二分图与网络流模型设计

GOOGLE 李煜东 2019/1/25

Review: 网络流基本定义

- 有向图G(V,E)，每条边(u,v)∈E有一个非负实数的容量c(u,v)
- 源点S和汇点T
- 实际流量f(u,v)，剩余容量c(u,v)-f(u,v)，净流f(u,v)-f(v,u)
- 容量限制：f(u,v)<=c(u,v)
- 斜对称：u到v的净流是v到u的净流的相反数
- 流守恒：除了源点和汇点外，其余各点流入和流出的流量相等
- 所有点和剩余容量大于0的边构成的图被称为残量网络。
- 残量网络中从源点S到汇点T的路径被称为增广路。

Review: 网络流基础算法

- Edmonds-Karp增广路算法
  - 求解最大流的基础算法
  - 求解最小费用最大流/最大费用最大流的常用算法
- Dinic算法 / SAP算法
  - 求解最大流的高效算法
- NOI中一般不需要应用其它算法

Review: 二分图

- 二分图
  - 无奇环无向图（点可以分成左右两部，每一部内没有边）
  - 判定：DFS 0/1染色
- 最大匹配
  - 增广路算法（匈牙利算法）、最大流
- 经典模型
  - 最小点覆盖、最大独立集、最小路径点覆盖
- 带权匹配
  - KM算法、最小/最大费用最大流

REVIEW: 二分图常见问题

- Points
  - 其一是如何选择点、边、部，其二是映射为何种经典模型，其三是建图和算法
- 矩形网格中的各类问题
  - 棋盘覆盖、放置車、放置马、带障碍放置(ZOJ1654)、带障碍覆盖(POJ2226)
- 多重匹配与拆点
  - 直接进行多重匹配：匈牙利算法的两种修改写法 or 直接用网络流
  - 拆点再做

例题

- HEOI2012 朋友圈
- http://www.lydsy.com/JudgeOnline/problem.php?id=2744

例题

- HEOI2012 朋友圈
- http://www.lydsy.com/JudgeOnline/problem.php?id=2744
- 观察原图：A国中友善度奇偶性不同的点之间有边，B国中所有奇数之间有边、所有偶数之间有边、奇偶不同的数之间一部分有边。
- 观察朋友圈的定义：朋友圈是一个团。
- 一般图的最大团问题是NPC的，然而这个图的补图很特殊。

例题

- 观察补图：A国的奇数值点构成完全图，偶数值点构成完全图；
- B国的奇数点和偶数点构成二分图。
- 补图上朋友圈的定义：朋友圈是一个独立集（最大团=补图最大独立集）
- A国最多取两个点，可以枚举这两个点x,y；
- 数据给定了A和B国之间有边的情况，删除与x,y之间无边的B国点；
- 然后在B国剩余的点上求二分图的最大独立集。

二分图的可行边与必须边

- 如果存在一组完备匹配?
  - 对于匹配边(u,v)，v到u连边；非匹配边u到v连边；
  - 求强连通分支，若(u,v)是匹配边或者u,v在同一个分支中——可行边；
  - 若(u,v)是匹配边且u,v不在同一个分支中——必须边。

二分图的可行边与必须边

- 如果不一定存在完备匹配?
  - 先用Dinic求出任意一组最大匹配。建一张新图:
  - 对于匹配边(u,v), v到u连边;非匹配边u到v连边;
  - 对于匹配的左部点u, u到S连边;未匹配的左部点u, S到u连边;
  - 对于匹配的右部点v, T到v连边;未匹配的右部点v, v到T连边。
  - 求强连通分支,若(u,v)是匹配边或者u,v在同一个分支中——可行边;
  - 若(u,v)是匹配边且u,v不在同一个分支中——必须边。

