浅谈图论模型的建立与应用

广东省中山市第一中学

黄源河

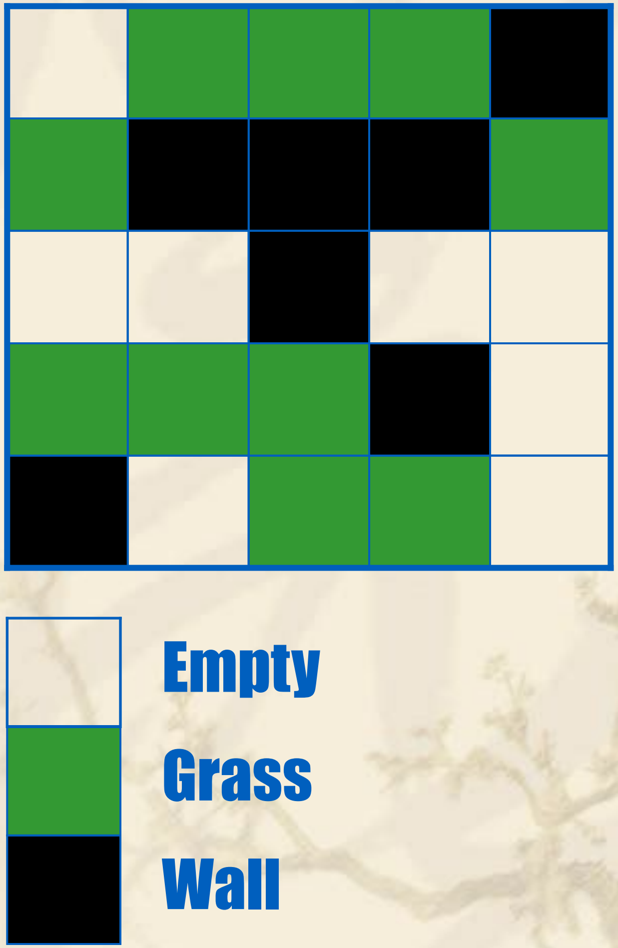
引言

- 图论是数学的一个有趣的分支。
- 图论的建模，就是要抓住问题的本质，把问题抽象为点、边、权的关系。
- 许多看似无从入手的问题，通过图论建模，往往能转化为我们熟悉的经典问题。

例题1 Place the Robots (ZOJ)

问题描述

有一个N*M(N,M<=50)的棋盘，棋盘的每一格是三种类型之一：空地、草地、墙。机器人只能放在空地上。在同一行或同一列的两个机器人，若它们之间没有墙，则它们可以互相攻击。问给定的棋盘，最多可以放置多少个机器人，使它们不能互相攻击。

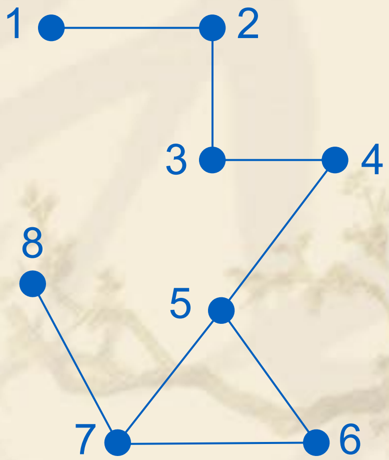


例题1 Place the Robots (ZOJ)

模型一

在问题的原型中，草地，墙这些信息不是我们所关心的，我们关心的只是空地和空地之间的联系。因此，我们很自然想到了下面这种简单的模型：

以空地为顶点，有冲突的空地间连边，我们可以得到右边的这个图：



于是，问题转化为求图的最大独立集问题。

例题1 Place the Robots (ZOJ)

