Bellman-Ford 的 Yen 优化

报告人 ： HNU 陈汉轩 19/6/10

原始的算法：

Algorithm 2 The basic Bellman–Ford algorithm

```
for i ← 1 to n − 1 do
    for each edge uv in graph G do
        relax(u, v)
```

Algorithm 1 Procedure `relax`(u, v): relax the edge from u to v.

```
if D[v] > D[u] + length(u, v) then
    D[v] ← D[u] + length(u, v)
    P[v] ← u
```

这个算法的原理：

- 每轮松弛操作中，至少会出现一次有效的修正，得到至少一个正确的v.d，在最多经过了|V|-1轮之后，所有的节点的距离值全部被修正。

算法的优化：

- 目前，对带负权的单源最短路径，Bellman-Ford算法就是渐进时间最优的，下面我们讨论一下怎么优化常数因子，并计算证明这些优化算法的有效性。

不使用Yen优化之前，朴素的优化策略是：

- 1.每一轮松弛操作中，只对v.d发生了改变的节点进行relax操作
- 2.如果一轮中没有任何合法的松弛，提前终止算法

- 对于一个稀疏图，影响不大，但是对于一个稠密图，效果明显。

对朴素优化的分析1：

- 对于外层循环 $\textbf{for } i \leftarrow 1 \textbf{ to } n-1 \textbf{ do}$ 假设已经进行了i次，那么在第i次循环开始前，至少应该有i个节点已经获得了正确的距离值，且在之后的循环中不再改变。

- 所以在第i次循环中，最多可能有n-i个节点发生距离值的改变，考虑图极度稠密的情况，假设图是完全图，即任意两个节点相互均有边可达，那么这n-i个节点中的任意一个必然与图中的其余n-1个节点连通。于是在第i次循环中，松弛操作数最大为(n-i)(n-1)。

对朴素优化的分析2：

- 然后对于整个过程求和，设$\mathrm{n} = |V|$，有：(稠密图默认E=O($|V|^2$))
- $\sum_{i=1}^{n-1}(n-i)(n-1) = \frac{n(n-1)^2}{2} < \frac{n^3}{2}$

- 即按照上面的优化策略，可以对稠密图的$\mathrm{O}(|\mathrm{V}||\mathrm{E}|) = \mathrm{O}(|V|^3)$的时间复杂度降低一半，这两个策略结合起来进一步优化，可以得到SPFA算法，可借助队列实现。

