网络流基础

杭州学军中学 郎思轲

一些定义

一些定义

- 流网络 $G=(V,E)$是一个有向图

一些定义

- 流网络 $G=(V,E)$是一个有向图
- 容量函数 c：$V\times V\rightarrow \mathbf{R}$

$$\forall (u,v), c(u,v)\geq 0 \quad \forall (u,v)\notin E, c(u,v)=0$$

一些定义

- 流网络 $G=(V,E)$是一个有向图
- 容量函数 c：$V\times V\rightarrow\mathbf{R}$

$$\forall(u,v),c(u,v)\geq 0 \quad \forall(u,v)\notin E,c(u,v)=0$$

- 流量函数 f：$V\times V\rightarrow\mathbf{R}$

$$\forall(u,v),0\leq f(u,v)\leq c(u,v)$$

$$\forall u\in V-\{s,t\},\sum_{v\in V}f(v,u)=\sum_{v\in V}f(u,v)$$

一些定义

一些定义

- $| f | = \sum_{v \in V} f(s,v) - f(v,s)$

一些定义

- $|f| = \sum_{v \in V} f(s,v) - f(v,s)$
- 割 (S,T) 的容量

$$c(S,T) = \sum_{u \in S} \sum_{v \in T} c(u,v)$$

一些定义

- $|f| = \sum_{v \in V} f(s,v) - f(v,s)$
- 割 (S,T) 的容量

  $$c(S,T) = \sum_{u \in S} \sum_{v \in T} c(u,v)$$

- 残量网络 G_f 和增广路径

一些定义

- $|f| = \sum_{v \in V} f(s,v) - f(v,s)$
- 割 (S,T) 的容量

  $$c(S,T) = \sum_{u \in S} \sum_{v \in T} c(u,v)$$

- 残量网络 G_f 和增广路径
- 最大流和最小割

基本结论

基本结论

- 流 f 与其残量网络 G_f 上的任意一个流的和为原网络的合法流，且流量为两流量之和

基本结论

- 流 f 与其残量网络 G_f 上的任意一个流的和为原网络的合法流，且流量为两流量之和
- 同一流网络上流 f_1 与流 f_2 的差为 G_{f_2} 上的合法流,且流量为两流量之差

基本结论

- 流 f 与其残量网络 G_f 上的任意一个流的和为原网络的合法流，且流量为两流量之和
- 同一流网络上流 f_1 与流 f_2 的差为 G_{f_2} 上的合法流,且流量为两流量之差
- 若某残量网络上存在增广路径，则该残量网络存在非零合法流

基本结论

- 流 f 与其残量网络 G_f 上的任意一个流的和为原网络的合法流，且流量为两流量之和
- 同一流网络上流 f_1 与流 f_2 的差为 G_{f_2} 上的合法流,且流量为两流量之差
- 若某残量网络上存在增广路径，则该残量网络存在非零合法流
- 对于同一个网络的一个流和一个割，流的流量总小于等于割的容量