模型一

在问题的原型中，草地，墙这些信息不是我们所关心的，我们关心的只是空地和空地之间的联系。因此，我们很自然想到了下面这种简单的模型：

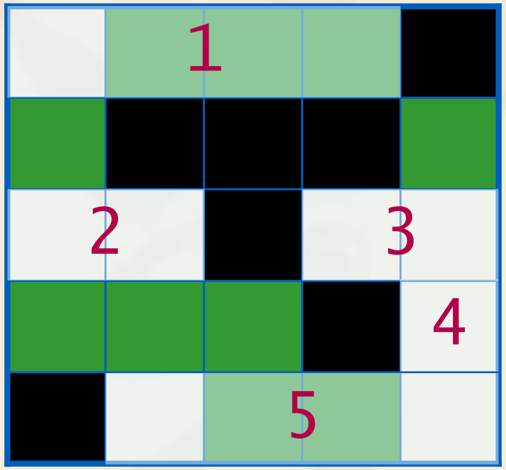
以空地为顶点，有冲突的空地间连边，我们可以得到右边的这个图：

这是NP问题！

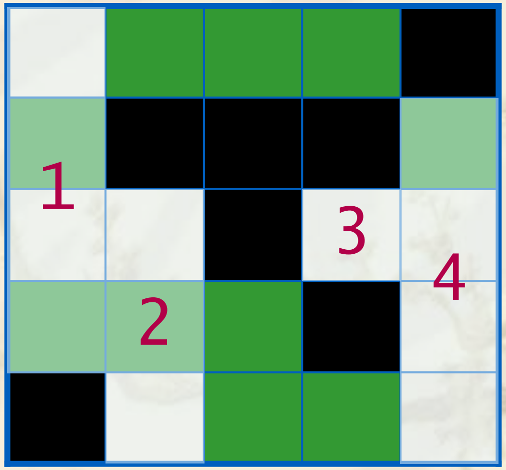
例题1 Place the Robots (ZOJ)

模型二

我们将每一行，每一列被墙隔开，且包含空地的连续区域称作“块”。显然，在一个块之中，最多只能放一个机器人。我们把这些块编上号。



同样，把竖直方向的块也编上号。

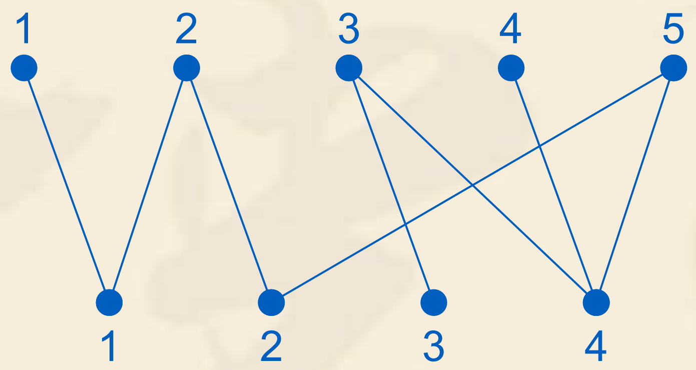


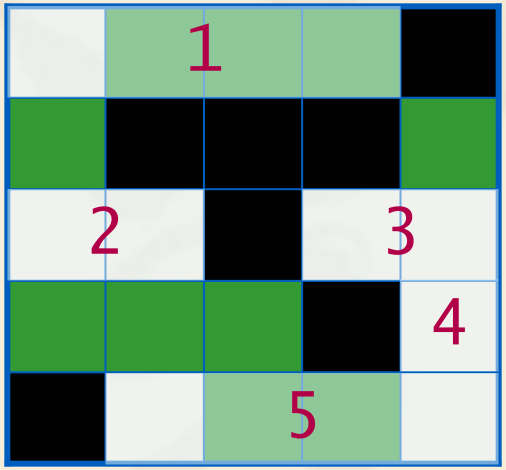
例题1 Place the Robots (ZOJ)

