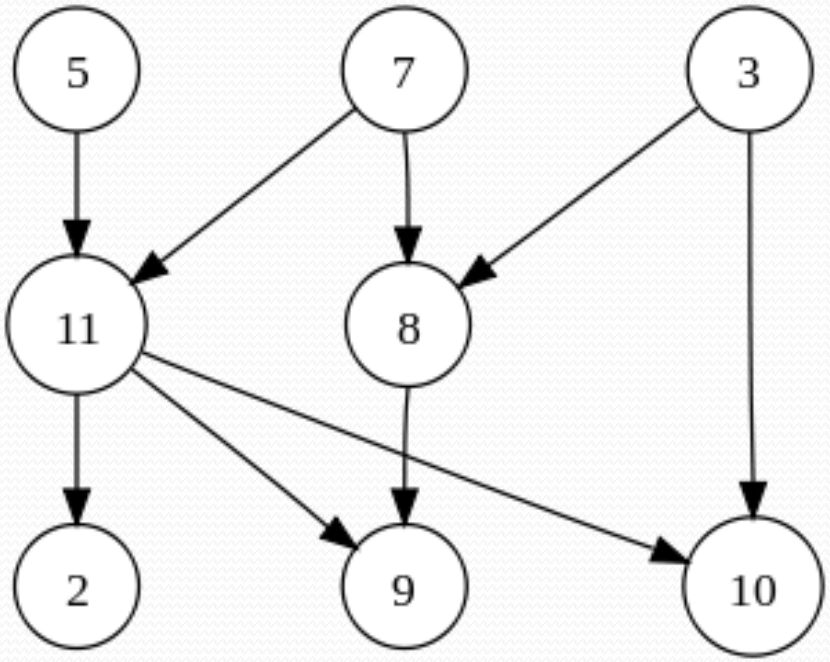
Network Flow

1120132001

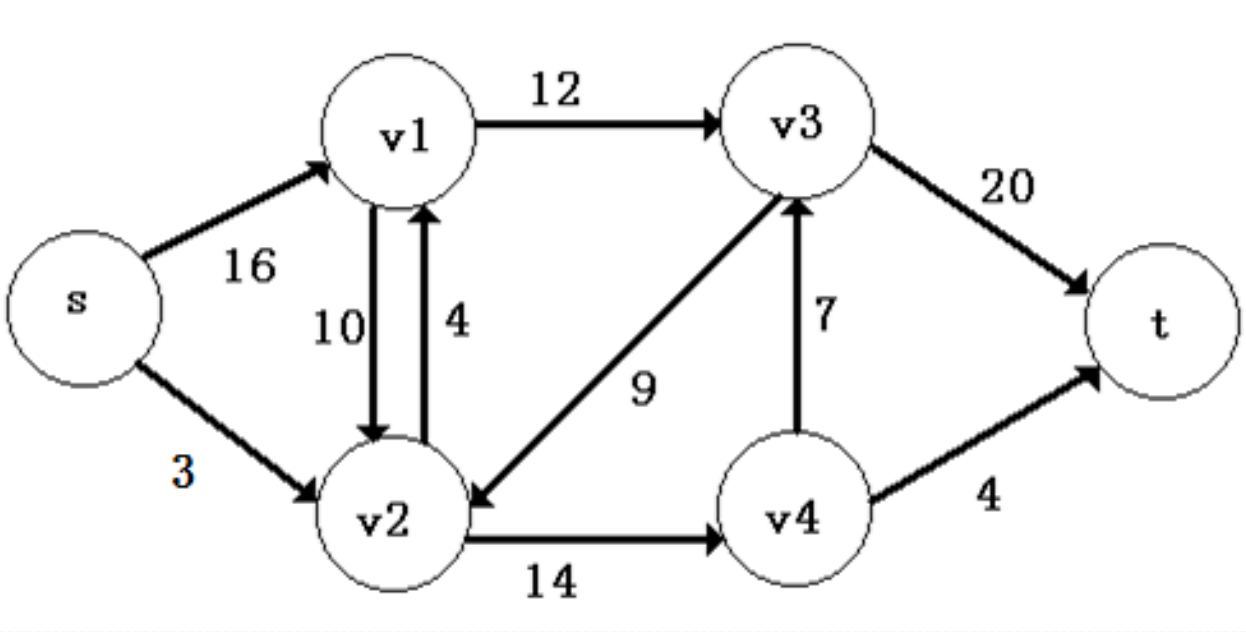
Directed Graph



- Vertices/Nodes
- Edges/Arcs
- Weights

Transportation Network

- Source
- Sink
- Capacity
- Distribution nodes
- Supply/Demand

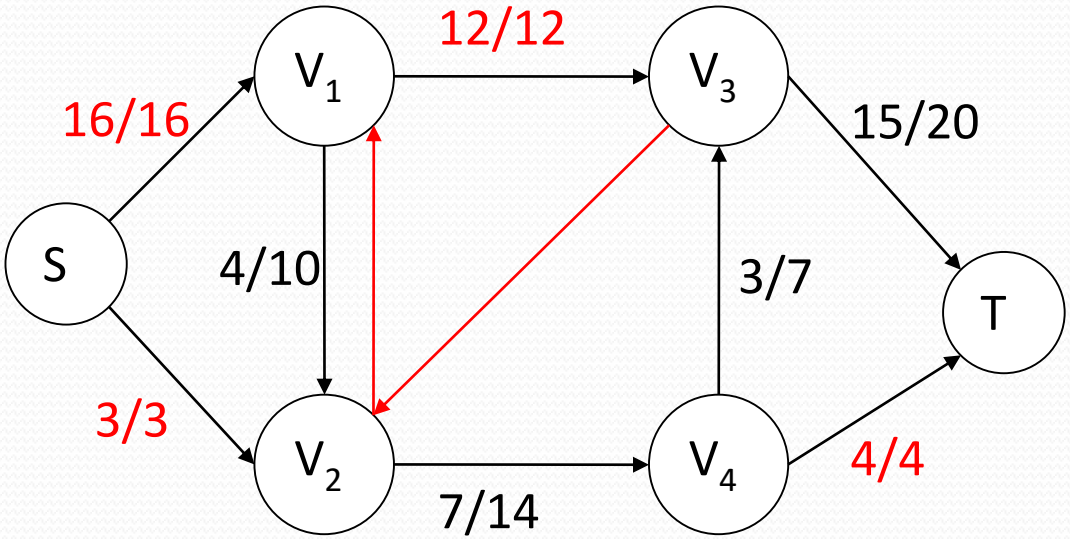


Network Flow

- Flow : a function from arcs to real number, satisfying
- Capacity : $f(u,v) \leq c(u,v)$
- Symmetry: $f(u,v) = -f(v,u)$
- Conservation: $\sum_{(u,v)\in E} f(u,v) = \sum_{(v,z)\in E} f(v,z)$

Max-flow Problem

- The maximum flow in a network.



- The max-flow is 16 + 3 = 15 + 4 = 19.
- Solution? SEARCH is plausible.....

Linear Programming (LP)

$$\min \mathbf{c}^T \mathbf{x}$$

$$\text{st.} \begin{cases} \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ \mathbf{x} \geq 0 \end{cases}$$

$$\mathbf{c}, \mathbf{x} \in R^n, \mathbf{b} \in R^m, A \in R^{m \times n}$$

对偶问题 (DP)

$$\max \mathbf{b}^{\mathrm{T}}\mathbf{y}$$

$$\text{st.}\begin{cases}\mathbf{A}^{\mathrm{T}}\mathbf{y} \leq \mathbf{c} \\ \mathbf{y} \geq 0\end{cases}$$

强对偶定理

- 对偶问题的任一可行解，其目标函数值小于原始问题的任一可行解对应的目标函数值。
- 两个问题或者都有有限的最优解，或者都没有。
- 如果两个问题存在最优解，在此最优解下两目标函数必相等。

互补松弛性定理

- 对于最优解X, Y（向量）和任意下标j
- 如果X[j]非o，那么Y的第j个约束一定为紧约束。
- 反之，如果Y的第j个约束不紧，那么X[j]=o

$$\forall 1 \leq j \leq n, \left(\sum_{i=1}^{m} A_{ij} Y_i - c_j \right) X_j = 0$$