最大流最小割定理

最大流最小割定理

- 设流 f 为流网络 G 的一个流，则以下命题等价：

  1) f 为流网络 G 的最大流

  2) 残量网络 G_f 没有增广路径

  3) 存在一个割 (S,T) ,使 $|f| = c(S,T)$

最大流最小割定理

- 设流 f 为流网络 G 的一个流，则以下命题等价：

  1) f 为流网络 G 的最大流

  2) 残量网络 G_f 没有增广路径

  3) 存在一个割 (S,T) ,使 $|f| = c(S,T)$

1)→2) 反证法

最大流最小割定理

- 设流 f 为流网络 G 的一个流，则以下命题等价：

  1) f 为流网络 G 的最大流

  2) 残量网络 G_f 没有增广路径

  3) 存在一个割 (S,T) ,使 $|f| = c(S,T)$

1)→2) 反证法

2)→3) 将残量网络上源点通过非零边能到达的点设为 S，其余设为 T 即可满足3)的条件

最大流最小割定理

- 设流 f 为流网络 G 的一个流，则以下命题等价：

  1) f 为流网络 G 的最大流

  2) 残量网络 G_f 没有增广路径

  3) 存在一个割 (S,T) ,使 $|f| = c(S,T)$

1)→2) 反证法

2)→3) 将残量网络上源点通过非零边能到达的点设为 S，其余设为 T 即可满足3)的条件

3)→1) 根据基本结论显然

Dinic 算法及复杂度分析

Dinic 算法及复杂度分析

- 在残量网络上根据到源点的最短距离分层，在分层图上增广，增广完成后重新构建分层图，直到源汇不连通

Dinic 算法及复杂度分析

- 在残量网络上根据到源点的最短距离分层，在分层图上增广，增广完成后重新构建分层图，直到源汇不连通
- 注意加当前弧优化

Dinic 算法及复杂度分析

- 在残量网络上根据到源点的最短距离分层，在分层图上增广，增广完成后重新构建分层图，直到源汇不连通
- 注意加当前弧优化
- 每次构建分层图最短路严格递增

Dinic 算法及复杂度分析

- 在残量网络上根据到源点的最短距离分层，在分层图上增广，增广完成后重新构建分层图，直到源汇不连通
- 注意加当前弧优化
- 每次构建分层图最短路严格递增
- 朴素实现：
  分层图构建次数 $O(n)$,找到一条增广路会使一条边在分层图中消失，总复杂度 $O(n^2m)$

Dinic 算法及复杂度分析

Dinic 算法及复杂度分析

- 使用 LCT 优化在分层图上增广过程：

Dinic 算法及复杂度分析

- 使用 LCT 优化在分层图上增广过程：

初始时每个点为独立的一棵树，每次找到源点所在的树根(到源点距离最大的点)，若该点还有未被删除的出边，则把这条边设为树边，否则将其所有入边删除；若源和汇在一棵树内，找出它们路径上最小值，进行一次增广(路径减这个值)，并将最小值所在位置的边删除。当源点为独立的一棵树且没有出边时算法结束

Dinic 算法及复杂度分析

- 使用 LCT 优化在分层图上增广过程：

初始时每个点为独立的一棵树，每次找到源点所在的树根(到源点距离最大的点)，若该点还有未被删除的出边，则把这条边设为树边，否则将其所有入边删除；若源和汇在一棵树内，找出它们路径上最小值，进行一次增广(路径减这个值)，并将最小值所在位置的边删除。当源点为独立的一棵树且没有出边时算法结束

$O(nm\log m)$

Dinic 算法及复杂度分析

- 使用 LCT 优化在分层图上增广过程：

初始时每个点为独立的一棵树，每次找到源点所在的树根(到源点距离最大的点)，若该点还有未被删除的出边，则把这条边设为树边，否则将其所有入边删除；若源和汇在一棵树内，找出它们路径上最小值，进行一次增广(路径减这个值)，并将最小值所在位置的边删除。当源点为独立的一棵树且没有出边时算法结束

$O(nm\log m)$

- ~~显然并没有什么用~~

Dinic 算法及复杂度分析

Dinic 算法及复杂度分析

- 单位容量网络：每条边容量均为1

Dinic 算法及复杂度分析

- 单位容量网络：每条边容量均为1
- 每次增广 $O(m)$

Dinic 算法及复杂度分析

- 单位容量网络：每条边容量均为1
- 每次增广 $O(m)$
- 分层图构建次数为 $O(m^{1/2})$

Dinic 算法及复杂度分析

- 单位容量网络：每条边容量均为1
- 每次增广 $O(m)$
- 分层图构建次数为 $O(m^{1/2})$
- 若无重边，分层图构建次数为 $O(n^{2/3})$

Dinic 算法及复杂度分析

- 单位容量网络：每条边容量均为1
- 每次增广 $O(m)$
- 分层图构建次数为 $O(m^{1/2})$
- 若无重边，分层图构建次数为 $O(n^{2/3})$
- $O(\min(n^{2/3}, m^{1/2}) \cdot m)$

Dinic 算法及复杂度分析

Dinic 算法及复杂度分析

- 单位网络：除源和汇以外其他点只有一条入边或只有一条出边的单位容量网络

Dinic 算法及复杂度分析

- 单位网络：除源和汇以外其他点只有一条入边或只有一条出边的单位容量网络
- 每次增广 $O(m)$

Dinic 算法及复杂度分析

- 单位网络：除源和汇以外其他点只有一条入边或只有一条出边的单位容量网络
- 每次增广 $O(m)$
- 残量网络仍为单位网络

Dinic 算法及复杂度分析

- 单位网络：除源和汇以外其他点只有一条入边或只有一条出边的单位容量网络
- 每次增广 $O(m)$
- 残量网络仍为单位网络
- 分层图构建次数为 $O(n^{1/2})$

