图论专题之
生成树

清华大学 唐文斌
tangwb06@gmail.com

定义：生成树

- 无向图G=(V,E)
- G的一个生成树T是G的一个子图(树)
  - 包含G的所有节点：$V' = V$
  - 包含G的部分$E' \in E$

- 性质：包含所有点，无环
  - 生成树：最大的无环边集
  - 生成树：最小的连接所有点的边集
  - 包含所有的桥(割边)

更多定义...

- 树的直径:
  - 树的直径定义为树上最远两点的距离
- 树的中心
  - 在树上选择一个点, 离其他所有点最远点距离最小
    - Q: 可能有多少个?
  - 性质: 一定是直径路径的中心点

最小生成树(MST)

- 最小生成树(Minimal Spanning Tree)
  - 带权无向图
  - 生成树的权定义为所有树边的权值之和
- 最小树形图(Minimum Arborescence)
  - 带权有向图
  - 从某一个节点出发，可以到达所有节点
- 生成森林
  - 原图不连通，每一个连通块的生成树集合

Prim

- Prim算法:
  - 与Dijkstra相近
  - 选任一点为根
  - 找不在树上且离树最近的点加入生成树
  - 时间复杂度O(*mlogn*) //优先队列优化

Kruskal

- Kruskal算法:
  - 一开始所有的点为独立连通块
  - 按边权从小到大检查每一条边
    - 如果这条边连接了两个不同的连通块(即不形成环)
    - 则将这条边加入, 并将两个连通块合并
  - 使用并查集进行连通块判定
  - O($mlogm + m\alpha(m)$)

Borůvka's algorithm

- Borůvka's algorithm
  - 类似Kruskal的多路增广版
  - 一开始所有点为独立连通块
  - 每一次:
    - 每一个连通块寻找一条往外连接的最小权值边
    - 将所有的这些边都加入
      - 边权两两不同时可以保证不形成环
      - 若边权有相同: (边权, 点标号)一起判最小
  - 合并次数: 最坏情况*logn*次
  - 时间复杂度: 最坏O((m+n) *logn*), 随机图 O(n+m)

By TANG Wenbin

最小瓶颈生成树
(Minimum Bottleneck Spanning Tree)

- 一个无向图中权重最大的边最小
- 算法一： Kruskal的最后一条边就是瓶颈
- 定理: 任意 MST 一定是 MBST.
  - Why?
  - 所以任何MST的算法均可.
  - 注意MBST并不一定是MST
- 存在更好算法?

最小瓶颈生成树
(Minimum Bottleneck Spanning Tree)

- 线性算法

- 类似 快排分治 查找第k小数
  - 按照边权的集合，选择当前的瓶颈值V
  - 寻找所有权值不超过V的边，构成子图
    - 若子图连通，V是答案的上界，继续在权值较小的部分寻找
    - 若子图不连通，按当前连通性进行缩点，在权值较大部分寻找

- 时间复杂度：$T(m) = O(m) + T(m/2)$
- 时间复杂度$O(n+m)$

扩展：最小瓶颈路径查询

- Q个query
  - 每次查询a, b两个点
  - 输出$a \sim b$点之间的最小瓶颈路径的瓶颈值
- 算法：
  - 先求生成树
  - 在生成树上做RMQ维护
- 再扩展：若加入动态修改边权？
  - 动态树

次小生成树

- 求图的次小生成树

- 扩展：
  - 严格次小生成树

次小生成树

- 先求出最小生成树MST
- 然后枚举不在MST上的边(u,v)，若将(u,v)替换掉MST上节点u与节点v之间权值最大的边，那么得到的生成树的权值为
  - $w(MST)+w(u,v)-maxw(u,v)$
- 取最小值即得到次小生成树。

- 用$F[i,j]$表示由节点i向上2^j条边中边权的最大值，那么查询两点之间边权最大值可以在$O(\log n)$时间内解决。

By TANG Wenbin

严格次小生成树

- 同最小生成树做法，但有略微不同。
- 先求出最小生成树MST
- 然后枚举不在MST上的边(u,v)，若将(u,v)替换掉MST上节点u与节点v之间权值最大的边，若这条边的权值与$w(u,v)$相同，那么替换后的树不可能成为严格次小生成树。所以我们要替换节点u与节点v之间边权严格次大的边，这样得到的树才有可能是严格次小生成树。