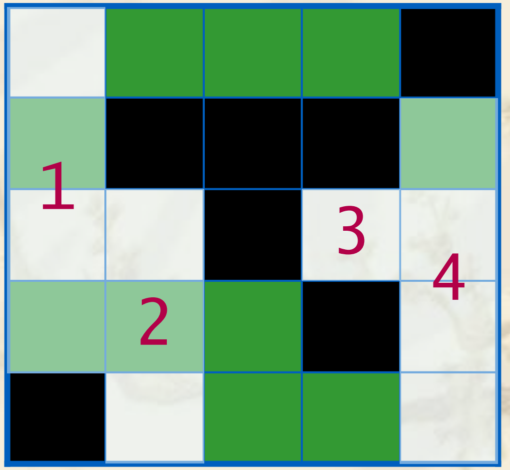
模型二

把每个横向块看作X部的点，竖向块看作Y部的点，若两个块有公共的空地，则在它们之间连边。

于是，问题转化成这样的一个二部图：

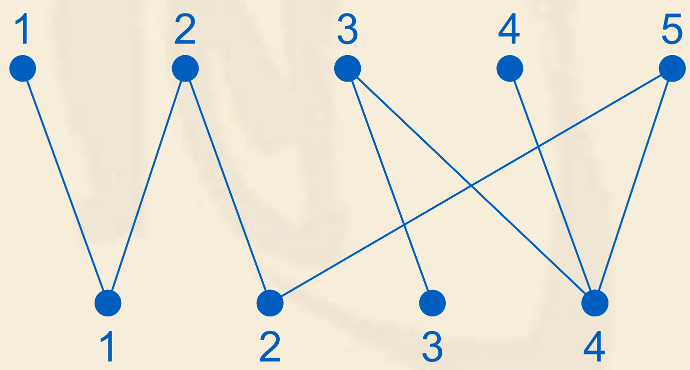


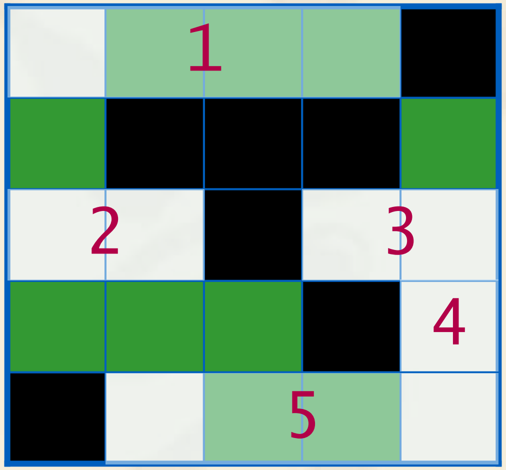


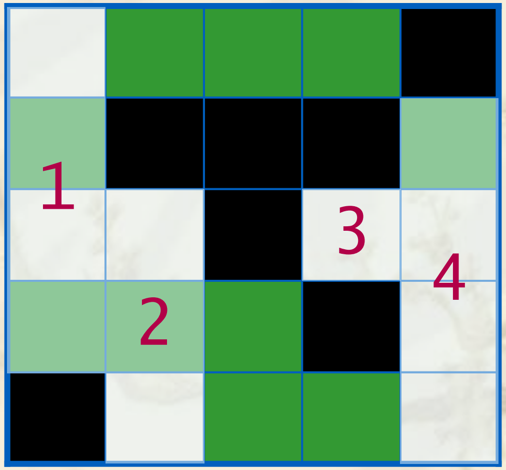


例题1 Place the Robots (ZOJ)

模型二







由于每条边表示一个空地，有冲突的空地之间必有公共顶点，所以问题转化为二部图的最大匹配问题。

例题1 Place the Robots (ZOJ)

小结

比较前面的两个模型：模型一过于简单，没有给问题的求解带来任何便利；模型二则充分抓住了问题的内在联系，巧妙地建立了二部图模型。为什么会产生这种截然不同的结果呢？其一是由于对问题分析的角度不同：模型一以空地为点，模型二以空地为边；其二是由于对原型中要素的选取有差异：模型一对要素的选取不充分，模型二则保留了原型中“棋盘”这个重要的性质。由此可见，对要素的选取，是图论建模中至关重要的一步。

