单源最短路径问题

目录

问题的导入

例

Professor Wang wishes to find the shortest possible route from Beijing to Shanghai. Given a road map of China on which the distance between each pair of adjacent intersections is marked, how can he determine this shortest route?

application	vertex	edge
map	intersection	road
network	router	connection
schedule	job	precedence constraint
arbitrage	currency	exchange rate

Typical shortest-paths applications

最短路径的定义(1/2)

定义

设G为一个带权有向图，权重函数为$w : E \to \mathbb{R}$. 图中一条路径$p=(v_0, ..., v_k)$的权重(*path weight*) $w(p)$为

$$\sum_{i=1}^{k} w(v_{i-1}, v_i).$$

定义结点u到结点v的最短路径距离

$$\delta(u, v) = \begin{cases} -\infty & \exists u \rightsquigarrow v\text{且包含权重为负值的回路} \\ \min\left\{ w(p) : u \overset{p}{\rightsquigarrow} v \right\} & \forall u \rightsquigarrow v\text{且不包含权重为负值的回路} \\ \infty & \text{否则} \end{cases}$$

结点u到v的最短路径定义为任何一条权重为$w(p)=\delta(u, v)$的从u到v的路径p.

最短路径的定义(2/2)

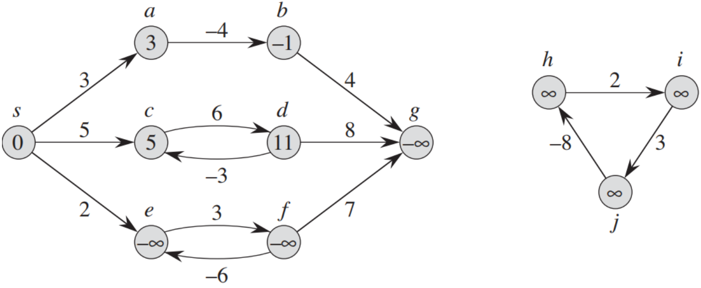


图: 有向图的负权重边

最短路径问题的不同形式

(1) 单源最短路径问题(Single-source shortest-paths problem): 源点$s \in V$, 求从s出发到达其它结点的最短路径.

(2) 单终点最短路径问题(Single-destination shortest-paths problem).

(3) 单结点对最短路径问题(Single-pair shortest-path problem).

(4) 所有结点对最短路径问题(All-pairs shortest-paths problem).

最短路径的最优子结构

引理 (Subpaths of shortest paths are shortest paths)

设G为一个带权有向图, 权重函数为$w: E \to \mathbb{R}$, 且

$$p = (v_0, \ldots, v_k)$$

是图G中从结点v_0到结点v_k的一条最短路径. 则对于任意i和j, $0 \le i \le j \le k$,

$$(v_i, v_{i+1}, \ldots, v_j)$$

是从结点v_i到结点v_j的一条最短路径.

最短路径与回路的关系

(1) 一条最短路径不能包含权重为负值的回路;

(2) 一条最短路径不能包含权重为正值的回路;

(3) 如果一条最短路径包含权重为0的回路, 则一定存在一条最短路径不包含权重为0的回路.

寻找最短路径只需要在至多只包含$|V|-1$条边的路径上.

最短路径的表示(1/2)

给定带权有向图G, 对于每个结点$v\in V$, 维持一个前驱结点$v.p$. 由p值所诱导的前驱子图(**predecessor subgraph**) $G_p=(V_p, E_p)$:

$$V_p=\{v\in V \mid v.p\neq \mathbf{NIL}\}\cup\{s\},$$

$$E_p=\{(v.p,\ v)\in E \mid v\in V_p-\{s\}\}.$$

一棵根结点s的最短路径树(**shortest-paths tree**)是一个有向子图$G'=(V', E')$, 这里$V'\subseteq V$, $E'\subseteq E$, 满足:

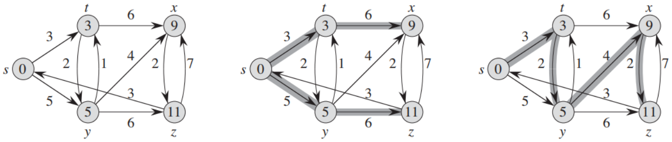
(1) V'是图G中从源结点s可以到达的所有结点的集合.

(2) G'形成一棵根结点为s的有向树.

(3) 对于所有的$v\in V'$, 有向树s到v的路径是图G中s到v的一条最短路径.

最短路径的表示(2/2)

最短路径和最短路径树可能都不是唯一的.



(a) (b) (c)

本章算法的两个基本操作:初始化

算法4.1 **Initialize-Single-Source**(G, s)

for each vertex $v \in V$
    $v.d \leftarrow \infty$
    $v.p \leftarrow$ **NIL**
$s.d \leftarrow 0$

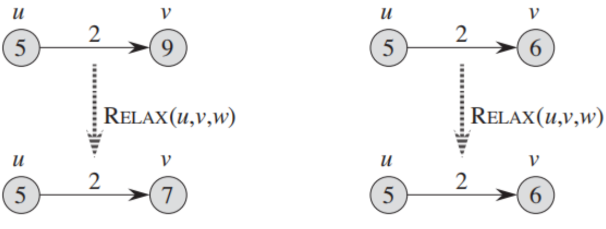
时间复杂度: $O(|V|)$.