Dinic 算法及复杂度分析

- 单位网络：除源和汇以外其他点只有一条入边或只有一条出边的单位容量网络
- 每次增广 $O(m)$
- 残量网络仍为单位网络
- 分层图构建次数为 $O(n^{1/2})$
- $O(n^{1/2} \cdot m)$

Dinic 算法及复杂度分析

- 单位网络：除源和汇以外其他点只有一条入边或只有一条出边的单位容量网络
- 每次增广 $O(m)$
- 残量网络仍为单位网络
- 分层图构建次数为 $O(n^{1/2})$
- $O(n^{1/2} \cdot m)$
- 二分图匹配

费用流相关

费用流相关

- 费用函数 $cost$：$V \times V \rightarrow \mathbf{R}$

费用流相关

- 费用函数 $cost$：$V \times V \rightarrow \mathbf{R}$
- 一个流的费用为 $\sum_{(u,v)} f(u,v) \cdot cost(u,v)$

费用流相关

- 费用函数 $cost$：$V \times V \rightarrow \mathbf{R}$
- 一个流的费用为 $\sum_{(u,v)} f(u,v) \cdot cost(u,v)$
- 若将费用作为边权，必须使得图中没有负环

费用流相关

- 费用函数 $cost$：$V \times V \to \mathbf{R}$
- 一个流的费用为 $\sum_{(u,v)} f(u,v) \cdot cost(u,v)$
- 若将费用作为边权，必须使得图中没有负环
- 在残量网络中反向边的费用为原费用的相反数

费用流相关

- 费用函数 $cost$：$V \times V \rightarrow \mathbf{R}$
- 一个流的费用为 $\sum_{(u,v)} f(u,v) \cdot cost(u,v)$

- 若将费用作为边权，必须使得图中没有负环
- 在残量网络中反向边的费用为原费用的相反数
- 最小费用流和最小费用最大流

基本结论

基本结论

- 若一个流的残量网络上不存在费用的负环，那么这个流是当前流量下的最小费用流

基本结论

- 若一个流的残量网络上不存在费用的负环，那么这个流是当前流量下的最小费用流
- 将一个不含负环的残量网络沿带权最短路进行增广后仍然不含负环

基本结论

- 若一个流的残量网络上不存在费用的负环，那么这个流是当前流量下的最小费用流
- 将一个不含负环的残量网络沿带权最短路进行增广后仍然不含负环
- 将一个不含负环的残量网络沿带权最短路进行增广后每个点到源点的带权最短路不降

费用流最短增广算法

费用流最短增广算法

- 不断在残量网络上沿带权最短路增广

费用流最短增广算法

- 不断在残量网络上沿带权最短路增广
- 最小费用流：最短路为正或不能增广时结束

费用流最短增广算法

- 不断在残量网络上沿带权最短路增广
- 最小费用流：最短路为正或不能增广时结束
- 最小费用最大流：不能增广时结束

Primal-Dual原始对偶算法

Primal-Dual原始对偶算法

- 初始运行一遍 Bellman-Ford Algorithm
  求出初始每个点到源点最短路

Primal-Dual原始对偶算法

- 初始运行一遍 Bellman-Ford Algorithm 求出初始每个点到源点最短路
- 定义势函数$h_u = dist(s,u)$,对于任意一条边(u,v)，将其边权修改为$cost(u,v)+h_u-h_v$，显然边权非负，可以运行Dijkstra's Algorithm 求出新的单源最短路并更新势函数

Primal-Dual原始对偶算法

- 初始运行一遍 Bellman-Ford Algorithm 求出初始每个点到源点最短路
- 定义势函数$h_u = dist(s,u)$,对于任意一条边(u,v)，将其边权修改为$cost(u,v)+h_u-h_v$，显然边权非负，可以运行Dijkstra's Algorithm 求出新的单源最短路并更新势函数
- 沿最短路增广，可以发现 $h_u+cost(u,v)\geq h_v$ 仍然成立