例题2 出纳员的雇佣（ACM Tehran 2000）

问题描述

有一家24小时营业的超市，需要雇佣一批出纳员。一天中每个小时需要出纳员的最少数量为$R_0,R_1,R_2,\ldots,R_{23}$。有N个人申请这项工作，每个申请者，从一个特定时刻开始连续工作恰好8个小时，设W_i（i=0...23）表示从时刻i开始工作的申请者的人数（$\sum W_i=N<=1000$）。

你的任务是计算出需要雇佣出纳员的最少数目，满足在每一时刻i，至少有R_i名出纳员在工作。

例题2 出纳员的雇佣（ACM Tehran 2000）

分析

初看本题，很容易使人往贪心、动态规划或网络流这些方面思考。然而，对于本题，这些算法都无能为力。

由于本题的约束条件很多，为了理清思路，我们先把题目中的约束条件用数学语言表达出来。设S[i]表示0～i时刻雇佣出纳员的总数，那么我们可以将题目中的约束条件转化为下面的不等式组：

$0 \le S[i]-S[i-1] \le W_i \quad (0 \le i \le 23)$

$S[i]-S[i-8] \ge R_i \quad (8 \le i \le 23)$

$S[23]+S[i]-S[i+16] \ge R_i \quad (0 \le i \le 7)$

例题2 出纳员的雇佣（ACM Tehran 2000）

分析

$0 \le S[i]-S[i-1] \le W_i \qquad (0 \le i \le 23)$

$S[i]-S[i-8] \ge R_i \qquad (8 \le i \le 23)$

$S[23]+S[i]-S[i+16] \ge R_i \qquad (0 \le i \le 7)$

怎么办

这样的不等式组，不禁使我们想到了差分约束系统。

对于每个不等式 $S[i]-S[j] \le K$，从顶点 j 向顶点 i 引一条权值为K的有向边。我们要求S[23]的最小值，就是要求顶点0到顶点23的最短路。

注意上面第三条不等式：它包含三个未知数，无法在图中表示为边的关系。

例题2 出纳员的雇佣（ACM Tehran 2000）

分析

$0 \le S[i]-S[i-1] \le W_i \quad (0 \le i \le 23)$

$S[i]-S[i-8] \ge R_i \quad (8 \le i \le 23)$

$S[23]+S[i]-S[i+16] \ge R_i \quad (0 \le i \le 7)$

退一步考虑：如果S[23]已经确定了，那么上面的不等式组可以完全转化为一个有向图，顶点0到顶点 i 的最短路，就是S[i]的解。而当图中存在负权回路时，不等式组无解。

至于S[23]，我们可以用二分法枚举，逐步缩小范围，用迭代法判断是否存在负权回路（判定可行性），最终求得S[23]的最小值。时间复杂度为$O(24^3*\log_2 N)$。

例题2 出纳员的雇佣（ACM Tehran 2000）

小结

本题用到了差分约束系统的理论，在竞赛中，这样的系统并不多见，但是却可以巧妙的解决一些难题。这类题目的模型都不明显，需要一定的思考和转化。做这类题目，关键是要把题目中的约束条件表示为不等式，再把不等式转化为图的最短路或最长路模型。

例题3 贪婪之岛（ZOJ）

问题描述

有N（N≤100000）张卡片，每张卡片有三种能力，每种能力的能力值分别为Ai，Bi，Ci。每张卡片可以使用其中一种能力，且每张卡片只能使用一次。现在需要A张卡片使用第一种能力，B张卡片使用第二种能力，C张卡片使用第三种能力（A+B+C≤100）。请计算使用哪些卡片，以及使用卡片的哪项能力，可以使相应的能力值之和最大。