本章算法的两个基本操作: 松弛过程

算法4.2 Relax(u, v)

```
if u.d+w(u, v)<v.d
        v.d ← u.d+w(u, v)
        v.p ← u
```

时间复杂度: $O(1)$.



(a) (b)

Bellman-Ford算法

给定带权有向图G, **Bellman-Ford算法**返回一个布尔值, 以表明是否存在一个从源结点可以到达的权重为负值的回路.

(1) 若存在权重为负值的回路, 算法告诉不存在解决方案.

(2) 如果没有权重为负值的回路, 算法将给出最短路径和它们的权重.

Bellman-Ford算法的伪代码

算法4.3 **Bellman-Ford**(G, w, s)

```
Initialize-Single-Source(G, s)
for i ← 1 to |V|−1
    for each edge (u, v)∈E
        Relax(u, v)
for each edge (u, v)∈E
    if v.d>u.d+w(u, v)
        return FALSE
return TRUE
```

时间复杂度: $O(|V||E|)$

Bellman-Ford算法执行过程

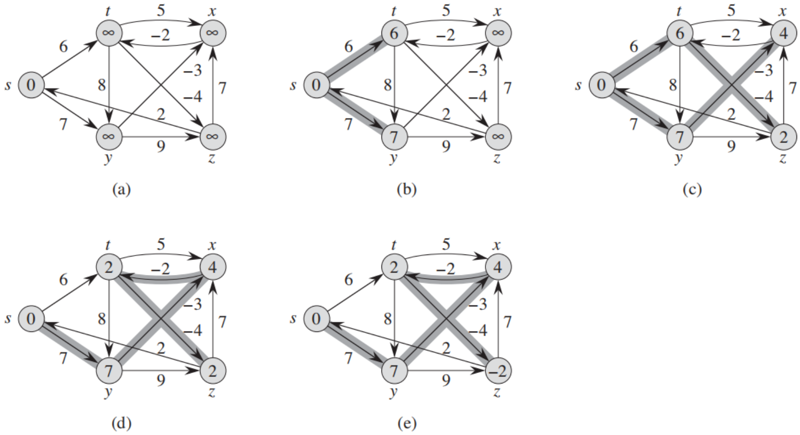


图: 每一次的松弛操作对边的处理次序都是:

$(t, x), (t, y), (t, z), (x, t), (y, x), (y, z), (z, x), (z, s), (s, t), (s, y)$

引理 (Triangle inequality)

设G为一个带权有向图, 权重函数为$w: E \to \mathbb{R}$, 其源结点为s. 则对于所有的边$(u, v) \in E$, 有

$$\delta(s, v) \leq \delta(s, u) + w(u, v).$$

最短路径估计值的松弛效果:上界性质

引理 (Upper-bound property)

设G为一个带权有向图, 权重函数为$w: E\rightarrow\mathbb{R}$, 其源结点为$s$, 该图由算法***Initialize-Single-Source***(G, s)执行初始化. 则:

(1) 对于所有的结点$v\in V$, $v.d\geq\delta(s, v)$, 并且该不变式在对图G的边进行任何次序的松弛过程中保持成立;

(2) 另外, 一旦$v.d=\delta(s, v)$, 不再发生变化.

推论 (No-path property)

设G为一个带权有向图, 权重函数为$w: E\rightarrow\mathbb{R}$, 设从源结点$s$到给定结点$v$之间不存在路径, 则在该图由算法***Initialize-Single-Source***(G, s)进行初始化后, 有$v.d=\delta(s, v)=\infty$, 且该等式作为不变式一直维持到图G的所有松弛操作结束.

最短路径估计值的松弛效果:上界性质的证明

证明 我们对松弛步骤的数量使用**归纳法**来证明不变式, 即对于所有的结点$v \in V$, $v.d \geq \delta(s, v)$.

基础步: 在初始化之后, 对于所有的结点$v \in V$, $v.d \geq \delta(s, v)$ 显然成立.

归纳步: 考虑对边(u, v)的松弛操作. 根据**归纳假设**, 在松弛之前, 对于所有的结点$v' \in V$, $v'.d \geq \delta(s, v')$.

对边(u, v)进行松弛的过程中, 唯一可能发生改变的d值只有$v.d$. 如果该值发生变化, 则有

$$\begin{aligned} v.d &= u.d + w(u, v) \\ &\geq \delta(s, u) + w(u, v) \\ &\geq \delta(s, v) \end{aligned}$$

因此不变式得到维持.

最短路径估计值的松弛效果

引理

设G为一个带权有向图，权重函数为$w: E \to \mathbb{R}$，且$(u, v) \in E$。则对边(u, v)进行松弛操作$\textbf{\textit{Relax}}(u, v)$后，有

$$v.d \leq u.d + w(u, v).$$

最短路径估计值的松弛效果:收敛性质

引理 (Convergence property)

设G为一个带权有向图, 权重函数为$w : E \to \mathbb{R}$.

(1) 设$u, v \in V$, $s \rightsquigarrow u \to v$为图$G$中的一条最短路径.

