网络流

diamond_duke

2019 年 7 月 1 日

最大流

最大流

定义

设 $N=(V,E)$ 为一个网络，其中 s 和 t 分别是 N 的源点和汇点 $(s,t\in V)$。

一个边的**容量**为映射 $c:E\to\mathbb{R}^+$，记为 c_{uv} 或 $c(u,v)$。它表示可以通过一条边的流量的最大值。

一个**流**为一个映射 $f:E\to\mathbb{R}^+$，记为 f_{uv} 或 $f(u,v)$，遵循下面两个限制：

- 对于每个 $(u,v)\in E$，有 $f_{uv}\le c_{uv}$（即容量限制：一个边的流量不能超过它的容量）；
- 对于每个 $v\in V\setminus\{s,t\}$，有 $\sum_{u:(u,v)\in E}f_{uv}=\sum_{u:(v,u)\in E}f_{vu}$（即流量平衡：流入一个节点的流的总和必须等于流出这个节点的流的总和，源点和汇点除外）。

流量定义为 $|f|=\sum_{v:(s,v)\in E}f_{sv}$，其中 s 为 N 的源点，它表示从源点到汇点的流的数量。

最大流问题就是最大化 $|f|$，即从 s 点到 t 点尽可能规划最大的流量。

Dinic 算法

设 $G=((V,E),c,s,t)$ 为一个每条边的容量为 $c(u,v)$，流为 $f(u,v)$ 的网络。**残留容量** $c_f\colon V\times V\to R^+$ 的定义为：

$$c_f(u,v)=(c(u,v)-f(u,v))\cdot[(u,v)\in E]$$
$$c_f(v,u)=(f(u,v))\cdot[(u,v)\in E]$$

则**残留网络**为 $G_f=\left((V,E_f),c_f|_{E_f},s,t\right)$，其中

$$E_f=\{(u,v)\in V\times V\colon c_f(u,v)>0\}$$

Dinic 算法

设 $G=((V,E),c,s,t)$ 为一个每条边的容量为 $c(u,v)$，流为 $f(u,v)$ 的网络。**残留容量** $c_f\colon V\times V\to R^+$ 的定义为：

$$c_f(u,v)=(c(u,v)-f(u,v))\cdot[(u,v)\in E]$$
$$c_f(v,u)=(f(u,v))\cdot[(u,v)\in E]$$

则**残留网络**为 $G_f=\left((V,E_f),c_f|_{E_f},s,t\right)$，其中

$$E_f=\{(u,v)\in V\times V\colon c_f(u,v)>0\}$$

增广路指通过残留网络 G_f 的从源点 s 到汇点 t 的一条有效路径。
定义 $\operatorname{dist}(v)$ 为 G_f 中从源点 s 到点 v 的最短距离。那么 G_f 的**高度标号**为 $G_L=(V,E_L,c_f|_{E_L},s,t)$，其中

$$E_L=\{(u,v)\in E_f:\operatorname{dist}(v)=\operatorname{dist}(u)+1\}$$

设图 $G'=(V,E'_L,s,t)$，其中 $E'_L=\{(u,v):f(u,v)<c_f|_{E_L}(u,v)\}$ 不包含从 s 到 t 的路径，则**阻塞流**为一条从 s 到 t 的流 f。
算法流程即为不断找到阻塞流 f' 后，将当前的流 f 增加到 f'。

最小费用最大流

最小费用最大流

定义

带权有向图 $G=(V,E)$ 是一个特殊的容量网络, 所有边 $(u,v)\in E$ 包含 $c(u,v)\in\mathbb{R}^+$，称为这条弧的容量；以及 $w(u,v)\in\mathbb{R}$ 称为这条边的**费用**。

容量网络中一个可行流的总费用为 $\sum(f(u,v)\times w(u,v))$。所有最大流中总费用最少的称为这个容量网络的**最小费用最大流**。

求解

在最短路问题中，我们利用 SPFA 算法求单源最短路，进而得到两个结点之间的最短路径 $dis_{u\to v}$。
使用类似的思想，将两点之间的距离转换为两点之间的费用，然后运行 SPFA 算法，同时维护可以从源点到达每个点的最大流量，得到从源点到汇点一条费用最小的增广路，并使用这条路径进行增广。
然后重复这个过程，直到找不到增广路。