例题3 贪婪之岛（ZOJ）

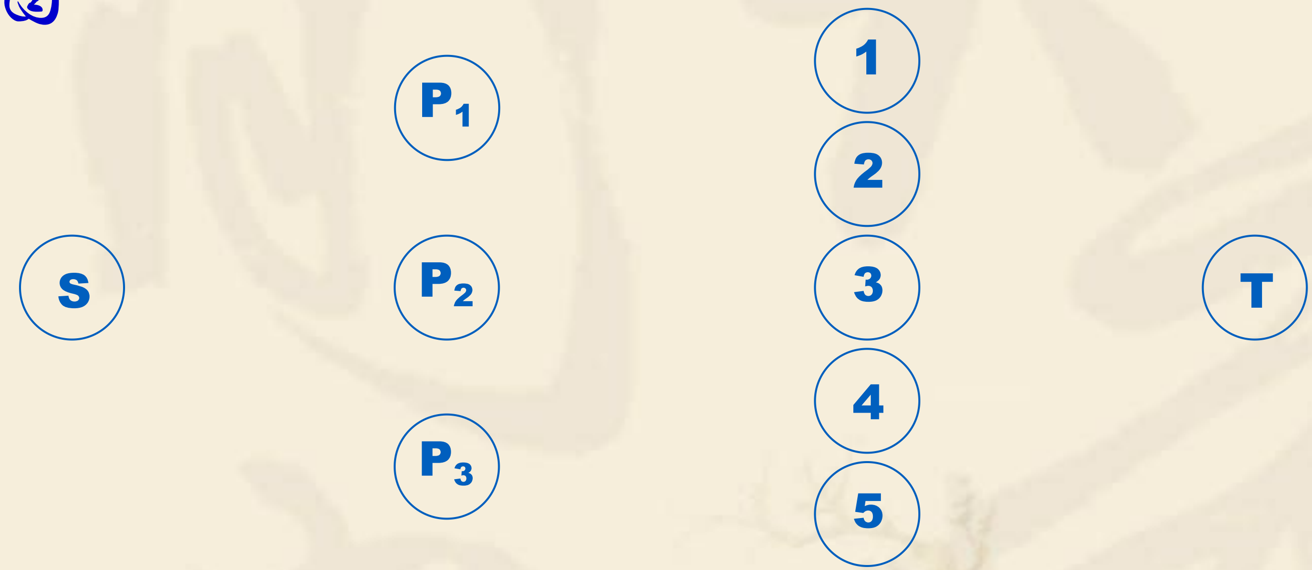
分析

最优化问题的解法有很多种，比如动态规划，网络流等，而本题就是一个比较明显的网络流模型。

网络流模型中，权的类型众多，有流量，容量，还可以有费用。在本题中，容量可以作为选取的约束，确保解的合法性；费用则表示选取的价值，确保解的最优性。因此，更确切地说，本题是一个最大费用最大流模型。