最小比率生成树

- 给定无向图G
  - 每条边e包含权值 a_e, b_e

- 求生成树
  - 最小化树中的权值之和比率
  - 即 minimize: sum(a) / sum(b)

最小比率生成树

- 二分答案
  - 求解 → 判定
  - 判定是否存在比率不超过x的生成树

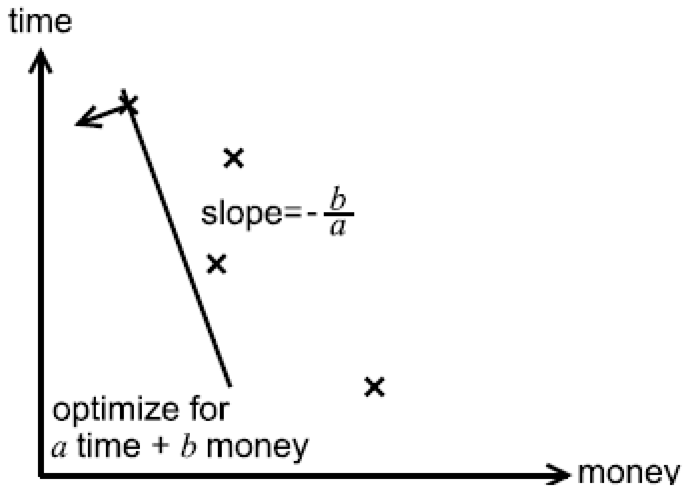
  - $Sum_e(a_e) / Sum_e(b_e) \le x$
  - $Sum_e(a_e) \le x * Sum_e(b_e)$
  - $Sum_e(a_e - x * b_e) \le 0$

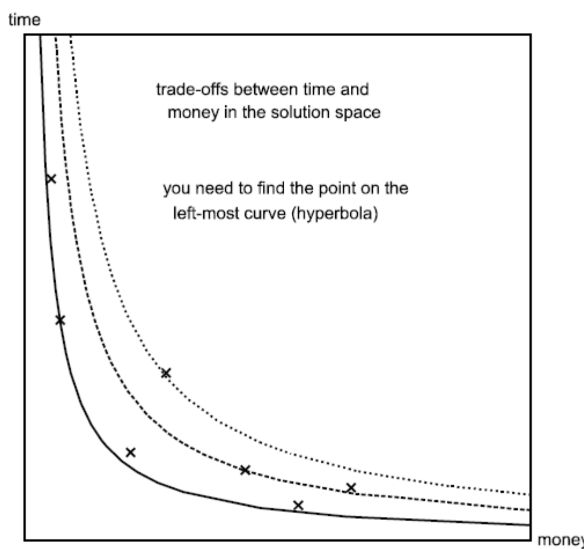
  - 判定方法: 以$a_e - x * b_e$为权值求最小生成树
  - 时间复杂度： MST * $O(logW)$

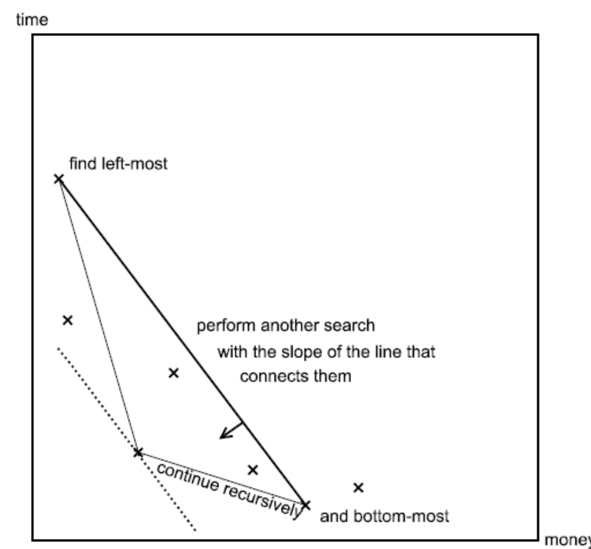
最小乘积生成树

- 给定无向图G
  - 每条边e包含权值 a_e, b_e
- 求生成树
  - 最小化树中的a, b权值之和乘积
  - 即 minimize: sum(a) * sum(b)
- 试题：[Balkan 2011 TimeIsMoney]