(2) 设图G由***Initialize-Single-Source***(G, s)算法进行初始化, 并在这之后进行了一系列边的松弛操作, 其中包括对边(u, v)的松弛操作 ***Relax***(u, v).

如果在对边(u, v)进行松弛操作之前的任意时刻有

$$u.d = \delta(s, u),$$

则在该松弛操作之后的所有时刻有

$$v.d = \delta(s, v).$$

最短路径估计值的松弛效果:收敛性质的证明

证明 根据**上界性质**, 如果在对边(u, v)进行松弛前的某个时刻有

$$u.d=\delta(s, u),$$

则该等式在松弛操作后仍然成立. 特别地, 在对边(u, v)进行松弛后, 有

$$\begin{aligned} v.d &\le u.d+w(u, v) \\ &=\delta(s, u)+w(u, v) \\ &=\delta(s, v) \end{aligned}$$

根据**上界性质**, 有$v.d\ge\delta(s, v)$.
因此$v.d=\delta(s, v)$, 并且该等式在此之后一直保持成立.

最短路径估计值的松弛效果:路径松弛性质

引理 (Path-relaxation property)

设G为一个带权有向图, 权重函数为$w: E\rightarrow\mathbb{R}$, 其源结点为$s$. 考虑从源结点$s$到结点$v_k$的任意一条最短路径

$$p=(v_0, v_1, ..., v_k),$$

这里$s=v_0$. 如果图G由***Initialize-Single-Source***(G, s) 算法进行初始化, 并在这之后进行了一系列的边松弛操作, 其中包括对边

$$(v_0, v_1), (v_1, v_2), ..., (v_{k-1}, v_k),$$

按照所列次序而进行的松弛操作, 则在所有这些松弛操作之后, 有

$$v_k.d=\delta(s, v_k),$$

并且在此之后该等式一直保持成立. 该性质的成立与其他边的松弛操作及次序无关.

引理 (Predecessor-subgraph property)

设G为一个带权有向图, 权重函数为$w: E \rightarrow \mathbb{R}$, 其源结点为$s$.

(1) 假设图G不包含从源结点s可以到达的权重为负值的环路.

(2) 假设调用***Initialize-Single-Source***(G, s)算法对图G进行初始化, 然后对图G的边进行任意次序的松弛操作. 该松弛操作序列将针对所有的结点$v \in V$生成$v.d = \delta(s, v)$,

则前驱子图G_p形成一棵根结点为s的最短路径树.

前驱子图性质的证明(1/5)

证明 必须证明**最短路径树**的三条性质对于G_p都成立.
要证明第一条性质, 必须证明:

V_p是从源结点s可以到达的结点的集合.

根据定义,

最短路径$\delta(s, v)$是有限值当且仅当结点v是从s可以到达的,

因此,

从源结点s可以到达的结点就是那些有着有限d值的结点.

但对于结点$v \in V-\{s\}$, 其被赋予有限d值当且仅当$v.p \neq \mathrm{NIL}$. 因此,

V_p中的结点就是那些可以从源结点s到达的结点.

前驱子图性质的证明(2/5)

单源最短路径问题

目录

最短路径的定义与性质

Bellman-Ford算法

最短路径性质

最短路径估计值的松弛效果

松弛操作与最短路径树

Bellman-Ford算法的正确性证明

有向无环图中的单源最短路径问题

Dijkstra算法

广度优先搜索算法

应用

第二条性质可以从以下结论直接推导出来.

在图G由**Initialize-Single-Source**(G, s)算法进行初始化之后，**前驱子图**G_p形成根结点为源结点s的有根树，并且任何对图G的边进行的任意松弛操作都将维持该性质不变.

在初始化时，G_p中的唯一结点是源结点s，结论显然成立.
考虑在一系列松弛操作后的**前驱子图**G_p. 首先证明G_p是无环路的. 假定在松弛序列的某个步骤上在图G_p中创立了一个环路. 设该环路为

$$c=(v_0, v_1, ..., v_k),$$

这里$v_k=v_0$. 则$v_i.p=v_{i-1}$, $i=1, 2, ..., k$, 并且不失一般性，假定在对边(v_{k-1}, v_k)进行松弛操作时创建了G_p中的该条环路.

所有环路c上的结点都可以从源结点s到达.

前驱子图性质的证明(3/5)

下面检查在调用**Relax**(v_{k-1}, v_k)操作之前c上面的最短路径估计值，并证明c是一个权重为负值的环路，从而推出与假设(图G不包含从源结点s可以到达的权重为负值的环路)矛盾.