求解

在最短路问题中，我们利用 SPFA 算法求单源最短路，进而得到两个结点之间的最短路径 $dis_{u\to v}$。
使用类似的思想，将两点之间的距离转换为两点之间的费用，然后运行 SPFA 算法，同时维护可以从源点到达每个点的最大流量，得到从源点到汇点一条费用最小的增广路，并使用这条路径进行增广。
然后重复这个过程，直到找不到增广路。
具体而言，记源点为 s，汇点为 t。设 $u \in V$，$d(u)$ 代表从 s 到 u 每单位流量花费的最小费用，$f(u)$ 代表使用上述每单位流量花费费用最小的路径能够让多少流量从源点流到 u。

求解

在最短路问题中，我们利用 SPFA 算法求单源最短路，进而得到两个结点之间的最短路径 $dis_{u\to v}$。
使用类似的思想，将两点之间的距离转换为两点之间的费用，然后运行 SPFA 算法，同时维护可以从源点到达每个点的最大流量，得到从源点到汇点一条费用最小的增广路，并使用这条路径进行增广。
然后重复这个过程，直到找不到增广路。
具体而言，记源点为 s，汇点为 t。设 $u\in V$，$d(u)$ 代表从 s 到 u 每单位流量花费的最小费用，$f(u)$ 代表使用上述每单位流量花费费用最小的路径能够让多少流量从源点流到 u。
在 SPFA 每一轮循环过程中，从队列中取出一个结点 u，并枚举每一条边 $(u,v)\in E$，如果满足 $d(v)>d(u)+w(u,v)$ 则更新相应的 $d(v)=d(u)+w(u,v)$ 和 $f(v)=\min\{f(u),f(u,v)\}$，同时记录 $last(v)$ 代表来到结点 v 使用了哪一条弧。

求解

在最短路问题中，我们利用 SPFA 算法求单源最短路，进而得到两个结点之间的最短路径 $dis_{u\to v}$。
使用类似的思想，将两点之间的距离转换为两点之间的费用，然后运行 SPFA 算法，同时维护可以从源点到达每个点的最大流量，得到从源点到汇点一条费用最小的增广路，并使用这条路径进行增广。
然后重复这个过程，直到找不到增广路。
具体而言，记源点为 s，汇点为 t。设 $u\in V$，$d(u)$ 代表从 s 到 u 每单位流量花费的最小费用，$f(u)$ 代表使用上述每单位流量花费费用最小的路径能够让多少流量从源点流到 u。
在 SPFA 每一轮循环过程中，从队列中取出一个结点 u，并枚举每一条边 $(u,v)\in E$，如果满足 $d(v)>d(u)+w(u,v)$ 则更新相应的 $d(v)=d(u)+w(u,v)$ 和 $f(v)=\min\{f(u),f(u,v)\}$，同时记录 $last(v)$ 代表来到结点 v 使用了哪一条弧。
求出单源最短路后，就等同于找到了一条增广路，花费 $f(t)\times d(t)$ 将流量增大 $f(t)$。增广结束后，我们需要更新这条增广路上弧和反向弧的流量。

上下界网络流

上下界网络流

无源汇上下界网络流

首先，因为我们所有边的流量不得小于其下界，我们将所有边的流量都设为其下界，然后构建另一套图，即原图的残量网络。

无源汇上下界网络流

首先，因为我们所有边的流量不得小于其下界，我们将所有边的流量都设为其下界，然后构建另一套图，即原图的残量网络。
仅仅这样的话很可能不是一个可行解，因为这个流不满足流量守恒。
此时原来流量不平衡的点需要用**虚拟源汇**来补偿流量，因此我们需要构建虚拟源点 SS，虚拟汇点 TT:

- 若 i 点原来入流量**大于**出流量，则 SS 向 i 连一条容量差的边；
- 若 i 点原来入流量**小于**出流量，则 i 向 TT 连一条容量差的边；

无源汇上下界网络流

首先，因为我们所有边的流量不得小于其下界，我们将所有边的流量都设为其下界，然后构建另一套图，即原图的残量网络。
仅仅这样的话很可能不是一个可行解，因为这个流不满足流量守恒。
此时原来流量不平衡的点需要用**虚拟源汇**来补偿流量，因此我们需要构建虚拟源点 SS，虚拟汇点 TT:

- 若 i 点原来入流量**大于**出流量，则 SS 向 i 连一条容量差的边；
- 若 i 点原来入流量**小于**出流量，则 i 向 TT 连一条容量差的边；