最小乘积生成树







By TANG Wenbin

最小乘积生成树

- 扩展:
  - 三个参数 a_e, b_e, C_e

单点度限制生成树

- 求满足单点度限制的最小生成树
- 即：其中某一个特定节点度数有限制
  - 例如： $deg(v_0) \le k$
- 扩展:
  - 多点度限制

单点度限制生成树

- 先不考虑v_0，求出$G - v_0$ 的最小生成森林
- 不同的连通块仅能通过v_0连接
  - 若块数超过k则误解
- 现在加入 v_0
  - 对于每一个连通块，选择一条连向v_0边权最小的边
- 现考虑逐步增大v_0的度数
  - 置换边： 置换一次$O(n)$
- 时间复杂度：$O(m\log n + kn)$

最小直径生成树

- 无权图
- 带权图

- 求一棵生成树，使得其直径最短

再引入一些概念

- 偏心距:
  - $Ecc(i) = \max_j \ d(i, j)$
  - 给定点的最远距离
- 图的直径:
  - $d(G) = \max_{i,j} d(i, j)$
- 图的半径
  - $r(G) = \min_i \ Ecc(i)$

无向图的中心

- 图的一般中心： 离图中最远点最近的点
  - 一个图可能有多个中心
- 枚举中心
  - BFS树(最短路径树)?

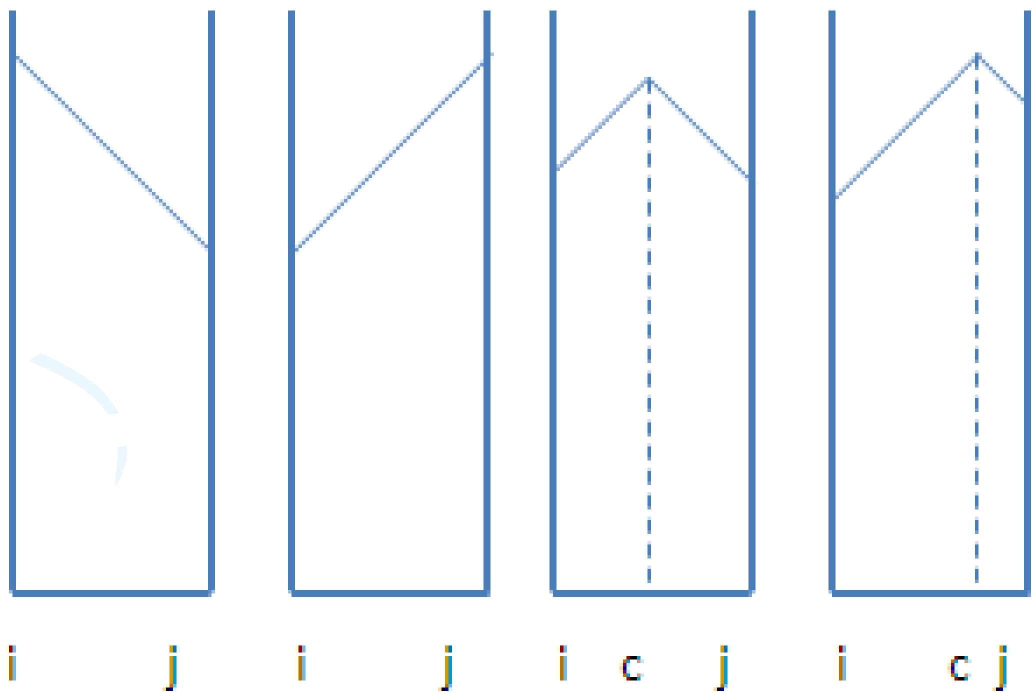
```
    0
   /|\
  / | 1
 /  |  \
4---3---2---5
```