上下界网络流

- 可行循环流：

上下界网络流

- 可行循环流：

  建立附加源、汇 s, t, 对于一条边(u, v), 容量限制$[l, r]$为在新图中添加：

  $$(s,v,l),(u,t,l),(u,v,r-l)$$

  运行最大流，若流量不为$\sum l$ 则无解，否则可根据残量网络还原出可行解

上下界网络流

- 可行循环流：
  建立附加源、汇 s, t, 对于一条边(u, v),
  容量限制$[l, r]$为在新图中添加：

  $$(s,v,l),(u,t,l),(u,v,r-l)$$

  运行最大流，若流量不为$\sum l$ 则无解，
  否则可根据残量网络还原出可行解

- 上下界最大流：
  先加边(t,s,∞)找出一个可行流，然后删去附加源汇和这条边，运行最大流

上下界网络流

上下界网络流

- 上下界最小流：

上下界网络流

- 上下界最小流：
  先加边(t,s,∞)找出一个可行流，然后删去附加源汇和这条边，交换源汇运行最大流

上下界网络流

- 上下界最小流：
  先加边(t,s,∞)找出一个可行流，然后删去附加源汇和这条边，交换源汇运行最大流

- 最小费用循环流：

上下界网络流

- 上下界最小流：
  先加边(t,s,∞)找出一个可行流，然后删去附加源汇和这条边，交换源汇运行最大流

- 最小费用循环流：
  给可行循环流的做法中加上费用即可

上下界网络流

- 上下界最小流：
  先加边(t,s,∞)找出一个可行流，然后删去附加源汇和这条边，交换源汇运行最大流

- 最小费用循环流：
  给可行循环流的做法中加上费用即可

- 最小费用最大流：

上下界网络流

- 上下界最小流：
  先加边(t,s,∞)找出一个可行流，然后删去附加源汇和这条边，交换源汇运行最大流

- 最小费用循环流：
  给可行循环流的做法中加上费用即可

- 最小费用最大流：
  先计算最小费用可行流，然后增广

一些经典模型

一些经典模型

- 最大权闭合子图：

  给定有向图 (V, E) , 求一个V 的子集 A，使得不存在$u \in A, v \in V - A$的边 (u,v) ,且 A 中点权和最大

一些经典模型

- 最大权闭合子图：

  给定有向图 (V, E) , 求一个V 的子集 A，使得不存在$u \in A, v \in V - A$的边 (u,v) ,且 A 中点权和最大

- 最大密度子图：

  给定正点权，正边权的有向图 (V, E) , 求一个子图使得边权与点权的比值尽量大

一些经典模型

- 最大权闭合子图：

  给定有向图 (V, E) , 求一个V 的子集 A，使得不存在$u \in A, v \in V - A$的边 (u,v) ,且 A 中点权和最大

- 最大密度子图：

  给定正点权，正边权的有向图 (V, E) , 求一个子图使得边权与点权的比值尽量大

- 区间 k 覆盖问题：

  给定 n 个带权区间，求一个区间的子集，使得任意一个点最多被 k 个区间覆盖，且在子集中的区间权值和最大

一些经典模型

一些经典模型

- 混合图欧拉路径问题：
  给定一张混合图，求一条欧拉路径

一些经典模型

- 混合图欧拉路径问题：
  给定一张混合图，求一条欧拉路径
- 先判断图的连通性

一些经典模型

- 混合图欧拉路径问题：
  给定一张混合图，求一条欧拉路径
- 先判断图的连通性
- 将边随意定向，统计每个点的度数(入-出)
  判断是否合法以及是否为回路，若不是回路需要枚举起点

一些经典模型

- 混合图欧拉路径问题：
  给定一张混合图，求一条欧拉路径
- 先判断图的连通性
- 将边随意定向，统计每个点的度数(入-出)
  判断是否合法以及是否为回路，若不是回路需要枚举起点
- 若点 u 的度应上升$2x$,加边 (u,t,x)
  若点 u 的度应下降$2x$,加边 (s,u,x)
  若边(u,v)能反向 ,加边 $(v,u,1)$