Yen 优化

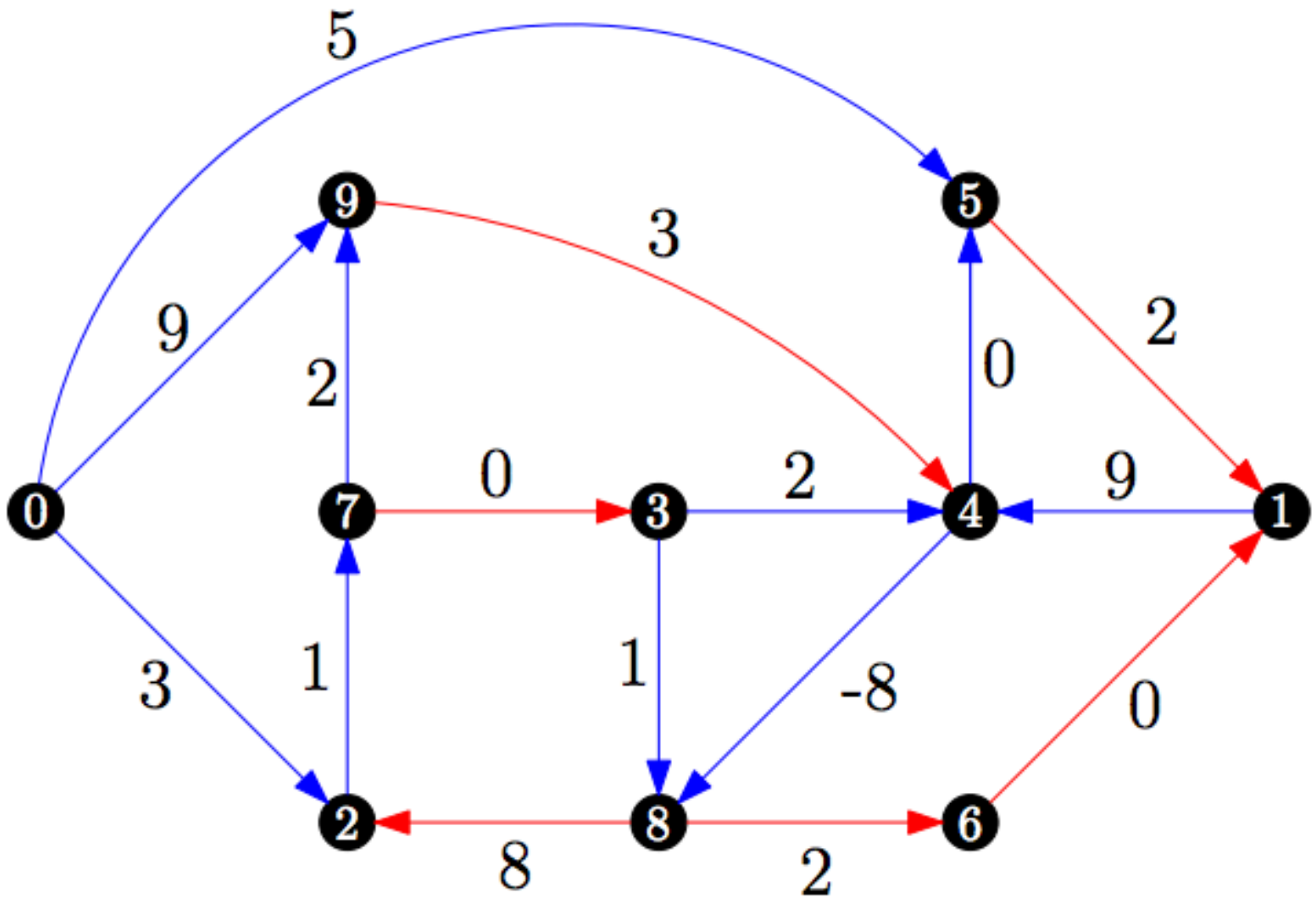
AN ALGORITHM FOR FINDING SHORTEST ROUTES FROM ALL SOURCE NODES TO A GIVEN DESTINATION IN GENERAL NETWORKS*

By JIN Y. YEN (*University of California, Berkeley*)

Summary. This paper presents an algorithm for finding all shortest routes from all nodes to a given destination in N-node general networks (in which the distances of arcs can be negative). If no negative loop exists, the algorithm requires $\frac{1}{2}M(N-1)(N-2)$, $1 < M \leqq N-1$, additions and comparisons. The existence of a negative loop, should one exist, is detected after $\frac{1}{2}N(N-1)(N-2)$ additions and comparisons.