图的绝对中心

- 绝对中心： 中心未必要在原图的点上
  - 可以在边上
  - 到最远点距离最小
- 最小直径生成树就是绝对中心的最短路树
  - 偏心距最小 → 直径最小
- 无权图：枚举绝对中心☺

带权图的MDST

- 求绝对中心
- 带区间的最短路算法



最小直径最小生成树

- ☹

双连通图的固定中心生成树

- 给定一个双连通图和一个顶点*s*，求一棵以*s*为中心的生成树。

- 双连通：
  - 连通图，且图中没有割点
  - 即，删除任意单个节点都不会导致图不连通

双连通图的固定中心生成树

- 算法：
  - 设t为s的任意一个相邻顶点。找到一组标号D，满足所有顶点被不重复地标为$1...n$，并且$D(s)=1,D(t)=n$。对于每个顶点$v(v \neq s,t)$，都存在两个与v相邻的顶点u和w，满足$D(u)<D(v)<D(w)$。

双连通图的固定中心生成树

- 将无向边(u,v)定向为从标号小的连向标号大的。求出新图中的以s为起点的BFS树T。
- 将t从T中删除作为T'的根，并将所有边(v,t)(定向后的)加入队列Q。
- 重复以下步骤，直到T的高度和T'的高度差不超过1。
  - 取出队首节点(v,w)。
  - 若v为T的叶节点：将其从T中删除，加入T'中作为w的子节点，并将所有边(u,v)加入队列Q。
- 最后将T和T'用边(s,t)连接得到以s为中心的生成树。

双连通图的固定中心生成树

- 如何找到一组标号D。
  - 先从t开始DFS，求出每个节点v的$Low(v)$。
  - $COUNT$=1。
  - 将t和s依次压入栈S，并将s,t和(s,t)标记为已访问。
  - 取出栈顶元素v，若所有v的相邻边都被访问，则$D(v)=COUNT$++。
  - 否则找到Path=$vv_1v_2...v_kw$，满足v和w是已访问的，其余点和边均是未访问的。然后依次将$v_k,v_{k-1},...,v_1,v$压入栈S。

双连通图的固定中心生成树

- 寻找Path
  - Case 1: 若存在一条未访问的返祖边(v,w)
    - Path=vw
  - Case 2: 若存在一条未访问的树边(v,w)
    - 令Path=$vww_1w_2 \ldots w_k$，其中$w_k=Low(w)$代表点。
  - Case 3: 若存在一条未访问的返祖边(w,v)
    - 令Path=$vww_1w_2 \ldots w_k$，其中w_k为一个已访问的点。

  - 找到Path后将Path上所有节点和边标记为已访问。

Most Vital Edge on Spanning Tree

- 删除一条边可能导致图的最小生成树变大
- Most Vital Edge:
  - 删掉哪条边使得图的最小生成树变大得最多

Most Vital Edge on Spanning Tree

- 基于可并堆合并
- 黑板☺

生成树的计数

- 给定无权图，求生成树的个数

- 扩展：
  - 给定带权图，求最小生成树的个数

生成树计数

- 无权图的生成树计数
  - 行列式：列出图G的Kirchhoff矩阵C：若$i=j$，则$C_{ij}=-degree(v_i)$；若$i\neq j$，则C_{ij}为v_i与v_j之间的边的个数。
  - 然后去掉C的任意第r行第r列得到的新矩阵C_r，C_r的行列式的绝对值即为生成树的个数。

生成树计数

- 带权图的最小生成树计数
  - 性质：在求最小生成树的过程中，若我们只用边权小于x的边，我们得到的是森林。由于选择的边不同，会得到不同的森林，但是森林的连通性是相同的。
  - 然后我们将森林中的每棵树缩成一个点，然后对于权值为x的边，若这条边连接了两棵树，则将对应点相连。设得到的图为G_x。
  - 将图G_x每个连通块分开处理，用无权图的生成树计数方法求出方案数。然后相乘即可。

Thank you!

最小修改边权

- 给定一个带权图和图中的一棵生成树
- 可以修改边权
  - 目标：使得给定的生成树是该图的MST
  - 要求修改边权的总量最小
    - 可以增加
    - 可以减少