最小割

- 删去之后使网络中源点S到汇点T不存在路径的边的集合称为网络的割。
- 最小割，就是使这个边集中所有边的容量之和最小。
- 最大流最小割定理：任何一个网络的最大流量等于该网络的最小割的容量。
- 如果最小割<最大流，那么根据最大流算法，割去这些边之后，仍然可以找到一条从S到T的增广路。所以最小割不小于最大流。因此如果我们能给出一个等于最大流的割集构造方案，就可以证明最小割=最大流。
- 求出最大流后，从源点开始沿残量网络BFS，标记能够到达的点。连接已标记的点和未标记的点的正向边就是该网络的一个最小割集。

最小割的可行边与必须边

- 最小割的必须边
  - 一定在最小割中的边、扩大容量后能增大最大流的边，Poj3204：
  - ① 满流；② 残余网络中S能到入点、出点能到T。
  - 从S开始DFS、T开始反向DFS，标记到达的点，然后枚举满流边即可。
- 最小割的可行边
  - 被某一种最小割的方案包含的边，AHOI2009：
  - ① 满流；② 删掉之后在残余网络中找不到u到v的路径。
  - 在残余网络中tarjan求SCC，(u,v)两点在同一SCC中说明残余网络中存在u到v路径。

网络流基本技巧

- 点边转化
  - 一个点拆成两个，中间加一条边，把点的各种信息反映在这条边上
  - 一条边截成两半，中间插入一个点，把边的各种信息反映在这个点上
  - 求无向图点/边连通度（Poj1966、Poj1815、Poj2914）
  - K取方格数（POJ3422）
- $+\infty$ 容量边
  - 防割
  - 流量传递（Poj1149 Pigs）

动态加点

- NOI2012 美食节
  http://www.lydsy.com/JudgeOnline/problem.php?id=2879

动态加点

- NOI2012 美食节 http://www.lydsy.com/JudgeOnline/problem.php?id=2879
- 源点向每道菜连边，容量为 p[i]，费用为0。
- 每个厨师拆成n个点，向汇点连边，容量为1，费用为0。
- 第i到菜向第j个厨师拆成的第k个点连边，容量1，费用k*a[i][j]。
- 求最小费用最大流（超时）。
- 使用动态加点法：起初每个厨师只拆成一个点，每次无法增广时，把满流的厨师拆出一个新点。

混合图欧拉回路 - Poj1637

- 一张图中既有无向边，也有有向边，求经过每条边恰好一次的回路。

混合图欧拉回路 - Poj1637

- 一张图中既有无向边，也有有向边，求经过每条边恰好一次的回路。
- 把图中的无向边任意定向。
- 如果有某个点的出入度之差为奇数，则不存在欧拉回路。
- 设K[i] = | inDeg[i] - outDeg[i] | / 2
- 删除有向边，无向边容量设为1，新建源汇。
- 对于入度大于出度的点u，连边(u, t)、容量K[u]。
- 对于入度小于出度的点v，连边(s, v)、容量K[v]。
- 计算最大流，若满流则有解。

混合图欧拉回路 - Poj1637

- 总入度=总出度，即源点连出的边容量和等于汇点连入的边容量和，源点和汇点一定同时满流。
- 每个入度 > 出度的点v（与T相连），都有K[v]条边流出去到T；
- 对于出度 > 入度的点u（与S相连），都有K[u]条边从S流进来。
- 对于入度 = 出度的点（未与S、T相连），流量平衡。
- 将所有流量不为0的边反向，使得出入度相差2*K[i]的点i关联的K[i]条边改变方向，就得到了每个点入度 = 出度的欧拉图。

Poj1895 bring them there

- 题目描述：在公元3141年人类的足迹已经遍布银河系。为了穿越那巨大的距离，人类发明了一种名为超时空轨道的技术。超时空轨道是双向的，连接两个星系，穿越轨道需要一天的时间。然而这个轨道只能同时给一艘飞船使用，也就是说，每条轨道每天只能有一艘飞船穿越。现在IBM公司要把K（K≤50）台超级计算机从地球运到Eisiem星系去，由于这些超级计算机个头巨大，一台计算机就要用一艘飞船来运。现在人类能够到达N（N≤50）个星系，拥有M（M≤200）条超时空轨道，太阳系的编号为S，Eisiem星系的编号为T。你需要求出至少需要几天才能将这些超级计算机全部运到目的地。注意，IBM公司是非常NB的公司，所有的超时空轨道都会优先给IBM公司使用。

