# **二分图总结【最大匹配、最小点覆盖、最少路径覆盖和最大独立集】**

### **二分图的最大匹配**

**给定一个[二分图](https://baike.baidu.com/item/%E4%BA%8C%E5%88%86%E5%9B%BE/9089095" \t "https://baike.baidu.com/item/%E4%BA%8C%E5%88%86%E5%9B%BE%E5%8C%B9%E9%85%8D/_blank)G，在G的一个子图M中，M的边集{E}中的任意两条边都不依附于同一个顶点，则称M是一个[匹配](https://baike.baidu.com/item/%E5%8C%B9%E9%85%8D/6397551" \t "https://baike.baidu.com/item/%E4%BA%8C%E5%88%86%E5%9B%BE%E5%8C%B9%E9%85%8D/_blank)。**

**匈牙利算法**

### **(2)二分图的最小点覆盖**

**二分图的最小点覆盖=二分图的最大匹配**

**求最小点覆盖：从右边所有没有匹配过的点出发，按照增广路的“交替出现”的要求DFS。最终右边没有访问过的点和左边访问过的点组成最小点覆盖。**

### **二分图的最少边覆盖**

**边覆盖是指E的一个子集X，使得对于V中每个顶点v，都能在X中找到某条边x满足v是x的一个端点。而最小边覆盖是指所有满足上述条件的边集中基数最小的一个。**

**二分图的最少边覆盖=点数-二分图的最大匹配**

**证明：**

**先贪心选一组最大匹配的边放进集合，对于剩下的没有匹配的点，随便选一条与之关联的边放进集合，那么得到的集合就是最小边覆盖。**

**所以有：最小边覆盖=最大匹配+点数-2\*最大匹配=点数-最大匹配**

### **二分图的最大独立集**

**二分图的最大独立集是：一个最大的点的集合，该集合内的任意两点没有边相连。**

**二分图的最大独立集=点数-二分图的最大匹配**

**证明：**

**我们可以这样想，先把所有的点放进集合，然后删去最少的点和与之相关联的边，使得全部边都被删完，这就是最小点覆盖。所以有：最大独立集=点数-最小点覆盖**

### **有向无环图的最少不相交路径覆盖**

**在一个有向图中，找出最少的路径，使得这些路径经过了所有的点。**

**最小路径覆盖分为**最小不相交路径覆盖**和**最小可相交路径覆盖**。**

****最小不相交路径覆盖**：每一条路径经过的顶点各不相同**

**我们把原图中的点VV拆成两个点VxVx和VyVy，对于原图中的边A−>BA−>B，我们在新图中连Ax−>ByAx−>By。那么最少不相交路径覆盖=原图的点数-新图的最大匹配**

**证明：**

**一开始每个点都独立为一条路径，在二分图中连边就是将路径合并，每连一条边路径数就减一。因为路径不能相交，所以不能有公共点，这恰好就是匹配的定义。所以有：最少不相交路径覆盖=原图的点数-新图的最大匹配**

### **有向无环图的最少可相交路径覆盖**

**每一条路径经过的顶点可以相同**

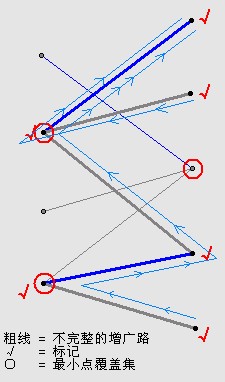
**先用floyd求出原图的传递闭包， 如果a到b有路， 那么就加边a->b。 然后就转化成了最少不相交路径覆盖问题。**

### **(7)有向无环图中最少不相交路径覆盖和最大独立集的相互转化**

**Dilworth定理：有向无环图的最大独立集=有向无环图最少不相交路径覆盖**

**注意要去重，因为要保证是有向无环图。**

## **IMG_256二分图——最大匹配、最小点覆盖**

** König定理是一个二分图中很重要的定理，它的意思是，一个二分图中的最大匹配数等于这个图中的最小点覆盖数。如果你还不知道什么是最小点覆盖，我也在这里说一下：假如选了一个点就相当于覆盖了以它为端点的所有边，你需要选择最少的点来覆盖所有的边。比如，下面这个图中的最大匹配和最小点覆盖已分别用蓝色和红色标注。它们**

**假如我们已经通过匈牙利算法求出了最大匹配（假设它等于M），下面给出的方法可以告诉我们，选哪M个点可以覆盖所有的边。  
    匈牙利算法需要我们从右边的某个没有匹配的点，走出一条使得“一条没被匹配、一条已经匹配过，再下一条又没匹配这样交替地出现”的路（交错轨，增广路）。但是，现在我们已经找到了最大匹配，已经不存在这样的路了。换句话说，我们能寻找到很多可能的增广路，但最后都以找不到“终点是还没有匹配过的点”而失败。我们给所有这样的点打上记号：从右边的所有没有匹配过的点出发，按照增广路的“交替出现”的要求可以走到的所有点（最后走出的路径是很多条不完整的增广路）。那么这些点组成了最小覆盖点集：右边所有没有打上记号的点，加上左边已经有记号的点。看图，右图中展示了两条这样的路径，标记了一共6个点（用 “√”表示）。那么，用红色圈起来的三个点就是我们的最小覆盖点集。  
    首先，为什么这样得到的点集点的个数恰好有M个呢？答案很简单，因为每个点都是某个匹配边的其中一个端点。如果右边的哪个点是没有匹配过的，那么它早就当成起点被标记了；如果左边的哪个点是没有匹配过的，那就走不到它那里去（否则就找到了一条完整的增广路）。而一个匹配边又不可能左端点是标记了的，同时右端点是没标记的（不然的话右边的点就可以经过这条边到达了）。因此，最后我们圈起来的点与匹配边一一对应。  
    其次，为什么这样得到的点集可以覆盖所有的边呢？答案同样简单。不可能存在某一条边，它的左端点是没有标记的，而右端点是有标记的。原因如下：如果这条边不属于我们的匹配边，那么左端点就可以通过这条边到达（从而得到标记）；如果这条边属于我们的匹配边，那么右端点不可能是一条路径的起点，于是它的标记只能是从这条边的左端点过来的（想想匹配的定义），左端点就应该有标记。  
    最后，为什么这是最小的点覆盖集呢？这当然是最小的，不可能有比M还小的点覆盖集了，因为要覆盖这M条匹配边至少就需要M个点（再次回到匹配的定义）。**