一些经典模型

- 混合图欧拉路径问题：
  给定一张混合图，求一条欧拉路径
- 先判断图的连通性
- 将边随意定向，统计每个点的度数(入-出)
  判断是否合法以及是否为回路，若不是回路需要枚举起点
- 若点 u 的度应上升$2x$,加边 (u,t,x)
  若点 u 的度应下降$2x$,加边 (s,u,x)
  若边(u,v)能反向 ,加边 $(v,u,1)$
- 若满流，找到可行解

一些经典模型

- 混合图欧拉路径问题：
  给定一张混合图，求一条欧拉路径
- 先判断图的连通性
- 将边随意定向，统计每个点的度数(入-出)
  判断是否合法以及是否为回路，若不是回路需要枚举起点
- 若点 u 的度应上升$2x$,加边 (u,t,x)
  若点 u 的度应下降$2x$,加边 (s,u,x)
  若边(u,v)能反向 ,加边 $(v,u,1)$
- 若满流，找到可行解
- $O(\min(n^{2/3},m^{1/2})\cdot m)$

Airport

- 有 n 架飞机和 m 个停机坪，停机坪中有 a 个是“好的”，剩下 m-a 个是“坏的”
- 每架飞机有给定的不同的降落时间、起飞时间和将登机的乘客数量
- 每个停机坪每个时刻只能停一架飞机，在一个停机坪中起飞和降落不能同时进行
- 在两个时刻之间，一架飞机可以从一个停机坪移动到另一个停机坪，若飞机乘客数为 x，这将花费 $\lfloor xp \rfloor$ 的费用，其中 p 为确定的常数且 $0 < p < 1$
- 一架飞机在降落时刻可以降落在任意一个停机坪上并瞬间完成登机，若登机在“坏的”停机坪上进行，设飞机乘客数为 x，完成登机将花费 x 的费用，否则无需费用
- 最小化总费用
- $1 \le n \le 1000, 1 \le m \le 500, 1 \le x \le 10^9, 0 \le a \le m$

Airport

Airport

- 判断是否有解，若有解可认为“坏的”停机坪无限

Airport

- 判断是否有解，若有解可认为“坏的”停机坪无限
- 若飞机要移动，一定从“好的”停机坪移到“坏的”停机坪

Airport

- 判断是否有解，若有解可认为“坏的”停机坪无限
- 若飞机要移动，一定从“好的”停机坪移到“坏的”停机坪
- 先假设所有飞机都停在“坏的”停机坪，费用为所有飞机乘客数之和，若飞机降落时刻为 l，起飞时刻为 r，乘客为 x，则可选择:
  1) 时刻 $[l, r]$占用一个“好的”停机坪，收益 x
  2) 时刻 l占用一个“好的”停机坪，收益 $x - \lfloor px \rfloor$
  3) 不占用“好的”停机坪，无收益

Airport

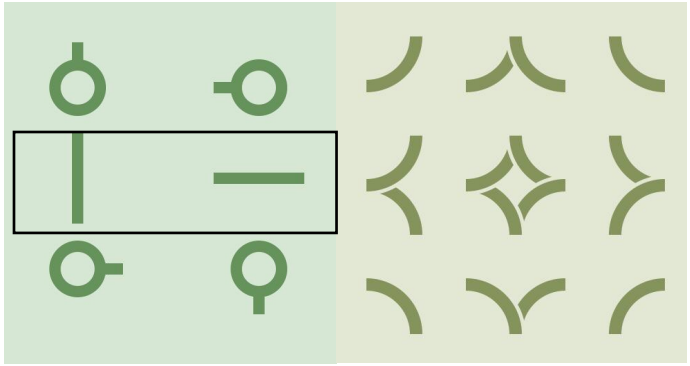
- 判断是否有解，若有解可认为“坏的”停机坪无限
- 若飞机要移动，一定从“好的”停机坪移到“坏的”停机坪
- 先假设所有飞机都停在“坏的”停机坪，费用为所有飞机乘客数之和，若飞机降落时刻为 l，起飞时刻为 r，乘客为 x，则可选择:
  1) 时刻 $[l, r]$占用一个“好的”停机坪，收益 x
  2) 时刻 l占用一个“好的”停机坪，收益 $x - \lfloor px \rfloor$
  3) 不占用“好的”停机坪，无收益
- 类似区间 k 覆盖问题，但每个飞机需新建点使一架飞机只能选择一个决策