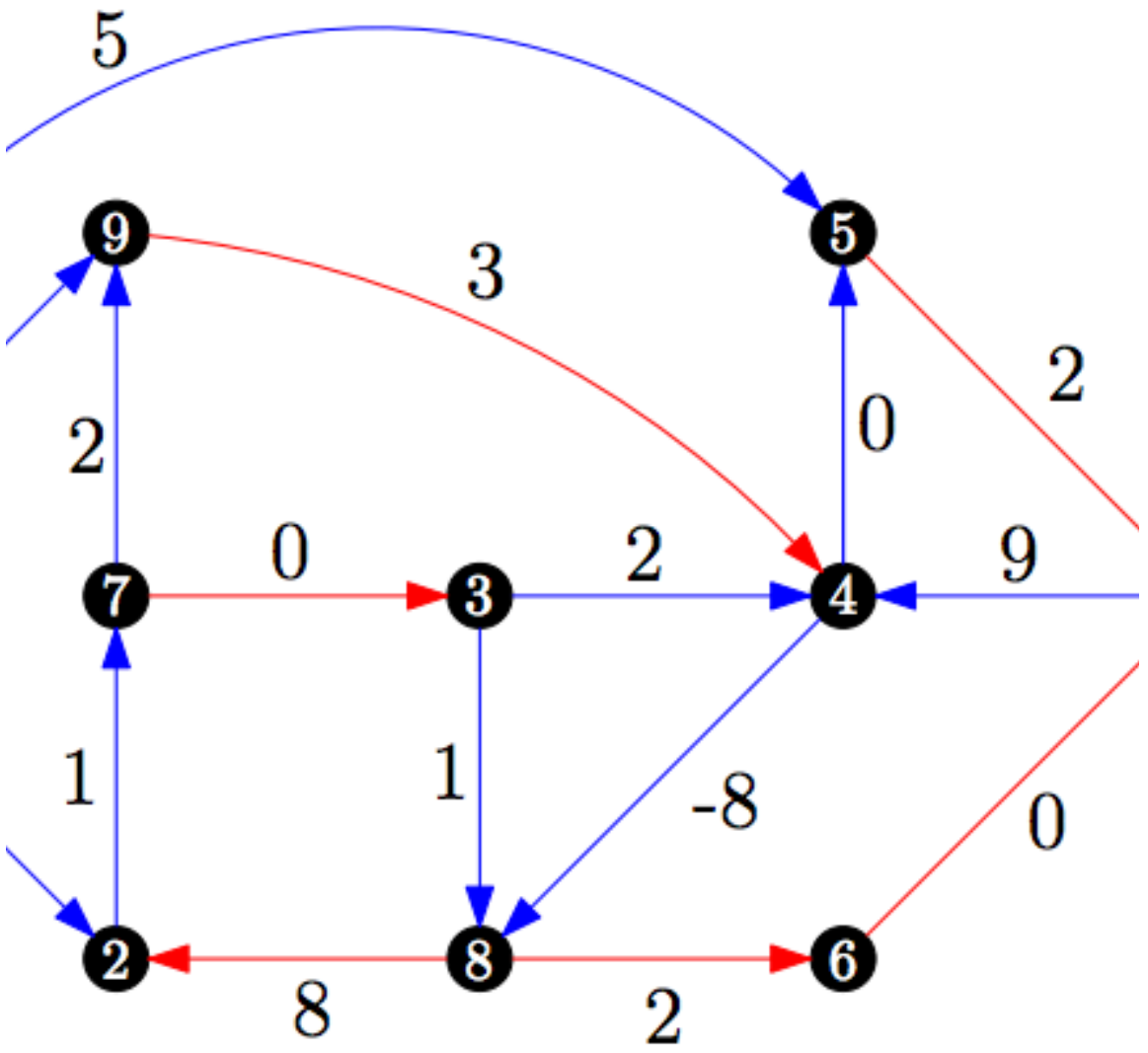
Yen的策略

- 1970年，Yen提出了一种策略，这种策略可以将Bellman-Ford外层循环的次数减少一半，他的策略是在进行松弛操作的时候，将原来的图分为两个有向无环的子图，改进原来的算法的每一轮松弛都按照一个固定的顺序松弛的方式，在每一轮松弛操作中都按照两个确定的顺序对子图分别进行松弛。


5
9
3
9
2
0
2
0
7
0
3
2
4
9
1
3
1
1
-8
0
2
8
8
2
6

Yen算法性质分析1

- 在算法开始任意生成点的下标，然后根据规则将图中所有的边划分为两类，这些边构和其顶点成两个子图G^+和G^-，其中G^+中的边(V_i, V_j)满足i ＜ j，G^-中的边相反，(V_i, V_j)满足i ＞ j。