全幺模矩阵(totally unimodular)

- 线性规划所有参数为正数不代表解为整数。
- 整数规划问题是NP问题。

- 然而.....
- 如果线性规划的矩阵A是全幺模矩阵，则此线性规划之解必为整点解。

- 定理：有向图和无向二分图的关联矩阵为全幺模矩阵。

何谓标号(label)

- 考虑s-t最短路问题。每条边或者在s-t的最短路上，或者不在。令x[i,j]表示之：1为在，0为不在。有下式：

$$\min \sum_{(i,j)\in A} w_{ij}x_{ij}$$

$$\text{s.t.} \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = \begin{cases} 1, & i = s, \\ -1, & i = t, \\ 0, & i \neq s,t. \end{cases}$$

$$x_{ij} \geqslant 0.$$

何谓标号

- 考虑其对偶问题。
- 原始变量在边，则对偶变量在点。
- 有下式：

$$\max(u_t - u_s)$$

$$\text{s.t. } u_j - u_i \leqslant w_{ij}, \forall\ (i,j) \in A.$$

- 由互补松弛性有：

$$x_{ij}(u_j - u_i - w_{ij}) = 0, \forall\ (i,j) \in A.$$

何谓标号

- 考虑到u的绝对值毫无意义，相对值有价值，定义u[s]=o
- 于是显见有，对st路上的边有u[j]=u[i]+w[i,j]
- 得到递推关系（DP方程）：

$$\begin{cases} u_s = 0, \\ u_j = \min\limits_{i \neq j} \{u_i + w_{ij}\}. \end{cases}$$

- u[i]可以理解为距离。

何谓标号

- 标号之于边权函数：

- 电流与电压
- 势能与速度
- 距离与最短路
- ？ 与流量

总结(1)

- 已经学过的内容 => 遗留的问题

- 网络与最大流 => 如何通过搜索计算最大流
- 线性规划 => 网络流问题的数学定义
- 原始对偶定理 => 最大流的对偶问题
- 标号设定 => 标号对于网络流问题有什么用

数学表述

$$\max\ v$$

$$\text{s.t.} \sum_{j,(i,j)\in A} x_{ij} - \sum_{j,(j,i)\in A} x_{ji} = \begin{cases} v, & i = s \\ -v, & i = t, \\ 0, & i \neq s, t, \end{cases}$$

$$0 \leqslant x_{ij} \leqslant u_{ij}, \qquad \forall\ (i, j) \in A.$$

- 线性规划

例题1、可行流问题

- 有的点有一些有限的供应量，有的点有一些需求量。
- 要求在流量满足约束条件的情况下满足供需平衡。
- 给出任一可行解。

- 转化：
- 新建源点s汇点t。
- 源点s到所有供应点连弧，流量为供应量。
- 所有需求点到汇点t连弧，流量为需求量。
- 跑一边最大流，看能不能流满供需量。