因为当且仅当 SS 的所有连边都满流时有解，所以跑 SS 到 TT 的最大流即可。最终流量即为下界加上新图中的流量。

有源汇上下界网络流

- 先考虑如何求出一个 **可行流**：

有源汇上下界网络流

- 先考虑如何求出一个 **可行流**：
  考虑连一条 T 到 S 的无限制的边，那么有源汇就转化成了无源汇。

有源汇上下界网络流

- 先考虑如何求出一个 **可行流**：
  考虑连一条 T 到 S 的无限制的边，那么有源汇就转化成了无源汇。
  根据流量守恒，可行流的流量就是 T 到 S 那条新加边的流量。

有源汇上下界网络流

- 先考虑如何求出一个 **可行流**：
  考虑连一条 T 到 S 的无限制的边，那么有源汇就转化成了无源汇。
  根据流量守恒，可行流的流量就是 T 到 S 那条新加边的流量。
- 然后再考虑 **最大流**：

有源汇上下界网络流

- 先考虑如何求出一个 **可行流**：
  考虑连一条 T 到 S 的无限制的边，那么有源汇就转化成了无源汇。
  根据流量守恒，可行流的流量就是 T 到 S 那条新加边的流量。
- 然后再考虑 **最大流**：
  先求出一个可行流，如果有解，那么 SS 的所有边都满流了，就不用管它了。接下来只要求 S 到 T 的最大流即可。

有源汇上下界网络流

- 先考虑如何求出一个 **可行流**：
  考虑连一条 T 到 S 的无限制的边，那么有源汇就转化成了无源汇。
  根据流量守恒，可行流的流量就是 T 到 S 那条新加边的流量。
- 然后再考虑 **最大流**：
  先求出一个可行流，如果有解，那么 SS 的所有边都满流了，就不用管它了。接下来只要求 S 到 T 的最大流即可。
- 然后是 **最小流**：

有源汇上下界网络流

- 先考虑如何求出一个 **可行流**：
  考虑连一条 T 到 S 的无限制的边，那么有源汇就转化成了无源汇。
  根据流量守恒，可行流的流量就是 T 到 S 那条新加边的流量。
- 然后再考虑 **最大流**：
  先求出一个可行流，如果有解，那么 SS 的所有边都满流了，就不用管它了。接下来只要求 S 到 T 的最大流即可。
- 然后是 **最小流**：
  还是要先求出可行流，然后此时需要把 S 到 T 的流量尽量减小。

有源汇上下界网络流

- 先考虑如何求出一个 **可行流**：
  考虑连一条 T 到 S 的无限制的边，那么有源汇就转化成了无源汇。
  根据流量守恒，可行流的流量就是 T 到 S 那条新加边的流量。
- 然后再考虑 **最大流**：
  先求出一个可行流，如果有解，那么 SS 的所有边都满流了，就不用管它了。接下来只要求 S 到 T 的最大流即可。
- 然后是 **最小流**：
  还是要先求出可行流，然后此时需要把 S 到 T 的流量尽量减小。
  考虑反向弧的本质：反向弧增加意味着正向弧减少。所以求 T 到 S 的最大流即可。

例题

例题

「SHOI 2017」寿司餐厅

有 n 种寿司，每种寿司有代号 a_i，每种寿司都有无限多。
每次可以取一个区间 $[L, R]$，并吃掉其中每个寿司各一个。收益以及代价的计算方式如下：

- 对于任意区间 $[l, r]$，如果存在一次取的寿司 $[L, R]$ 满足 $[l, r] \subseteq [L, R]$，则获得 $d_{l,r}$ 的收益，否则不获得收益；
- 如果吃了 c **种代号**为 x 的寿司，则需要花费 $mx^2 + cx$，其中 m 为常数。

最大化总收益减去总花费。
$1 \leq n \leq 100$。

「JSOI2018」绝地反击

有 n 个点 (x_i, y_i)，你要把它们移动到一个半径为 R，圆心为原点的单位圆上，使得相邻两个点距离相同（即一个正 n 边形，每个顶点距离原点的距离都是 R）。最小化最大移动距离的最小值。

$3 \leq n \leq 200$。

「TJOI2015」线性代数

给出一个 $N \times N$ 的矩阵 $\mathbf{B}$ 和一个 $1 \times N$ 的矩阵 $\mathbf{C}$。
求出一个 $1 \times N$ 的 $0/1$ 矩阵 $\mathbf{A}$，使得 $D = (\mathbf{AB} - \mathbf{C})\,\mathbf{A}^T$ 最大。输出 D。
$1 \leq N \leq 500$。