在该调用发生前，有$v_i.p=v_{i-1}$, $i=1, 2, ..., k-1$. 因此，对于$i=1, 2, ..., k-1$，对于$v_i.d$ 的最后一次更新必定是$v_i.d=v_{i-1}.d+w(v_{i-1}, v_i)$. 如果$v_{i-1}.d$在此之后发生变化，则一定是减少了. 因此，在调用**Relax**(v_{k-1}, v_k)操作之前，有

$$v_i.d \geq v_{i-1}.d+w(v_{i-1}, v_i), \forall i = 1, 2, ..., k-1 \tag{1}$$

因为$v_k.p$在该调用中发生改变，所有在此之前有

$$v_k.d > v_{k-1}.d+w(v_{k-1}, v_k) \tag{2}$$

从以上两个不等式，我们有:

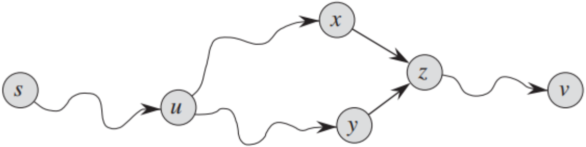
$$\sum_{i=1}^{k} v_i.d > \sum_{i=1}^{k} (v_{i-1}.d+w(v_{i-1}, v_i)) \Longrightarrow 0 > \sum_{i=1}^{k} w(v_{i-1}, v_i).$$

前驱子图性质的证明(4/5)

为了证明其形成一棵根结点为s的有根树，只要证明对于每个结点$v \in V_p$，在图G_p中存在一条从源结点s到结点v的唯一简单路径.

首先证明对于结点$v \in V_p$，在图G_p中存在一条从源结点s到结点v的路径.

为了完成对引理的证明，我们还必须证明对于每个结点$v \in V_p$，图G_p至多包含一条从源结点s到结点v的路径. 我们采用反证法.



前驱子图性质的证明(5/5)

目录
最短路径的定义与性质
Bellman-Ford算法
最短路径性质
最短路径估计值的松弛效果
松弛操作与最短路径树
Bellman-Ford算法的正确性证明
有向无环图中的单源最短路径问题
Dijkstra算法
广度优先搜索算法
应用

最后证明最短路径树的第三条性质：对于每个结点$v \in V_p$, G_p中的唯一简单路径$p=(v_0, v_1, ..., v_k)$, $v_0=s$, $v_k=v$, 是图G中从s到v的一条最短路径.

对于$i = 1, 2, \ldots, k$, 有$v_i.d=\delta(s, v_i)$ 和

$$v_i.d \geq v_{i-1}.d + w(v_{i-1}, v_i).$$

所以:

$$w(v_{i-1}, v_i) \leq \delta(s, v_i) - \delta(s, v_{i-1}).$$

$$w(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i) \leq \sum_{i=1}^{k} (\delta(s, v_i) - \delta(s, v_{i-1})) = \delta(s, v_k)$$

Bellman-Ford算法的正确性

定理 (Correctness of the Bellman-Ford algorithm)

设***Bellman-Ford***算法运行在一个带权重的源结点为s的有向图G上，其权重函数为$w: E \rightarrow \mathbb{R}$.

(1) 如果图G不包含从源结点s可以到达的权重为负值的环路，则：

- 算法将返回***TRUE***值；
- 对于所有结点$v \in V$, $v.d = \delta(s, v)$;
- **前驱子图**G_p是一棵根结点为s的最短路径树.

(2) 如果图G包含从源结点s可以到达的权重为负值的环路，则算法将返回***FALSE***值.

Bellman-Ford算法的正确性证明(1/3)

证明 (1) 使用**路径松弛性质**证明:

对于所有结点$v \in V$, $v.d=\delta(s, v)$.

- 考虑任意从源结点s可以到达的结点v.
  设$p=(v_0, v_1, \ldots, v_k)$为从源结点$s$到结点$v$之间的任意一条最短路径, 其中$v_0=s$, $v_k=v$. p最多有$|V|-1$条边, 因此$k \leq |V|-1$. 算法第2-4行的**for**循环每次松弛所有的$|E|$条边. 在第i次松弛操作时, 这里$i=1, 2, \ldots, k$, 被松弛的边中包含(v_{i-1}, v_i). 根据**路径松弛性质**,

$$v.d=v_k.d=\delta(s, v_k)=\delta(s, v).$$

- 结点v是不能从源结点s可以到达的结点.
  由**非路径性质**, 得到$v.d=\delta(s, v)$.

结合**前驱子图性质**可以推出G_p是一棵最短路径树.

Bellman-Ford算法的正确性证明(2/3)

算法终止时, 对于所有的边$(u, v)\in E$, 有

$$
\begin{aligned}
v.d &= \delta(s, v) \\
&\leq \delta(s, u) + w(u, v) \\
&= u.d + w(u, v)
\end{aligned}
$$

因此, 算法第6行中没有任何测试可以让**Bellman-Ford**算法返回**FALSE**值. 因此, 它一定返回的是**TRUE**值.

Bellman-Ford算法的正确性证明(3/3)

(2) 假设图G包含一个权重为负值的环路, 且该环路可以从源结点s到达; 设该环路为$c=(v_0, v_2, ..., v_k)$, 这里$v_0=v_k$, 则有

$$\sum_{i=1}^{k} w(v_{i-1}, v_i)<0.$$

下面使用反证法. 假设算法返回的是**TRUE**, 则

$$v_i.d\leq v_{i-1}.d+w(v_{i-1}, v_i),\ i=1, 2, ..., k$$

$$\sum_{i=1}^{k} v_i.d\leq \sum_{i=1}^{k}(v_{i-1}.d+w(v_{i-1}, v_i)) \Longrightarrow 0\leq \sum_{i=1}^{k} w(v_{i-1}, v_i).$$