Poj1895 bring them there

- 从小到大枚举答案ans，构造(ans+1)层的图，每层都是原图的n个点。
- 源点连第一层的S，容量为k；
- 所有层的T连汇点，容量为inf；
- 每层的点i 都向下一层的点i 连边，容量为inf；
- 对于原图的无向边(u,v)，在相邻两层之间连边(u,v),(v,u)，容量1。
- 求最大流，如果最大流等于k说明可以满足题目要求，找到了答案。
- 每次不用清空以前的图，直接加一层继续增广即可。这样的时间复杂度基本上相当于只在最终的(ans+1)层的图上求了一个最大流。

Poj1895 bring them there

- 输出方案：从源点出发dfs k次，得到k条路径。每次dfs时寻找一条从当前点出发有流的边走过去，并且把这条边的流减一，直到到达汇点。注意以下几点：
- (1) 如果从当前层的i 走到了下一层的i ，实际上这一天仍在i 星系中没有移动，因此这条边不能记录到方案里。
- (2) 如果相邻两层之间的(u,v)和(v,u)都有流，根据题目要求，一条路上不能有两个不同方向的流同时流过。但是这样的方案可以等效成两个流各自停留在u和v没有移动，没有流过这两条边，而在后边的路程中两个流的路径进行了交换。因此此时的边也不能记录到方案中。

最小路径边覆盖

- 题意：求有向无环图可重叠的最小路径边覆盖（二分图中解决的是点覆盖）。

最小路径边覆盖

- 题意：求有向无环图可重叠的最小路径边覆盖（二分图中解决的是点覆盖）。
- 如果不能重复覆盖，那么记一个点的入度为in[i]，出度为out[i]，那么答案就是∑max(in[i]-out[i],0)。方案直接搜索就行了。
- 重复走一条边，可以看做加入了一条重边。若加入边(u,v)，则out[u]++, in[v]++。添加之前如果in[u]>out[u]，in[v]<out[v]，那么这次加边会使答案变优1。
- 进一步扩展，如果把连续的若干条边加上重边，那么相当于这条路径的起点s，out[s]++，终点t，in[t]++。
- 问题变为：添加一些从入度>出度的点到入度<出度的点的路径，使答案最优。

最小路径边覆盖

- 我们根据以往的经验知道，网络流寻找路径的问题，应当把每个点拆点，然后把路径拆成若干条边、以及拆点之后两个同点之间的边。
- 把每个点拆成左、右两个，右点向左点连容量为+∞的边。
- 从源点S向所有in[i]>out[i]的左点连边，容量为in[i]-out[i]。
- 从所有in[i]<out[i]的右点向汇点T连边，容量为out[i]-in[i]。
- 对于原图中的边(u,v)，从u的左点向v的右点连容量为+∞的边。
- 求最大流，那么答案就是满流减去最大流。
- 一条边上有多少流量，就要添加多少条重边，然后dfs输出方案。

星际竞速 - Bzoj1927

- http://www.lydsy.com/JudgeOnline/problem.php?id=1927

星际竞速 - Bzoj1927

- http://www.lydsy.com/JudgeOnline/problem.php?id=1927
- 每个点只访问一次，就是说回路中每个点的入度出度都是1。这有点类似于匹配，启发我们用二部图来处理。
- 如果能够选出一些边，使得每个点仅包含在一条入边和一条出边里，那么最后把这些边组合一下就可以得到答案。
- 每个点拆成一个入点（右边一排）和一个出点（左边一排）。

星际竞速 - Bzoj1927

- 源点S向出点连容量1费用0的边，入点向汇点连容量1费用0的边。
- 高速航道(x,y)，x<y，从x的出点到y的入点连容量1费用为边权的边。
- 从S向所有入点连容量1费用为定位费用的边。
- 求最小费用最大流。
- 到汇点的边的容量限制、以及最大流保证了每个入点有且仅有一条有流量的入边。这条边要么从某个出点来，要么从源点定位瞬移过来。