Yen算法性质分析2

- 子图G^+和G^-满足一个有用的性质：
- G^+的拓扑排序就是G^+中的顶点的升序排列，G^-的拓扑排序就是G^-中的顶点的降序排列，这样的性质在进行松弛操作的时候会很有用。
- 在每一轮松弛操作的时候，Yen优化算法对于两个子图分别处理，对于两个子图，分别按照其顶点的拓扑序列进行松弛，这样每轮操作都可以保证至少有两个点的distance值被修正，从而降低了整体的时间复杂度。

Yen优化思想1

- 对于我们的目标，找到图G中的单源最短路径来说，由于整个图G被切成了两个子图G^+和G^-

- 这就导致原来G图中的单源最短路径$Path_i$也会被切成两段，设为$subPath_{i+}$和$subPath_{i-}$，当然，$subPath_{i+}$和$subPath_{i-}$可能为$\emptyset$

- 考虑在第i轮松弛中，由于同时对两个子图进行操作，不妨设首先对G^+进行松弛操作，然后再对G^-进行松弛操作。

Yen优化思想2

- 在对G^+进行操作后，$Path_i \cap G^+$的部分就完成了一轮松弛，然后再对G^-进行操作后，$Path_i \cap G^-$的部分也成了一轮松弛，这样相当原来同样的一轮循环里做了两份工。在进行了若干轮之后，$Path_i$的两个部分被完全修正，此时整个图的单源最短路径也就得到了。
- 假设进行k轮松弛后，所有的$Path_i$都已经被解决，那么算法会在k+1轮终止，内循环时间复杂度为O(|E|)，那么整体的时间开销为O((K+1)|E|)，K的取值由不等式$K \leq \lceil |V| / 2 \rceil$给出
- 所以，整体的时间复杂度为 $O((\lceil |V| / 2 \rceil + 1)|E|) = O(|E||V|)$

为什么Yen比朴素优化更优

- 对稠密图进行分析，我们会发现Yen策略比上面的两条组合更优：
- 对第i轮松弛操作，由于i轮之前至少完成了2i个顶点的松弛，
- 所以第i轮，最多有(n-2i)的顶点可能被松弛，假设图为完全图，每个顶点应该可以和图中剩下顶点连通，那么这一轮最多可能有(n-1)(n-2i)条边被松弛，求和：
- $\sum_{i=1}^{V/2}(n-2i)(n-1) < \frac{n^3}{4}$

进一步优化

- 之后有人提出了更优的随机化算法，对于稠密图，可以证明期望时间复杂度的常数因子可以限制到小于1/6的范围

Randomized variant of the Bellman–Ford algorithm

ertices randomly such that all permutations with s first

do
ertex u in numerical order **do**
or $D[v]$ has changed since start of iteration **then**
ch edge uv in graph G^+ **do**
ax(u, v)
ertex u in reverse numerical order **do**
or $D[v]$ has changed since start of iteration **then**
ch edge uv in graph G^- **do**
ax(u, v)
ices v for which $D[v]$ changed}

Yen的伪代码

Algorithm 4 Yen's algorithm (adaptive version with early termination)

number the vertices arbitrarily, starting with s
$C \leftarrow \{s\}$
while $C \neq \emptyset$ **do**
  for each vertex u in numerical order **do**
    if $u \in C$ or $D[u]$ has changed since start of iteration **then**
      for each edge uv in graph G^+ **do**
        `relax`(u, v)
  for each vertex u in reverse numerical order **do**
    if $u \in C$ or $D[u]$ has changed since start of iteration **then**
      for each edge uv in graph G^- **do**
        `relax`(u, v)
  $C \leftarrow \{$vertices v for which $D[v]$ changed$\}$