例题3 贪婪之岛（ZOJ）

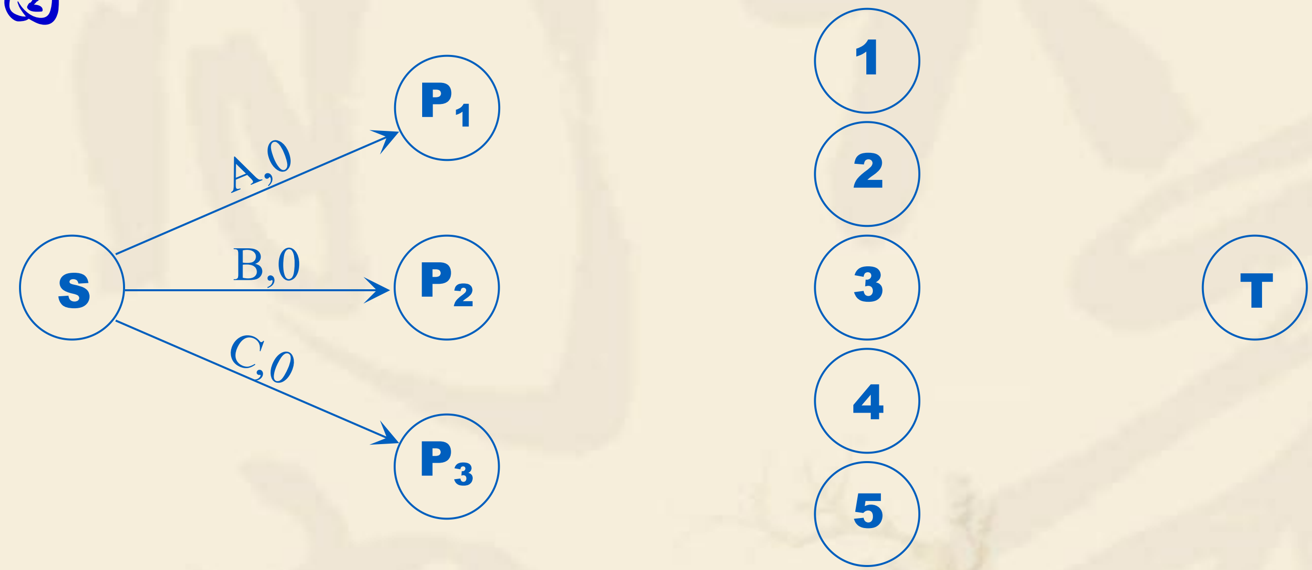
构图



每张卡片 i 用顶点 i 表示，另外加三个顶点P_1，P_2，P_3，表示三种能力，还有源点S，汇点T。

例题3 贪婪之岛（ZOJ）

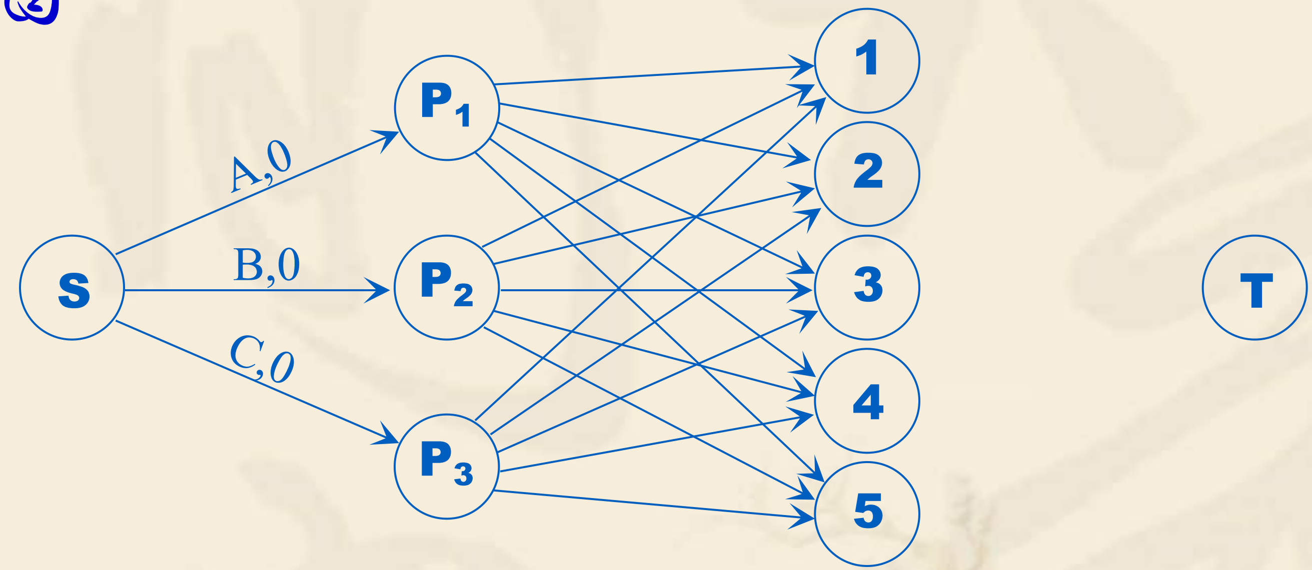
构图



从源点分别向P_1，P_2，P_3引一条弧，容量分别为A，B，C，费用为0。

例题3 贪婪之岛（ZOJ）

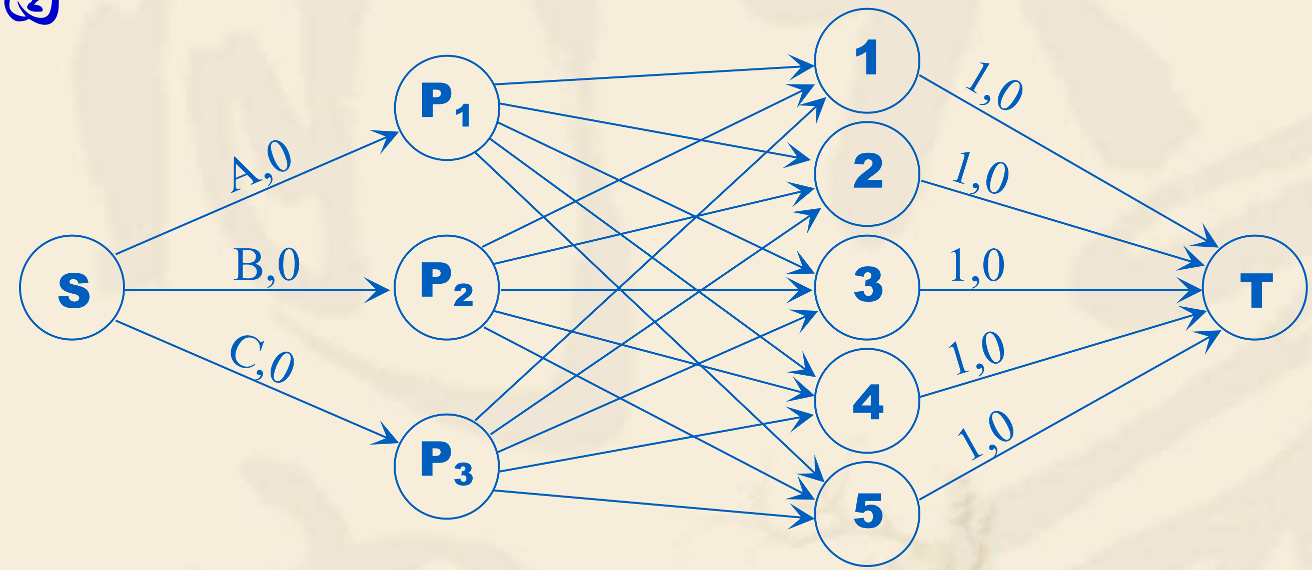
构图



从P1，P2，P3向顶点 i (1≤i≤N) 分别引一条弧，容量为1，费用分别为Ai，Bi，Ci。

例题3 贪婪之岛（ZOJ）

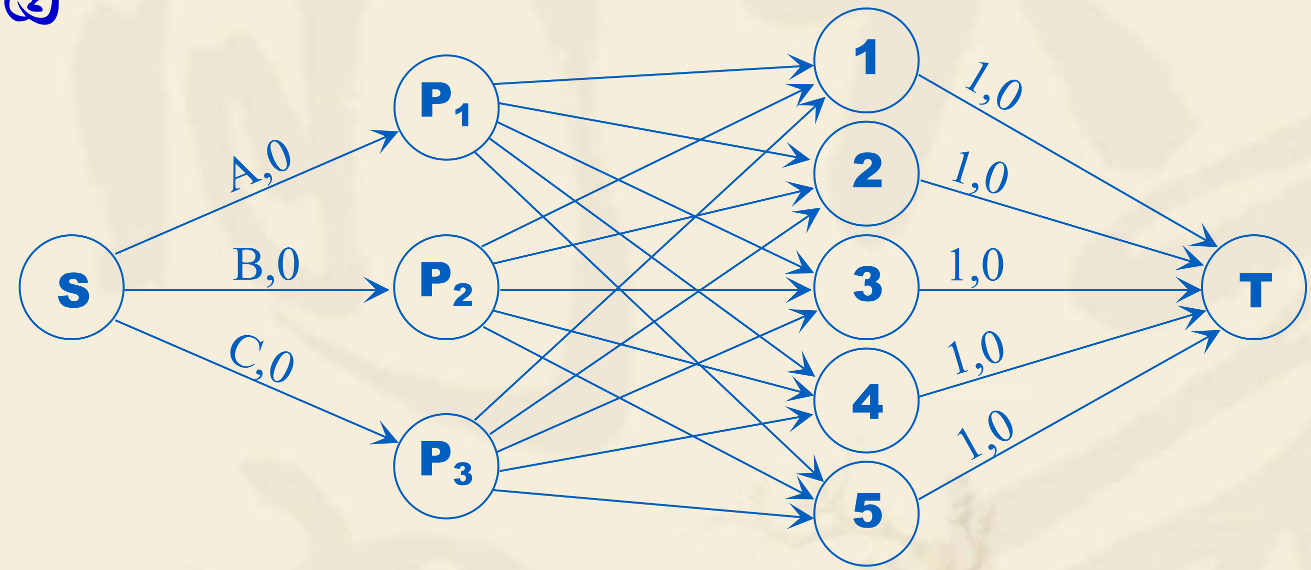
构图



从顶点 i (1≤i≤N) 向汇点引一条弧，容量为1，费用为0。

例题3 贪婪之岛（ZOJ）

构图



构图之后，求出从S到T的最大费用最大流，再检查流出P_1，P_2，P_3的弧，并输出最优方案。