上下界的网络流

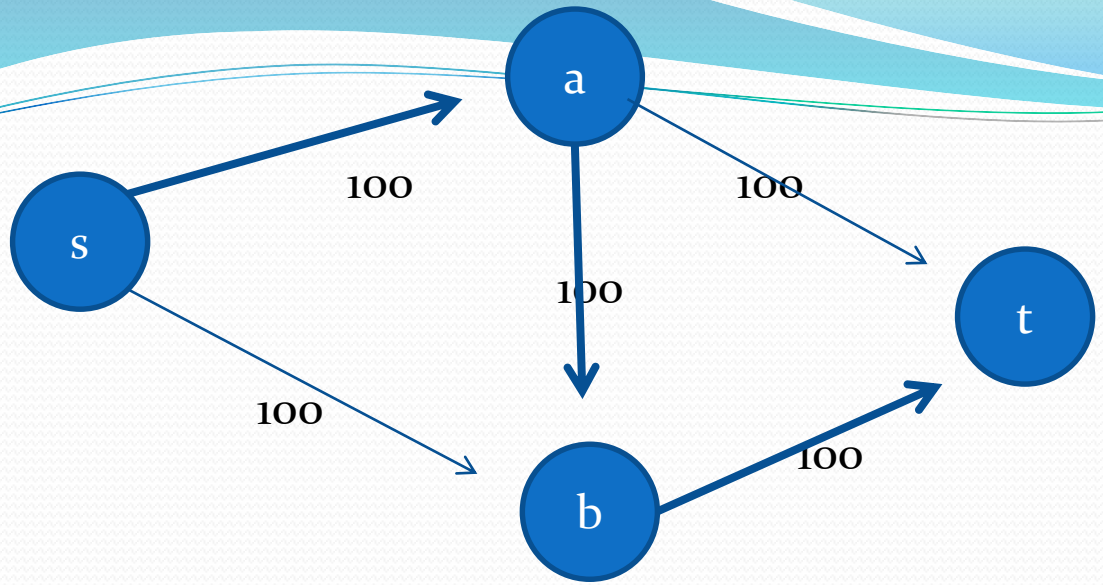
- 如果给上一题中的某些边增加流量下界，即要求这条边最小流过若干容量，怎么解？

- 转化：
- s向强流边终点连流量为下界的弧
- 强流边起点向汇点连流量为下界的弧
- 原始的边流量减去下界，直接做。
- （这样，相当于预先把必须流的流量流满）
- 看能不能满流就行

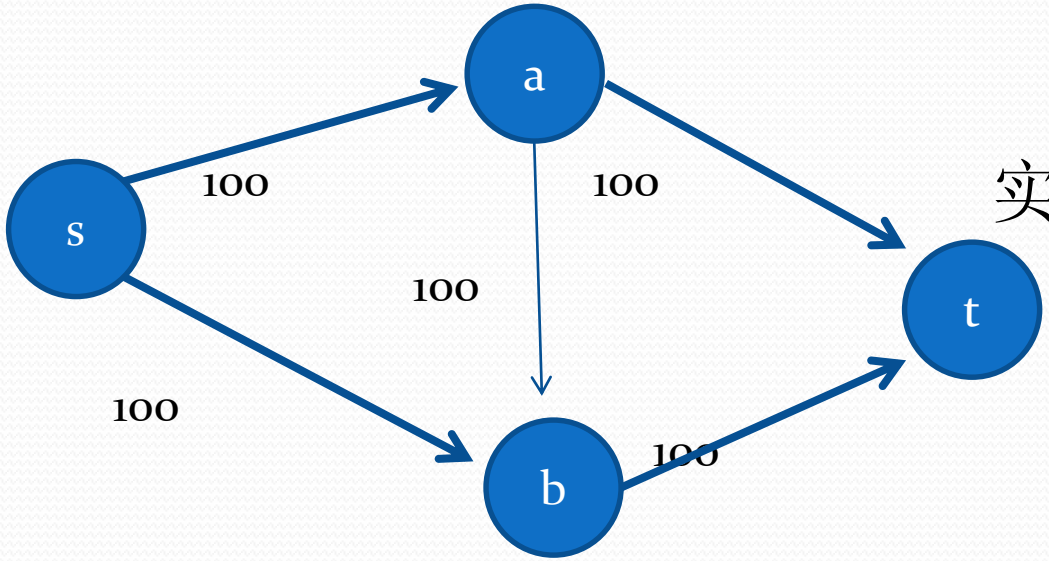
反向弧

- 一条流量达到上界的弧称为饱和弧(saturated arc)
- 流量为o的弧称为空弧(void arc)

- 暴搜的算法：
- DFS找到一条s-t路，向这条路上增加流量。显然最大只能增加到这条路上最小的（容量-流量）的值。
- 但是我们可能犯错误.....



- 如果我们沿着s-a-b-t路线走 仅能得到一个100的流



实际上此图存在流量为200的流

反向弧

- 我们需要反悔的机会！
- 对每条弧，增加一条与其反向，容量相等，但流量为满值的弧（不影响原网络）
- 每当增加一条弧的容量的时候我们就相应地减少反向弧的流量，反之亦然。
- 死循环？
- （搜索过程中，流量是单调增加的）

反向弧

- 注意：反向弧是一种实现方式，不是数学内容。

- 从数学上表述算法，我们暴搜的时候是按无向边搜索路径的。其中，和路径方向相同的边增加流量，相反的边减少流量。

- 以后，表述数学定义的时候按弧与路同向反向来，表述算法实现的时候按另建反向弧来。
- PS:增加了反向弧后，我们得到了一个慢的要死但是能过的DFS暴搜算法，称为Ford-Fulkerson算法.....