「TJOI2015」线性代数 Solution

$$D=\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{A}_i\mathbf{A}_j\mathbf{B}_{i,j}-\sum_{i=1}^{N}\mathbf{A}_i\mathbf{C}_i$$

「TJOI2015」线性代数 Solution

$$D=\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{A}_i\mathbf{A}_j\mathbf{B}_{i,j}-\sum_{i=1}^{N}\mathbf{A}_i\mathbf{C}_i$$

$$\sum_{x\in S}\left(\sum_{y\in S}\mathbf{B}_{x,y}-\mathbf{C}_x\right)$$

AtCoder Grand Contest 031 Problem E

有 N 个珠宝，第 i 个位于 (x_i, y_i)，价值为 v_i。你可以选择一些珠宝，有如下四种限制：

- $x \leq a_i$ 的珠宝只能选择不超过 b_i 个；
- $x \geq a_i$ 的珠宝只能选择不超过 b_i 个；
- $y \leq a_i$ 的珠宝只能选择不超过 b_i 个；
- $y \geq a_i$ 的珠宝只能选择不超过 b_i 个；

$1 \leq N \leq 80$，$1 \leq x_i, y_i, a_i \leq 100$。

微积分真简单

给一个多项式 P，P 的每一项都是 x_i，y_i 或者 x_iy_j（$1 \le i,j \le n$）的形式。判断下式是否满足：

$$\frac{\partial}{\partial x_1}\frac{\partial}{\partial y_1}\frac{\partial}{\partial x_2}\frac{\partial}{\partial y_2}\cdots\frac{\partial}{\partial x_n}\frac{\partial}{\partial y_n} \quad P^k\Big|_{x_1,x_2,y_1,y_2,\ldots,x_n,y_n=0} = 0 \qquad (1)$$

例如，当 $n=1$，$k=2$，$p=x_1+y_1$ 时，有：

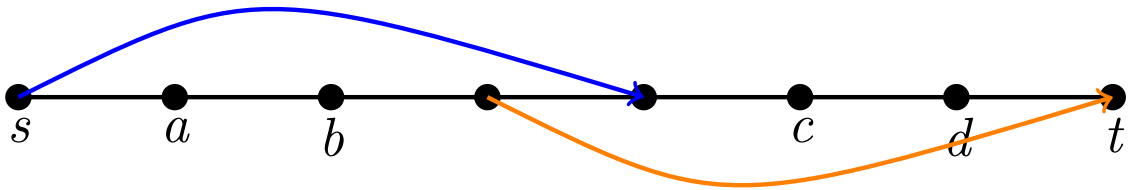
$$\begin{aligned}
&\frac{\partial}{\partial x_1}\frac{\partial}{\partial y_1}\left(x_1+y_1\right)^2\Big|_{x_1=0,y_1=0} \\
&= \frac{\partial}{\partial y_1} \quad 2x_1+2y_1\Big|_{x_1=0,y_1=0} \\
&= 2
\end{aligned}$$

$1 \le n \le 300$，$1 \le k \le 10^5$。

一道例题

给定 N 个点 M 条边的有向图 G，求出一个割，使得每条 $1 \to N$ 的路径都有**恰好**一条边被割掉。最小化割的权值。
$1 \leq N \leq 1000$，$1 \leq M \leq 2000$。

一道例题 Solution



「雅礼集训 2017 Day2」棋盘游戏

有一个 $n \times m$ 的棋盘，可能有障碍。后手先选择一个起始点，并放进去一个棋子，然后从先手开始，把棋子向相邻的，不是障碍，且没有经过的格子移动。不能操作者输，求出后手选择哪些位置的时候有必胜策略。

$1 \leq n, m \leq 100$。

另一道例题

给定两边都是 n 个节点，m 条从左往右的有向边的二分图 G，每条边均有一个流量限制。如下构造一族网络 $\{F_k\}$：

- 首先把图 G 复制 k 份，记作 $G_1, G_2, \cdots, G_k$，原先的边均保留；
- 对于所有 $1 \leq i < k$ 和 $1 \leq u \leq n$，从 G_i 右边第 u 个节点向 G_{i+1} 左边第 u 个节点连流量无限的有向边。
- 从源点向 G_1 左边部分每个节点连一条流量无限的有向边。
- 从 G_k 右边部分每个节点向汇点连一条流量无限的有向边。

求当 $k \to +\infty$ 时，该图的最大流。
$1 \leq n \leq 2000$，$1 \leq m \leq 4000$。

Thank You!