Airport

- 判断是否有解，若有解可认为“坏的”停机坪无限
- 若飞机要移动，一定从“好的”停机坪移到“坏的”停机坪
- 先假设所有飞机都停在“坏的”停机坪，费用为所有飞机乘客数之和，若飞机降落时刻为 l，起飞时刻为 r，乘客为 x，则可选择:
  1) 时刻 $[l, r]$占用一个“好的”停机坪，收益 x
  2) 时刻 l占用一个“好的”停机坪，收益 $x - \lfloor px \rfloor$
  3) 不占用“好的”停机坪，无收益
- 类似区间 k 覆盖问题，但每个飞机需新建点使一架飞机只能选择一个决策
- 第一次最短路使用dp计算，$O(an \log n)$

无限之环

- 有 一个$n \times m$ 的网格状棋盘，有些方格上有水管，水管可能在方格某些方向的边界的中点有接口，如果两个相邻方格的公共边界的中点都有接头，那么可以看作这两个接头互相连接。水管有以下15种形状：



- 你可以选定一个含有非直线型水管的方格，将其中的水管绕方格中心顺时针或逆时针旋转90度。
- 给出一个初始局面，请问最少进行多少次操作可以使棋盘上不存在漏水的地方。
- $nm \leq 2000$

无限之环

无限之环

- 考虑用1的流量代表一个吻合了的接口，那么如果最大吻合接口对数为(总接口数/2)那么是一个合法的方案

无限之环

- 考虑用1的流量代表一个吻合了的接口，那么如果最大吻合接口对数为(总接口数/2)那么是一个合法的方案
- 由于是网格图，考虑黑白染色，并将一个网格拆为4个点，每个点代表一个接口，源点向黑格子所有当前存在的接口连(1,0)的边，白格子所有当前存在的接口向汇点连(1,0)的边，相邻的接口连(1,0)的边

无限之环

- 考虑用1的流量代表一个吻合了的接口，那么如果最大吻合接口对数为(总接口数/2)那么是一个合法的方案
- 由于是网格图，考虑黑白染色，并将一个网格拆为4个点，每个点代表一个接口，源点向黑格子所有当前存在的接口连(1,0)的边，白格子所有当前存在的接口向汇点连(1,0)的边，相邻的接口连(1,0)的边
- 现在我们可以通过旋转改变接口，可以发现有3种本质不同的调整方案：

无限之环

- 考虑用1的流量代表一个吻合了的接口，那么如果最大吻合接口对数为(总接口数/2)那么是一个合法的方案
- 由于是网格图，考虑黑白染色，并将一个网格拆为4个点，每个点代表一个接口，源点向黑格子所有当前存在的接口连(1,0)的边，白格子所有当前存在的接口向汇点连(1,0)的边，相邻的接口连(1,0)的边
- 现在我们可以通过旋转改变接口，可以发现有3种本质不同的调整方案：

(1,2,1,1)
(1,3,1,2)
(1,4,1,1)

无限之环

- 考虑用1的流量代表一个吻合了的接口，那么如果最大吻合接口对数为(总接口数/2)那么是一个合法的方案
- 由于是网格图，考虑黑白染色，并将一个网格拆为4个点，每个点代表一个接口，源点向黑格子所有当前存在的接口连(1,0)的边，白格子所有当前存在的接口向汇点连(1,0)的边，相邻的接口连(1,0)的边
- 现在我们可以通过旋转改变接口，可以发现有3种本质不同的调整方案：

(1,2,1,1)
(1,3,1,2)
(1,4,1,1)

(1,3,1,1)
(2,4,1,1)

无限之环

- 考虑用1的流量代表一个吻合了的接口，那么如果最大吻合接口对数为(总接口数/2)那么是一个合法的方案
- 由于是网格图，考虑黑白染色，并将一个网格拆为4个点，每个点代表一个接口，源点向黑格子所有当前存在的接口连(1,0)的边，白格子所有当前存在的接口向汇点连(1,0)的边，相邻的接口连(1,0)的边
- 现在我们可以通过旋转改变接口，可以发现有3种本质不同的调整方案：

(1,2,1,1)
(1,3,1,2)
(1,4,1,1)

(1,3,1,1)
(2,4,1,1)

(1,4,1,1)
(2,4,1,2)
(3,4,1,1)