时间复杂度：$\cancel{O(N^3)}$ N太大了，需要进一步优化！

例题3 贪婪之岛（ZOJ）

优化

本题的卡片总数有十万之多，而最终要选取的卡片数不超过100张。如果在构图之前，把没有用的卡片先删掉，必将大大提高效率。

什么样的卡片是没有用的呢?

先考虑第一种能力的选取：如果把全部卡片按第一种能力值从大到小排序，显然我们应该尽量从前面选A张出来，由于每张卡片只能使用一次，所以有可能会和其他的两种能力发生冲突，而冲突的卡片数最多是B+C张，所以实际上对我们有用的卡片只是前面的A+B+C张。

例题3 贪婪之岛（ZOJ）

优化

同理，对于第二种和第三种能力的选取，也只需保留其能力值最大的前A+B+C张卡片。这一步可以在线性时间内解决。

这是一个既简单又有效的方法，经过这一步处理，保留下来的卡片数不会超过3(A+B+C)张，顶点数大大减少，求解最大费用最大流的时间复杂度降为$O((A+B+C)^3)$。

至此，算法已经优化到了一个可以接受的地步，时间复杂度仅为$O(N+(A+B+C)^3)$。

例题3 贪婪之岛（ZOJ）

优化

如果还要进一步提高效率，可以用更有效的算法删掉多余的顶点。不过这样做意义不大，而且也不是本文讨论的要点。

另外，本题还可以转化为二部图模型，用最佳匹配算法求解。这一步留给读者自己思考。

例题3 贪婪之岛（ZOJ）

小结

本题建立的是网络流模型。这类模型的算法系数大，编程复杂度也大，在竞赛中往往作为走投无路时的“候补算法”。但是，网络流模型的适用性广，一些较复杂，或者约束较多的问题，网络流模型可以很好地解决，而基于网络流模型的问题又比较明显，这使得网络流模型有着广泛的应用。

结语

问题是千变万化的，如何建立问题的图论模型并没有通用的准则。前面的几个例子都比较简单，在更复杂的问题中，有时我们会感到难以建立适当的模型，这时，我们需要在不改变问题原型本身的性质的前提下，对原型进行抽象，简化，在此基础上建立合适，有效的模型。有时，我们建立了问题的一个模型之后，可能会感到难以求解，这时，我们可能需要对模型进行修改，转化，或者对原型进行更深入的分析，抽取其中较关键的要素，建立一个易于求解的模型。这些都需要我们有丰富的经验，灵活的思维以及良好的创造力。

谢谢！