 BFS 构造（类似于 BFS 树）。从一个未匹配点出发运行 BFS（唯一的限制是，必须走交替路），直到不能再扩展为止。例如，由图 7，可以得到如图 8 的一棵 BFS 树：



这棵树存在一个叶子节点为非匹配点（7 号），但是匈牙利树要求所有叶子节点均为匹配点，因此这不是一棵匈牙利树。如果原图中根本不含 7 号节点，那么从 2 号节点出发就会得到一棵匈牙利树。这种情况如图 9 所示（顺便说一句，图 8 中根节点 2 到非匹配叶子节点 7 显然是一条增广路，沿这条增广路扩充后将得到一个完美匹配）。下面给出匈牙利算法的 DFS 和 BFS 版本的代码：

匈牙利算法的要点如下

1. 从左边第 1 个顶点开始，挑选未匹配点进行搜索，寻找增广路。
   1. 如果经过一个未匹配点，说明寻找成功。更新路径信息，匹配边数 +1，停止搜索。
   2. 如果一直没有找到增广路，则不再从这个点开始搜索。事实上，此时搜索后会形成一棵匈牙利树。我们可以永久性地把它从图中删去，而不影响结果。
2. 由于找到增广路之后需要沿着路径更新匹配，所以我们需要一个结构来记录路径上的点。

DFS 版本通过函数调用隐式地使用一个栈，而 BFS 版本使用 prev 数组。

性能比较

两个版本的时间复杂度均为O(V ⋅E) O(V⋅E)。DFS 的优点是思路清晰、代码量少，但是性能不如 BFS。我测试了两种算法的性能。对于稀疏图，BFS 版本明显快于 DFS 版本；而对于稠密图两者则不相上下。在完全随机数据 9000 个顶点 4,0000 条边时前者领先后者大约 97.6%，9000 个顶点 100,0000 条边时前者领先后者 8.6%, 而达到 500,0000 条边时 BFS 仅领先 0.85%。

补充定义和定理：

大匹配数： 大匹配的匹配边的数目

小点覆盖数：选取 少的点，使任意一条边至少有一个端点被选择

大独立数：选取 多的点，使任意所选两点均不相连

小路径覆盖数：对于一个 DAG（有向无环图），选取 少条路径，使得每个顶点属于且仅属于一条路径。路径长可以为 0（即单个点）。

定理1： 大匹配数 = 小点覆盖数（这是 Konig 定理）

定理2： 大匹配数 = 大独立数

定理3： 小路径覆盖数 = 顶点数 - 大匹配数