无限之环

- 考虑用1的流量代表一个吻合了的接口，那么如果最大吻合接口对数为(总接口数/2)那么是一个合法的方案
- 由于是网格图，考虑黑白染色，并将一个网格拆为4个点，每个点代表一个接口，源点向黑格子所有当前存在的接口连(1,0)的边，白格子所有当前存在的接口向汇点连(1,0)的边，相邻的接口连(1,0)的边
- 现在我们可以通过旋转改变接口，可以发现有3种本质不同的调整方案：

(1,2,1,1)
(1,3,1,2)
(1,4,1,1)

(1,3,1,1)
(2,4,1,1)

(1,4,1,1)
(2,4,1,2)
(3,4,1,1)

- 运行最小费用最大流，若最大流为(总接口数/2)则答案为费用值，否则无解

无限之环

- 考虑用1的流量代表一个吻合了的接口，那么如果最大吻合接口对数为(总接口数/2)那么是一个合法的方案
- 由于是网格图，考虑黑白染色，并将一个网格拆为4个点，每个点代表一个接口，源点向黑格子所有当前存在的接口连(1,0)的边，白格子所有当前存在的接口向汇点连(1,0)的边，相邻的接口连(1,0)的边
- 现在我们可以通过旋转改变接口，可以发现有3种本质不同的调整方案：

(1,2,1,1)
(1,3,1,2)
(1,4,1,1)

(1,3,1,1)
(2,4,1,1)

(1,4,1,1)
(2,4,1,2)
(3,4,1,1)

- 运行最小费用最大流，若最大流为(总接口数/2)则答案为费用值，否则无解
- $O(Mcmf(nm,nm))$

平衡

- 你有 n 件事情要做，每件事情有一个原定需要时间，你需要修改这些时间使它们满足一些要求
- 有两种限制共 m 条：
  1) 事情 i 和 j 需要花费相同的时间
  2) 事情 i 需要花费的时间不多于事情 j
- 修改一件事情的时间花费的代价是原定时间和现在时间的差的绝对值
- 最小化代价
- Task1：$n, m \leq 50$
- Task2：$n \leq 500, m \leq 100000$

平衡

- 你有 n 件事情要做，每件事情有一个原定需要时间，你需要修改这些时间使它们满足一些要求
- 有两种限制共 m 条：
  1) 事情 i 和 j 需要花费相同的时间
  2) 事情 i 需要花费的时间不多于事情 j
- 修改一件事情的时间花费的代价是原定时间和现在时间的差的绝对值
- 最小化代价
- Task1：$n, m \leq 50$
- Task2：$n \leq 500, m \leq 100000$
- ~~对网络流的速度要有信仰~~

平衡

平衡

- Task1：

显然最终每个时间的取值都是原定时间中的某些值；可以对每个事件建一排点，每个点向左，源点向最左端，最右端向汇点连 *inf* 边，每个点向右连相应代价；第二种限制可以将事情 *i* 对应的一排点的每个点向事情 *j* 对应的一排点的同位置的点连 *inf* 边，第一种限制可以转化为第二种限制

点数$O(n^2)$ 边数 $O(n\cdot(n+m))$

平衡

- Task1：

显然最终每个时间的取值都是原定时间中的某些值；可以对每个事件建一排点，每个点向左，源点向最左端，最右端向汇点连 inf 边，每个点向右连相应代价；第二种限制可以将事情 i 对应的一排点的每个点向事情 j 对应的一排点的同位置的点连 inf 边，第一种限制可以转化为第二种限制

点数$O(n^2)$ 边数 $O(n\cdot(n+m))$

- Task2：

考虑一个值mid，对于一个点 u ，设其代价函数为 f_u ，则为其赋权为 $f_u(mid+1)-f_u(mid)$

若 u 的最终值要求小于等于 v 的最终值，那么连有向边(u,v),考虑这张图中的最小权闭合子图，一定存在一个最优解使得该子图中点最终时间均大于 mid ，而非子图中则小于等于 mid