流分解定理

- 给我们暴搜中遇到的东西一个数学定义吧：

- 路流(path flow)：一条路，所有边流量全为v，v小于路上所有边的容量。v称为流值。
- 圈流(cycle flow)：一个圈，所有边流量全为v，v小于圈上所有边的容量。v称为流值。

- 显然，任何一个流，不管是不是最大的，可以分解为少于m+n个s-t的路流和圈流之和。其中圈流不超过m个。（m是边数，n是点数）
- 称为流分解定理。

增广路

- DFS的过程中，不能增加流量的s-t路是无意义的。
- 定义：在一个已经有一定流量的网络上，如果一条s-t路P满足如下条件：
- 1.P上，所有和P同向的弧都不是饱和弧
- 2.P上，所有和P逆向的弧都不是空弧
- 那么P称为一条**增广路**。所有反向弧当前流量，及所有正向弧容量减流量之差，取最小值v，是这个路的增广值。

增广路

- 对于一条增广路，所有正向弧增加增广值v，逆向弧减少增广值v，得到的流一定比原来的大。
- 这种构造性增加流量的方法称为增广。

- 一个赋有流量的网络称为残量网络(residual network)
- 残量网络增广后，一定变成流量比原来大的残量网络。所以可以初步认为最大流一定是不能增广的残量网络。
- 从这个角度，我们可以得到一些性质.....

割

- 对于任一网络（可以理解为没有流量），把所有的点分为两个集合S与T，要求s在S中，t在T中。称集合(S,T)为s-t割。
- 定义s-t割的容量（割值）为所有起点在S中而终点在T中的前向弧的容量之和。（注意：反向弧不算，而且T到S的弧的容量也不要减）
- 最小的s-t割称为最小割。

最小割：对偶

- 任一流的流值不超过任一割的割值：

$$
\begin{aligned}
v(x) &= \sum_{i\in S}\Big(\sum_{j\in V} x_{ij} - \sum_{j\in V} x_{ji}\Big) \\
&= \sum_{i\in S}\sum_{j\in S}(x_{ij} - x_{ji}) + \sum_{i\in S}\sum_{j\in T}(x_{ij} - x_{ji}) \\
&= \sum_{i\in S}\sum_{j\in T}(x_{ij} - x_{ji}) \\
&\leqslant \sum_{i\in S}\sum_{j\in T} x_{ij} \qquad (\text{因为 } x_{ji} \geqslant 0) \\
&\leqslant \sum_{i\in S}\sum_{j\in T} u_{ij} \qquad (\text{因为 } x_{ij} \leqslant u_{ij}) \\
&= U(S, T)
\end{aligned}
$$

最小割定理

- 对任一网络，最大流等于最小割。

现有的算法

- FF算法
- Edmonds-Karp 算法(EK)
- SAP算法（常用）
- Dinic算法 （N^2M，常用且较快）
- 预留推进算法（一般不用）
- 最高标号预流推进算法（N^3，一般不用）
- 图单纯形法（NM，从来不用）

Dinic算法简介

- 用BFS对残余网络分层，设定标号为到S的距离（经过的边数）
- 增广过程中，每一步必须走向不同层的节点。
- 详见张伯威PPT。

- 优化：
- 当前弧优化
- 多路增广优化
- 网上有资料，也可以看我的代码。

例题2. POJ2112

- 有K台挤奶机器和C头牛(统称为物体），每台挤奶机器只能容纳M头牛进行挤奶。现在给出dis[K + C][K + C]的矩阵，dis[i][j]若不为o则表示第i个物体到第j个物体之间有路，dis[i][j]就是该路的长度。（1 <= K <= 30,1 <= C <= 200）