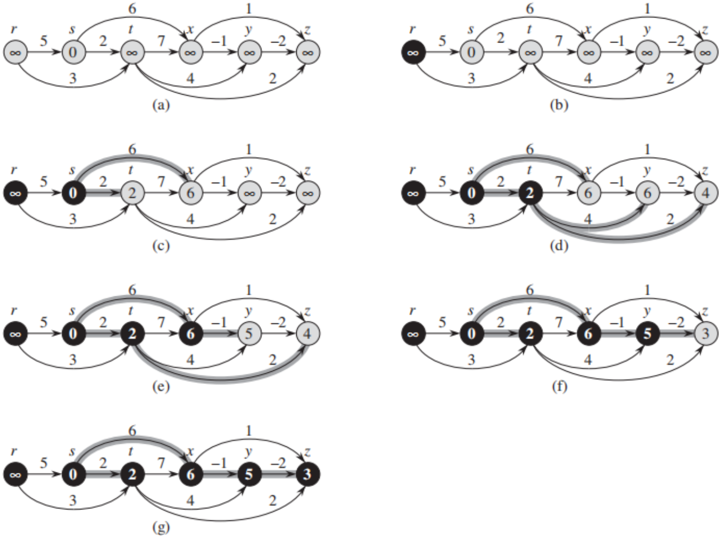
有向无环图中的单源最短路径问题

算法4.4 **DAG-Shortest-Paths**(G, w, s)

```
topologically sort the vertices of G
Initialize-Single-Source(G, s)
for each vertex u, taken in topologically sorted order
        for each edge (u, v)∈E
                Relax(u, v)
```

算法的时间复杂度: $O(|V|+|E|)$.

实例



DAG-Shortest-Paths算法的正确性

定理

如果带权重无环有向图$G=(V, E)$有一个源结点s, 则在算法*DAG-Shortest-Paths*终止时, 对于所有的结点$v \in V$, 有$v.d=\delta(s, v)$, 且**前驱子图**G_p是一棵**最短路径树**.

证明 首先证明对于所有的结点$v \in V$, 在算法**DAG-Shortest-Paths** 终止时, $v.d=\delta(s, v)$.

- 如果结点v不能从源结点s到达, 则根据**非路径性质**, 有

$$v.d=\delta(s, v)=\infty.$$

- 如果结点v能从源结点s到达, 因此, 图中存在一条最短路径$p=(v_0, v_1, ..., v_k)$, 其中$v_0=s$, $v_k=v$. 因为算法是按照拓扑排序的次序来对结点进行处理, 所以对路径p上的边的放松次序为$(v_0, v_1), (v_1, v_2), ..., (v_{k-1}, v_k)$. 根据**路径松弛性质**, 对于$i=0, 1, ..., k$, 在算法终止时有$v_i.d=\delta(s, v_i)$.

最后, 根据**前驱子图性质**, G_p是一棵**最短路径树**.

Dijkstra算法的基本思想

Dijkstra算法正确性前提: 所有边的权重都是非负值.

Dijkstra算法的基本思想:

(1) 一组结点集合S, $v \in S \Longleftrightarrow$从$s$到$v$的最短路径长度已知.

(2) 算法重复从结点集$V-S$中选择最短路径估计最小的结点u, 将u加入到集合S, 然后对所有的从u发出的边进行松弛.

(3) 使用一个**最小优先队列**Q来保存结点集合, 每个结点的关键值为其d值.

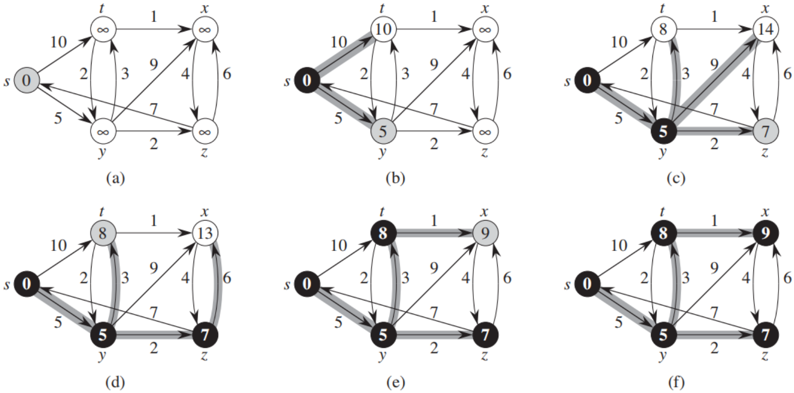
Dijkstra算法的伪代码

算法4.5 **Dijkstra** (G, w, s)

Initialize-Single-Source(G, s)
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$
  $u \leftarrow$ **Extract-Min**(Q)
  $S \leftarrow S \cup \{u\}$
  for each edge $(u, v) \in E$
    if $v \notin S$ and $v.d > u.d + w(u, v)$
      Decrease-Key(Q, v, $u.d + w(u, v)$)
      $v.p \leftarrow u$

时间复杂度: $O(|E| \log |V|)$.