平衡

按此结论进行分治，最终可确定每个点的答案；

每层分治点数规模的和为 $O(n)$，边数规模的和为 $O(m)$，层数规模为$O(\log n)$

$$O(Maxflow(n,m)\cdot\log n)$$

Parametric Circulation

- 你有一个上下界循环流，其中的每一条边的上界和下界都是关于变量 t 的线性函数，即对于一条边 e：

$$l_e(t) = a_e t + b_e$$
$$r_e(t) = c_e t + d_e$$

- 现在 t 是一个[0,1]中的随机变量(每条边的 t 相同)，保证 t 取[0,1]中的任何值时 $0 \le l_e(t) \le r_e(t) \le 10^4$，求该网络存在可行循环流的概率，误差不超过 10^{-6}

- $1 \le n \le 1000, 1 \le m \le 2000, -10^4 \le a_e, c_e \le 10^4$

Parametric Circulation

Parametric Circulation

- 将存在循环流转化为最大流等于源点出边的容量和

Parametric Circulation

- 将存在循环流转化为最大流等于源点出边的容量和
- 考虑某一个割的容量，显然它是一个形如 $at+b$ 的关于 t 的线性函数，那么这个流网络的最大流值显然是一个凸函数

Parametric Circulation

- 将存在循环流转化为最大流等于源点出边的容量和
- 考虑某一个割的容量，显然它是一个形如 $at+b$ 的关于 t 的线性函数，那么这个流网络的最大流值显然是一个凸函数
- 由于源点出边的容量和是一个固定的形如 $at+b$的函数，那么记函数 $g(t)$ 表示 t 时刻的容量和减最大流，显然 g 的导函数单调不降

Parametric Circulation

- 将存在循环流转化为最大流等于源点出边的容量和
- 考虑某一个割的容量，显然它是一个形如 $at+b$ 的关于 t 的线性函数，那么这个流网络的最大流值显然是一个凸函数
- 由于源点出边的容量和是一个固定的形如 $at+b$的函数，那么记函数 $g(t)$ 表示 t 时刻的容量和减最大流，显然 g 的导函数单调不降
- 因此合法的 t 值一定在一段区间上，我们只需计算这一段的长度

Parametric Circulation

- 将存在循环流转化为最大流等于源点出边的容量和
- 考虑某一个割的容量，显然它是一个形如 $at+b$ 的关于 t 的线性函数，那么这个流网络的最大流值显然是一个凸函数
- 由于源点出边的容量和是一个固定的形如 $at+b$的函数，那么记函数 $g(t)$ 表示 t 时刻的容量和减最大流，显然 g 的导函数单调不降
- 因此合法的 t 值一定在一段区间上，我们只需计算这一段的长度
- 只需要找出一个合法的 t 值就可以两边二分找出端点，因此我们可以根据斜率单调性二分出极值点，若为0则找到合法点，否则没有合法 t 值，判断一个点的斜率可以运行最大流构造一个最小割的划分。

Parametric Circulation

- 将存在循环流转化为最大流等于源点出边的容量和
- 考虑某一个割的容量，显然它是一个形如 $at+b$ 的关于 t 的线性函数，那么这个流网络的最大流值显然是一个凸函数
- 由于源点出边的容量和是一个固定的形如 $at+b$的函数，那么记函数 $g(t)$ 表示 t 时刻的容量和减最大流，显然 g 的导函数单调不降
- 因此合法的 t 值一定在一段区间上，我们只需计算这一段的长度
- 只需要找出一个合法的 t 值就可以两边二分找出端点，因此我们可以根据斜率单调性二分出极值点，若为0则找到合法点，否则没有合法 t 值，判断一个点的斜率可以运行最大流构造一个最小割的划分。
- 实现需要注意精度

Parametric Circulation

- 将存在循环流转化为最大流等于源点出边的容量和
- 考虑某一个割的容量，显然它是一个形如 $at+b$ 的关于 t 的线性函数，那么这个流网络的最大流值显然是一个凸函数
- 由于源点出边的容量和是一个固定的形如 $at+b$的函数，那么记函数 $g(t)$ 表示 t 时刻的容量和减最大流，显然 g 的导函数单调不降
- 因此合法的 t 值一定在一段区间上，我们只需计算这一段的长度
- 只需要找出一个合法的 t 值就可以两边二分找出端点，因此我们可以根据斜率单调性二分出极值点，若为0则找到合法点，否则没有合法 t 值，判断一个点的斜率可以运行最大流构造一个最小割的划分。
- 实现需要注意精度
- $O(Maxflow(n,m)\cdot\log V)$

Thanks!