题目24-1 解题

24-1 （Yen 对 Bellman-Ford 算法的改进） 假定对 Bellman-Ford 算法中对边的每一遍松弛操作的次序做出如下规定：在第一遍松弛前，我们给输入图 $G=(V, E)$ 的所有结点赋予一个随机的线性次序 $v_1, v_2, \cdots, v_{|V|}$。然后，将边集合 E 划分为 $E_f \cup E_b$，这里 $E_f=\{(v_i, v_j) \in E: i<j\}$，$E_b=\{(v_i, v_j) \in E: i>j\}$。（假定图 G 不包含自循环，因此一条边要么属于 E_f，要么属于 E_b。）定义 $G_f=(V, E_f)$ 和 $G_b=(V, E_b)$。

a. 证明：G_f 是无环的，且其拓扑排序为 $\langle v_1, v_2, \cdots, v_{|V|} \rangle$；$G_b$ 是无环的，且其拓扑排序为 $\langle v_{|V|}, v_{|V|-1}, \cdots, v_1 \rangle$。

假定我们以下面的方式来实现 Bellman-Ford 算法的每一遍松弛操作：以 $v_1, v_2, \cdots, v_{|V|}$ 的次序访问每个结点，并对从每个结点发出的 E_f 边进行松弛。然后，再以次序 $v_{|V|}, v_{|V|-1}, \cdots, v_1$ 来访问每个结点，并对从每个结点发出的 E_b 边进行松弛。

b. 证明：在上述操作方式下，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，则在 $\lceil |V|/2 \rceil$ 遍松弛操作后，对于所有的结点 $v \in V$，有 $v.d=\delta(s, v)$。

c. 上述算法是否改善了 Bellman-Ford 算法的渐近运行时间？

a. **a.** 证明：G_f 是无环的，且其拓扑排序为$\langle v_1, v_2, \cdots, v_{|V|} \rangle$；$G_b$ 是无环的，且其拓扑排序为$\langle v_{|V|}, v_{|V|-1}, \cdots, v_1 \rangle$。

- 反证法，假设图G_f是有环的，那么一个有环的图必然至少有一条边(v_i, v_j)，满足i > j，否则无法成环，这与E_f的定义相矛盾，所以G_f中没有环，同理可证G_b中没有环。
- 然后证明G_f的拓扑排序结果为$< V_1, V_2, \ldots, V_{|V|} >$：
- 由G_f的定义$G_f = (\mathrm{V}, E_f)$知其边集为E_f，而E_f的定义$E_f = \{(V_i, V_j) \in \mathrm{E}: \mathrm{i} < \mathrm{j}\}$知对于$E_f$中的任意有向边$(V_i, V_j)$均满足i < j，即在拓扑排序中，若有$V_i$,在$V_j$前，那么可以得到i < j成立，所以$G_f$的拓扑排序结果为$< V_1, V_2, \ldots, V_{|V|} >$为升序，同理可证$G_b$的拓扑排序为$V_{|V|}, V_{|V|-1}, \ldots, V_1 >$

b. **b.** 证明：在上述操作方式下，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，则在 $\lceil |V|/2 \rceil$ 遍松弛操作后，对于所有的结点 $v \in V$，有 $v.d = \delta(s, v)$。

- 在每一轮松弛操作的时候，Yen优化算法对于两个子图分别处理，对于两个子图，分别按照其顶点的拓扑序列进行松弛，这样每轮操作都可以保证至少有两个点的distance值被修正，图G中一共有|V|个顶点，所以没有负权环路的情况下，最多需要[|V| / 2]遍松弛操作后，就可以得到正确解。

C. c. 上述算法是否改善了 Bellman-Ford 算法的渐近运行时间？

- 没有改善渐进运行时间

- 在进行Yen优化之后，外层循环可以只进行⌈|V| / 2⌉次，整体渐进时间复杂度为O(|V||E|)不变

REMARK. Note that when the distances of arcs are all positive, Dijkstra's algorithm is superior to the new algorithm in most cases. Therefore, the new algorithm, in general, should be applied only when the distances of some arcs in the network are negative.