Dijkstra算法的实例



Dijkstra算法的正确性(1/2)

定理 (Correctness of Dijkstra's algorithm)

***Dijkstra*算法**运行在带权重的有向图G时, 如果所有的权重都为非负值, 则在算法终止时, 对于所有结点$u \in V$, 有

$$u.d = \delta(s, u),$$

且**前驱子图**G_p是一棵**最短路径树**.

证明 我们使用下面的循环不变式:

在算法第4-10行的**while**语句的每次循环开始前, 对于每个结点$v \in S$, 有$v.d = \delta(s, v)$.

由**上界性质**, 我们只要证明:

对于每个结点$u \in V$, 当结点u被加入到集合S时, 有$u.d = \delta(s, u)$.

Dijkstra算法正确性证明(2/2)

初始化: $S=\emptyset$, 因此, 循环不变式成立.

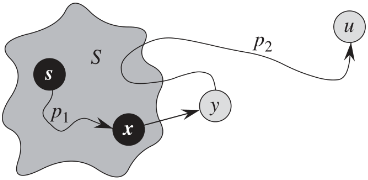
保持: 用反证法证明: 对于每个结点$u\in V$, 当结点u被加入到集合S 时, 有$u.d=\delta(s, u)$.

设结点u是第一个加入到集合S时, 使得$u.d\neq\delta(s, u)$的结点. 则:

(1) $u\neq s$. 因此把结点u放入到集合S时, $S\neq\emptyset$.

(2) s到u有路径, 否则根据**非路径性质**, $u.d=\delta(s, u)=\infty$.

因为s到u有路径, 则存在一条从s到u的最短路径p.



断言: 根据**收敛性质**, 在将结点u放入到集合S时, $y.d=\delta(s, y)$.

Dijkstra算法正确性证明(2/2)

通过反证法证明$u.d=\delta(s, u)$. $\delta(s, y)\le\delta(s, u)$成立.

$$y.d=\delta(s, y)$$
$$\le\delta(s, u)$$
$$\le u.d$$

但是, 在算法第5行选择结点u时, 结点u, $y\in V-S$, 所以有

$$u.d\le y.d.$$

因此

$$u.d=y.d.$$

广度优先搜索算法的数据结构

(1) 存储结点v在搜索过程的状态;

(2) 队列Q存储灰色结点集;

(3) 变量$v.d$记录源结点s到结点v最短路径距离.

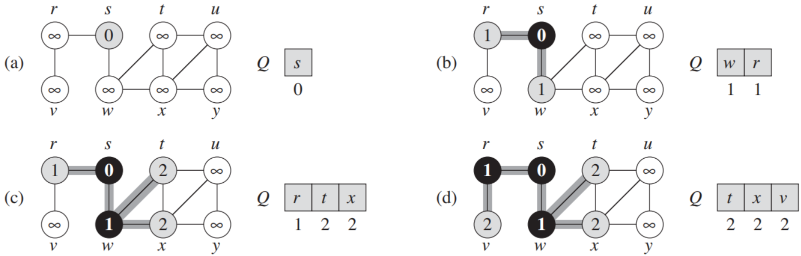
BFS算法的伪代码

算法4.6 **BFS**(G, s)

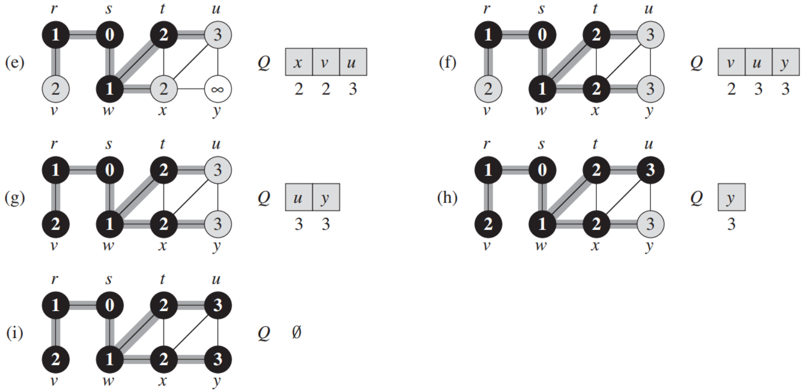
```
for each vertex u∈V−{s}
        u.color ← white, u.d ←∞, u.p← NIL
s.color ← gray, s.d ← 0, s.p ← NIL
Q← ∅
Enqueue(Q, s)
while Q≠∅
        u← Dequeue(Q)
        for each (u, v)∈E
            if v.color=white
                v.color← gray
                v.d←u.d+1
                v.p←u
                Enqueue(Q, v)
        u.color← black
```

存储结构为链接表时, **BFS**算法的时间复杂度: $O(|V|+|E|)$.