- 现在问你怎么安排这C头牛到K台机器挤奶，使得需要走最长路程到挤奶机器的奶牛所走的路程最少，求出这个最小值。

利用Floyd算法求出每个奶牛到每个挤奶机的最短距离。

则题目变为：

已知C头奶牛到K个挤奶机的距离，每个挤奶机只能有M个奶牛，每个奶牛只能去一台挤奶机，求这些奶牛到其要去的挤奶机距离的最大值的最小值。

网络流模型：

每个奶牛最终都只能到达一个挤奶器，每个挤奶器只能有M个奶牛，可把奶牛看做网络流中的流。

每个奶牛和挤奶器都是一个节点，添加一个源，连边到所有奶牛节点，这些边容量都是1。

添加一个汇点，每个挤奶器都连边到它。这些边的容量都是M。

网络流模型：

先假定一个最大距离的的最小值 maxdist, 在上述图中，如果奶牛节点i和挤奶器节点j之间的距离<= maxdist,则从i节点连一条边到j节点，表示奶牛i可以到挤奶器j去挤奶。该边容量为1。该图上的最大流如果是C(奶牛数），那么就说明假设的 maxdist成立，则减小 maxdist再试

总之，要二分 maxdist, 对每个maxdist值，都重新构图，看其最大流是否是C，然后再决定减少或增加 maxdist

例题3. HOJ2634

- 有M个项目和N个员工。做项目i可以获得Ai元，但是必须雇用若干个指定的员工。雇用员工j需要花费Bj元，且一旦雇用，员工j可以参加多个项目的开发。问经过合理的项目取舍，最多能挣多少钱。（1 <= M, N <= 100）

最小割建模

- 我们把源连向所有人，流量为雇佣的花费Bj。把所有项目连向汇，流量为收益Ai。然后每个人连向需要他的项目，流量为无穷。

- 对于一个项目来讲，它有两种割法。
- 一种是割掉那些连向他的人的汇连向他的边，另一种是割掉这个项目连向汇的边。
- 刚好就对应了对这个项目的两种决策

- 答案就是sigma(Ai) - maxflow

例题4. HDU4940

- 给定一个强连通图网络，对于任何一个s-t划分[S,T]，其价值为所有S->T的边的A价值，减去T->S的边的B价值。任何一条边的A价值一定小于B价值。
- 问是不是所有的划分价值都是负的。

- 解法：
- 若存在一个可行流使得每条边的流量大于A而小于B，则显然对每一划分，其A值和小于B值和。

HDU 4940

证明：

设 f_{ij} 为图中从 i 到 j 的流量，若原图存在可行流，则对于任意非空真子集 S，流出该点集的流量必然等于流入该点集的流量：

$$\sum_{(i,j)\in[s,\bar{s}]} f_{ij} = \sum_{(i,j)\in[\bar{s},s]} f_{ij}$$

$$A_{ij} \le f_{ij} \le B_{ij}$$

联立上述两式：

$$\sum_{(i,j)\in[s,\bar{s}]} A_{ij} \le \sum_{(i,j)\in[\bar{s},s]} B_{ij}$$

最小割的若干应用

- 最大权闭合图
- 最大密度子图
- 二分图匹配
- 二分图最小点覆盖
- ……

- 其实是推荐论文的，建议读一下。
- 记不住就把构图方法放在板子里吧。这年头最小割考的比最大流多多了。

2014西安C题

- 给定一个序列，求它的一个子序列，其逆序对数之比于子序列长度最大。

- 这不是裸的最大密度子图么……

最小割查割边

- 边的流量不同时，最小割是流量数不是边数。
- 令所有零流边流量1，其他边流量无限大，再跑一次网络流，得到的就是最少的割边数目。
- 然后查看残余网络中，每条边的起点和终点是否从s可达。如果起点可以到达，而终点不可到达，那么它是割边。

总结(2)