实例(1/2)


(a)
r s t u
∞ 0 ∞ ∞
∞ ∞ ∞ ∞
v w x y
Q s
0
(b)
r s t u
1 0 ∞ ∞
∞ 1 ∞ ∞
v w x y
Q w r
1 1
(c)
r s t u
1 0 2 ∞
∞ 1 2 ∞
v w x y
Q r t x
1 2 2
(d)
r s t u
1 0 2 ∞
2 1 2 ∞
v w x y
Q t x v
2 2 2

实例(2/2)


(e)
r s t u
1 0 2 3
2 1 2 ∞
v w x y
Q x v u
2 2 3
(f)
r s t u
1 0 2 3
2 1 2 3
v w x y
Q v u y
2 3 3
(g)
r s t u
1 0 2 3
2 1 2 3
v w x y
Q u y
3 3
(h)
r s t u
1 0 2 3
2 1 2 3
v w x y
Q y
3
(i)
r s t u
1 0 2 3
2 1 2 3
v w x y
Q ∅

BFS算法的正确性的证明(1/7)

引理

给定有向或无向图G, 假设从以结点$s \in V$开始广度优先搜索, 则在算法终止时, 对每个结点$v \in V$, 有

$$v.d \geq \delta(s, v).$$

证明 采用数学归纳法, 对队列上的操作个数做归纳, 也就是要证明, 无论队列上执行了多少操作, 下面的不等式总是成立: 对于每个结点$v \in V$, 有$v.d \geq \delta(s, v)$.

- 归纳基础: 队列执行第一个操作, 即将源结点s加入队列Q中.
- 对于归纳步, 考虑从结点u进行边搜索时发现白色结点v. 根据归纳假设, 有$u.d \geq \delta(s, u)$. 根据算法第11行, 有

$$\begin{aligned} v.d &= u.d + 1 \\ &\geq \delta(s, u) + 1 \\ &\geq \delta(s, v). \end{aligned}$$

BFS算法的正确性的证明(2/7)

引理

设***BFS*算法**在图G上运行的过程中，队列Q包含的结点为

$$(v_1,\ v_2, ..., v_r),$$

其中v_1是队列Q的头，v_r是队列Q的尾. 则:

(1) $v_r.d \le v_1.d+1$;

(2) 对于$i=1,\ 2, ..., r-1$，有$v_i.d \le v_{i+1}.d$.

证明 通过对算法里面入队操作的次数进行数学归纳法来证明本引理.

- 归纳基础: 在初始情况下，队列Q里仅包含源结点s，引理明显成立.

BFS算法的正确性的证明(3/7)

- **归纳步骤**: 要证明在出队和入队操作时, 引理成立.
  如果头结点v_1出队, v_2就变成队列中的新的头结点, 根据**归纳假设**, 有$v_2.d \le \ldots \le v_r.d$.
  注意到:

$$v_r.d \le v_1.d+1,\ v_1.d \le v_2.d$$

所以:

$$v_r.d \le v_2.d+1.$$

因此, 在v_2为头结点时, **引理成立**.
根据BFS算法, 当头结点v_1出队后, 设结点v_{r+1}入队, 且

$$v_{r+1}.d = v_1.d+1.$$

因为$v_1.d \le v_2.d$, 所以$v_{r+1}.d \le v_2.d+1$. 因为

$$v_r.d \le v_1.d+1,\ v_{r+1}.d = v_1.d+1,$$

所以$v_r.d \le v_{r+1}.d$.

BFS算法的正确性的证明(4/7)

推论

假设在运行***BFS*算法**时,结点v_i和结点v_j都加入到队列Q中,且v_i在v_j前面入队,则在v_j入队时,有

$$v_i.d \le v_j.d.$$

BFS算法的正确性的证明(5/7)

定理

设G是一个有向或无向图，又假设设***BFS***算法以s为源结点在图G上运行. 则:

(1) 在算法执行过程中，***BFS***算法将发现从源结点s可以到达的所有结点$v \in V$;

(2) 在算法终止时，对于所有的结点$v \in V$, $v.d = \delta(s, v)$;

(3) 对于任意可以从s到达的结点$v (v \neq s)$, 从源结点s到结点v的其中一条最短路径为从结点s到结点$v.p$的最短路径加上边$(v.p, v)$.

证明 (2) 采用反证法, 假设存在一些结点, 它们的d不等于结点s到它们的最短路径距离. 在这些结点中, 取结点s到其距离最短的结点, 记为v, 显然, $v \neq s$. 根据已知结论, 有

$$v.d > \delta(s, v).$$

注意s到v必然可达, 否则$\delta(s, v) = \infty \geq v.d$.
考察s到v的最短路径. 记u为该路径上在v前面的结点, 则

$$\delta(s, v) = \delta(s, u) + 1.$$

根据选取结点v的特定方式, 有$\delta(s, u) = u.d$. 因此

$$\begin{aligned} v.d &> \delta(s, v) \\ &= \delta(s, u) + 1 \\ &= u.d + 1. \end{aligned}$$

BFS算法的正确性的证明(7/7)

下面考察结点u刚刚出队的时刻(算法第7行). 此时结点v可能有3种颜色, 分情形讨论:

- 如果v.color=*white*, 根据**BFS算法**, $v.d=u.d+1$.
- 如果v.color=*gray*, 则v是在某个结点w出队时被涂上灰色的, 而结点w是在结点u之前出队. 于是有

$$\begin{aligned} v.d &= w.d+1 \\ &\le u.d+1. \end{aligned}$$

- 如果v.color=*black*, 此时该结点已经出队, 则有

$$\begin{aligned} v.d &\le u.d \\ &\le u.d+1. \end{aligned}$$

差分约束系统

(1) 设A是一个$m \times n$的矩阵, b是一个m维向量, 希望找到一个可行解, 即找到任何满足

$$Ax \le b$$

的向量x, 或者判断不存在可行解.

(2) 差分约束系统(**system of difference constraints**):
$Ax \le b$中的矩阵A:

- 每一行包括一个1和一个-1,
- 其他所有项都为0.

差分约束系统的例子

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 & -1 \\ -1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \le \begin{bmatrix} 0 \\ -1 \\ 1 \\ 5 \\ 4 \\ -1 \\ -3 \end{bmatrix}$$

即

$$x_1 - x_2 \le 0 \qquad x_1 - x_5 \le -1$$

$$x_2 - x_5 \le 1 \qquad x_3 - x_1 \le 5$$

$$x_4 - x_1 \le 4 \qquad x_4 - x_3 \le -1$$

$$x_5 - x_3 \le -3$$

约束图

给定差分约束系统

$$Ax \le b,$$

其对应的约束图(constraint graph)是一个带权重的有向图

$$G=(V, E),$$

其中:

$$V=\{v_0, v_1, \ldots, v_n\},$$

$$E=\{(v_i, v_j) : x_j - x_i \le b_k \text{是一个约束条件}\} \cup$$

$$\{(v_0, v_1), \ldots, (v_0, v_n)\},$$

权重函数$w : E \rightarrow \mathbb{R}$定义为:

$$w(v_0, v_i)=0,\ i=1, \ldots, n$$

$$w(v_i, v_j)=b_k, \text{如果} x_j - x_i \le b_k \text{是一个约束条件}$$

约束图的实例

$x_1 - x_2 \le 0$

$x_1 - x_5 \le -1$

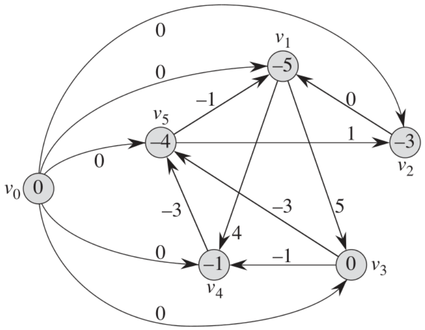
$x_2 - x_5 \le 1$

$x_3 - x_1 \le 5$

$x_4 - x_1 \le 4$

$x_4 - x_3 \le -1$

$x_5 - x_3 \le -3$



约束图的性质

定理

给定差分约束系统 $Ax \le b$, 设 $G=(V, E)$ 是该差分约束系统所对应的约束图.

(1) 如果图 G 不包含权重为负值的环路, 则

$$x=(\delta(v_0, v_1), \ldots, \delta(v_0, v_n))$$

是该系统的一个可行解.

(2) 如果图 G 包含权重为负值的环路, 则该系统没有可行解.

证明 (1)考虑任意一条边 $(v_i, v_j) \in E$, 根据三角不等式,

$$\delta(v_0, v_j) \le \delta(v_0, v_i) + w(v_i, v_j)$$

令 $x_i=\delta(v_0, v_i)$, $x_j=\delta(v_0, v_j)$. 则

$$x_j - x_i \le w(v_i, v_j).$$

约束图的性质的证明

(2) 设权重为负值的环路为$c=(v_1, v_2, ..., v_k)$, 其中$v_1=v_k$. 环路c对应的差分约束条件组:

$$x_2-x_1\leq w(v_1, v_2)$$

$$x_3-x_2\leq w(v_2, v_3)$$

……

$$x_{k-1}-x_{k-2}\leq w(v_{k-2}, v_{k-1})$$

$$x_k-x_{k-1}\leq w(v_{k-1}, v_k).$$

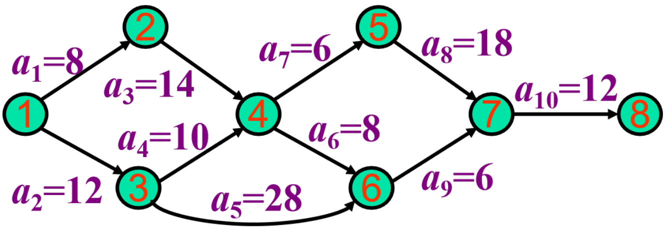
假设向量x有一个满足上述k个不等式的解, 则$w(c)\geq 0$.

定义

(1) 如果在**有向无环图**中,
  (1) 有向边表示一个工程中的各项活动, 用有向边上的权值表示活动的持续时间,
  (2) 用结点表示事件,

则这种有向图称为用边表示活动的网络.

(2) 用边表示活动的网络中, 从源点到汇点路径长度最长的路径称为**关键路径**.



关键路径算法的伪代码

算法4.7 **CriticalPath**(G, s, t)

topologically sort the vertices of G
for each vertex $v \in V$
    $v.d \leftarrow 0$
for each vertex u, taken in topologically sorted order
    for each edge $(u, v) \in E$
        if $v.d < u.d + w(u, v)$ **then**
            $v.d \leftarrow u.d + w(u, v)$
return $t.d$

算法的时间复杂度: $O(|V|+|E|)$.

作业

(1) 编程实现**Bellman-Ford**算法.

(2) 编程实现有向无环图中的单源最短路径问题的算法.

Thank you!