- 最大流的数学模型
- 反向弧，增广路
- 上下界的转化
- 最小割定理
- 最小割经典构图

最小费用流问题

- 每个节点有一个供需量di，正为流入，负为流出。
- 每条边有一个流量上界ui。
- 每条边有一个单位费用ci。当留量为x时，需要支付代价x*ci。
- 在满足流量守恒的情况下，最小化总代价。

$$\min\ c(x) = \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$\text{s.t.} \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = d_i, \forall\ i \in V,$$

$$0 \leqslant x_{ij} \leqslant u_{ij},\ \forall\ (i,j) \in A.$$

最小费用最大流问题

- 标准的费用流是可行流问题。
- 然而如果增加源点s和汇点t，把正负供需量转化为s和t的连边，可以变成最小费用最大流。
- 亦即：求s到t的最大流，在流量最大的基础上，要求最小化费用。
- 通常我们假定最小费用最大流没有单位费用和为负值的环。
- 注意实现的时候，反向边的费用是正向费用的负值。

消圈算法

- 先构造一个任意的最大流。
- 然后暴力枚举所有费用和为负数的增广圈（自然是有反向边的），增广这个圈。
- 等到没有负权圈的时候，费用即到最小。

- 先最大流，再最小费用。
- 时间复杂度很高，没见有人写过.....

最小费用路增广法

- 用最短路算法找s到t的费用最小的路径。（必须SPFA）
- 然后每次沿着这条路径增广。
- 因为是每次取最小费用，所以结果费用一定也是最小的。因为没有增广路了，所以流量最大。

- 先最小费用，再最大流。
- 虽然不是多项式算法，但是最常用。

费用流的对偶

- 原始的问题：

$$\min \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$\text{s.t.} \sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = \begin{cases} v, & i = s, \\ -v, & i = t, \\ 0, & i \in V, i \neq s,t, \end{cases}$$

$$0 \leqslant x_{ij} \leqslant u_{ij}, \qquad (i,j) \in A.$$

费用流的对偶

- 对偶问题

$$\max\ w(\pi,z)=\sum_{i\in V}d_i\pi_i-\sum_{(i,j)\in A}u_{ij}z_{ij}$$

$$\text{s.t. } \pi_i-\pi_j-z_{ij}\leqslant c_{ij},\qquad (i,j)\in A,$$

$$z_{ij}\geqslant 0,\qquad (i,j)\in A.$$

- 式中π是节点流量守恒式的对偶变量，z是流量上界的对偶变量。

费用流的对偶

- 由互补松弛性定理：

$$x_{ij}(\pi_i - \pi_j - z_{ij} - c_{ij}) = 0, \qquad (i,j) \in A,$$
$$z_{ij}(x_{ij} - u_{ij}) = 0, \qquad (i,j) \in A.$$

- 或写成：

当 $\pi_i - \pi_j < c_{ij}$ 时，$x_{ij} = 0$；

当 $\pi_i - \pi_j > c_{ij}$ 时，$x_{ij} = u_{ij}$；

当 $0 < x_{ij} < u_{ij}$ 时，$\pi_i - \pi_j = c_{ij}$.

$$z_{ij} = \max\{\pi_i - \pi_j - c_{ij}, 0\}, \qquad (i,j) \in A,$$

费用流的势

- 注意最短路问题本身也是费用流。所以π可以当成距离看待。
- 定义每条边的既约代价：$c_{ij}^{\pi} = c_{ij} - \pi_i + \pi_j$

- 原始-对偶算法维护互补松弛性条件恒成立并为此不断增广和修改标号。于是在增广步骤中我们只沿着既约代价为o的弧增广，然后修改标号。

可行树定理

- 在最小费用可行流问题的最优解中，一定存在一颗生成树，使得除了树上的边以外，其他的弧不是空弧就是饱和弧。
- 满足流量守恒条件的这样一颗树称为可行树。可行树上全部边的既约代价为o。
- 进一步地，如果树上所有的空弧都想着远离根的方向，而饱和弧都朝向根，称为强可行树。

Network Simplex Method

- 网络单纯形法维护一棵强可行树，每次找到一条不在树上的边，在其与生成树树构成的环上消圈，然后加入这条边并删去一条树边，变成一棵新的强可行树。
- 可行树的变化可以用LCT优化和离开查找优化。
- 每次操作费用减少，流量维持，直到达到最小费用为止。
- 实践最快算法，做题优势不明显。点数破10万时用。

例题5 HOJ2713

- 一个N*M的网格，每个单元都有一块价值Cij的宝石。问最多能取多少价值的宝石且任意两块宝石不相邻。（1 <= N, M <= 50, 0 <= Cij <= 40000）

- 分析：
- 我们考虑下黑白染色之后的图形。
- 对于每个黑点上的宝石来说，你只有2个选项，选择这个宝石不选与它相邻的白格子中的宝石或者不选这个宝石选择与它相邻的白格子中的宝石。

解法

- 从源点到每个黑点连一条边，从每个白点到汇点连一条边，容量均为相应宝石的价值。每个黑点向与其相邻的四个白点连边，容量为无穷。
- 结果即为sigma(Cij) – maxflow。

- 类似的还有上海大都会赛的热身赛的B。
- 考场是个矩形，有些地方不能坐人，每个人可以看见左右以及左前右前的人的试卷。求最大安排人数。
- 二分图匹配，最大流直接搞即可。
- 好吧，这不是费用流的题.....

例题6 HDU4780

- 有一个糖果工厂，有M个机器生产N种糖果。你需要把N种糖果都生产出来。对于每个机器j，都可以生产任意糖果i。每种糖果有一个生产时间的要求(s_i, t_i)，意思是最晚要在s_i开始生产，在t_i这个时间完成生产。如果实际开始生产的时间是p_i($s_i < p_i < t_i$)，这种糖果依然会在t_i完成，不过需要额外花费K*(pi - si)。如果$p_i >= t_i$则无法进行生产。对于每个机器，他们在开始时(时间0)处于初始态。将处于初始态的机器j调整到生产糖果i的状态需要花费C_{ij}的时间和D_{ij}的钱。将任意一台处于i_1态的机器调整i_2态需要花费E_{i1i2}的时间和F_{i1i2}的钱。询问将N种糖果都生产出来的最小花费，无法完成生产输出-1。

分析

- 易知机器在生产完第一种糖果就会完全相同(当然状态是不同的)。由于糖果完成生产的时间是确定的，那么此时我们可以用最后生产的糖果来代表这个机器(这点比较显然吧)。那么思考我们需要表示的状态。
- 每台机器最开始的状态
- 他们生产过糖果之后的状态
- 貌似就够了的样子吧。

构图

- 于是我们将图分为三层(M1, M2, M3)：
  最开始的机器
  被某些机器不是在最开始生产的糖果
  生产完毕的糖果
- S->M1： 流量1，花费0，代表每个机器最开始生产的糖果。
- S->M2：流量1，花费0，代表不是被某台机器在最开始生产的那些糖果。
- M1->M3：流量1，花费为切换过去的代价(也许还有晚开始的花费)
- M2->M3：根据M2完成的时间和M3需要的开始时间决定流量以及花费。
- M3->T：流量1，花费0，表示完成生产。

例题7 HDU4085

- N个节点的无向图，前K个节点为住民，后K个节点为避难所。要求在图中修最短的路，使得每个住民可以到达一个避难所，而且没有两个住民公用一个避难所。
- N只有50，K只有5.
- 怎么构图？
- 没法构图，这题根本就不是网络流……

解法

- 首先我们知道，最优解必然是一棵树，这棵树又是由若干棵子树合并成。于是我们可以状态压缩。
- dp[i][j]表示以i为根和连通状态为j的子树的最小权值。有两种转移方法：
- 枚举子树的形态：dp[i][j]=min{ dp[i][j], dp[i][k]+dp[i][l] }，其中k和l是对j的一个划分。
- 按照边进行松弛：dp[i][j]=min{ dp[i][j], dp[i'][j]+w[i][i'] }，其中i和i'之间有边相连。
- Floyd预处理距离。

要点

- 网络流一般不是水题就是神题。
- 网络流具体算法不重要，板子会用就行。重点在构图。要想构图好，理论知识最好都清楚。
- 看着像DP，其实可能是二分图网络流。
- 看着像网络流，其实可能是DP或者LCA。
- 做题经验是一方面；不要被经验所误导，注意问题本身的性质，发散思考，因地制宜是另一方面。

谢谢！