

# Design and Analysis of Algorithms

## Part IV: Graph Algorithms

### Lecture 30: Single Source Shortest Path

Bellman-Ford

童咏昕

北京航空航天大学  
计算机学院



- 在算法课程第四部分“图算法”主题中，我们将主要聚焦于如下经典问题：

- Basic Concepts in Graph Algorithms (图算法的基本概念)
- Breadth-First Search (BFS, 广度优先搜索)
- Depth-First Search (DFS, 深度优先搜索)
- Cycle Detection (环路检测)
- Topological Sort (拓扑排序)
- Strongly Connected Components (强连通分量)
- Minimum Spanning Trees (最小生成树)
- Single Source Shortest Path (单源最短路径)**
- All-Pairs Shortest Paths (所有点对最短路径)
- Bipartite Graph Matching (二分图匹配)
- Maximum/Network Flows (最大流/网络流)



问题背景

算法思想

算法实例

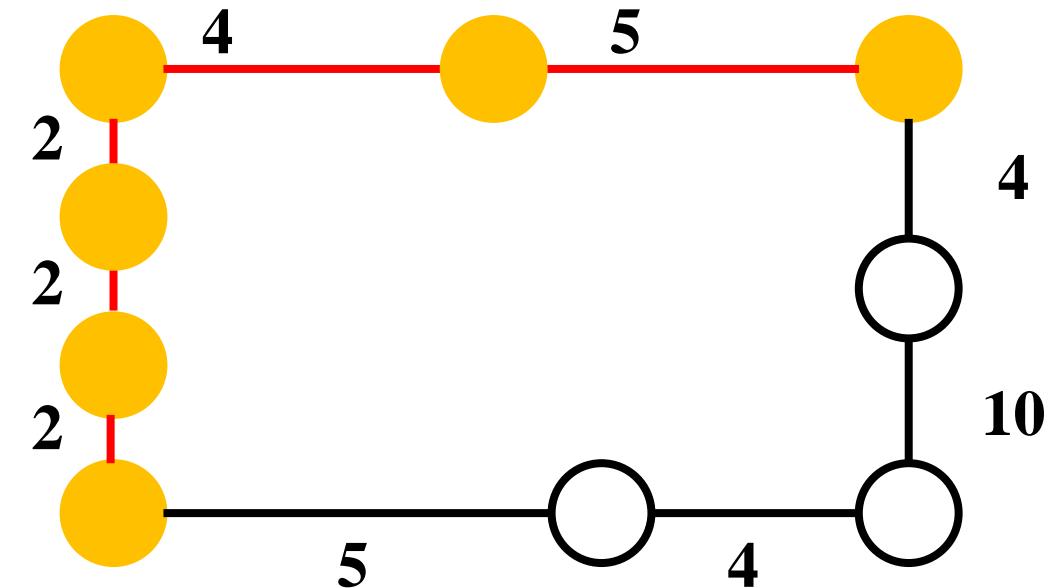
算法分析

算法性质

# 问题背景



- 从知春路到其他站点，如何安排路线？

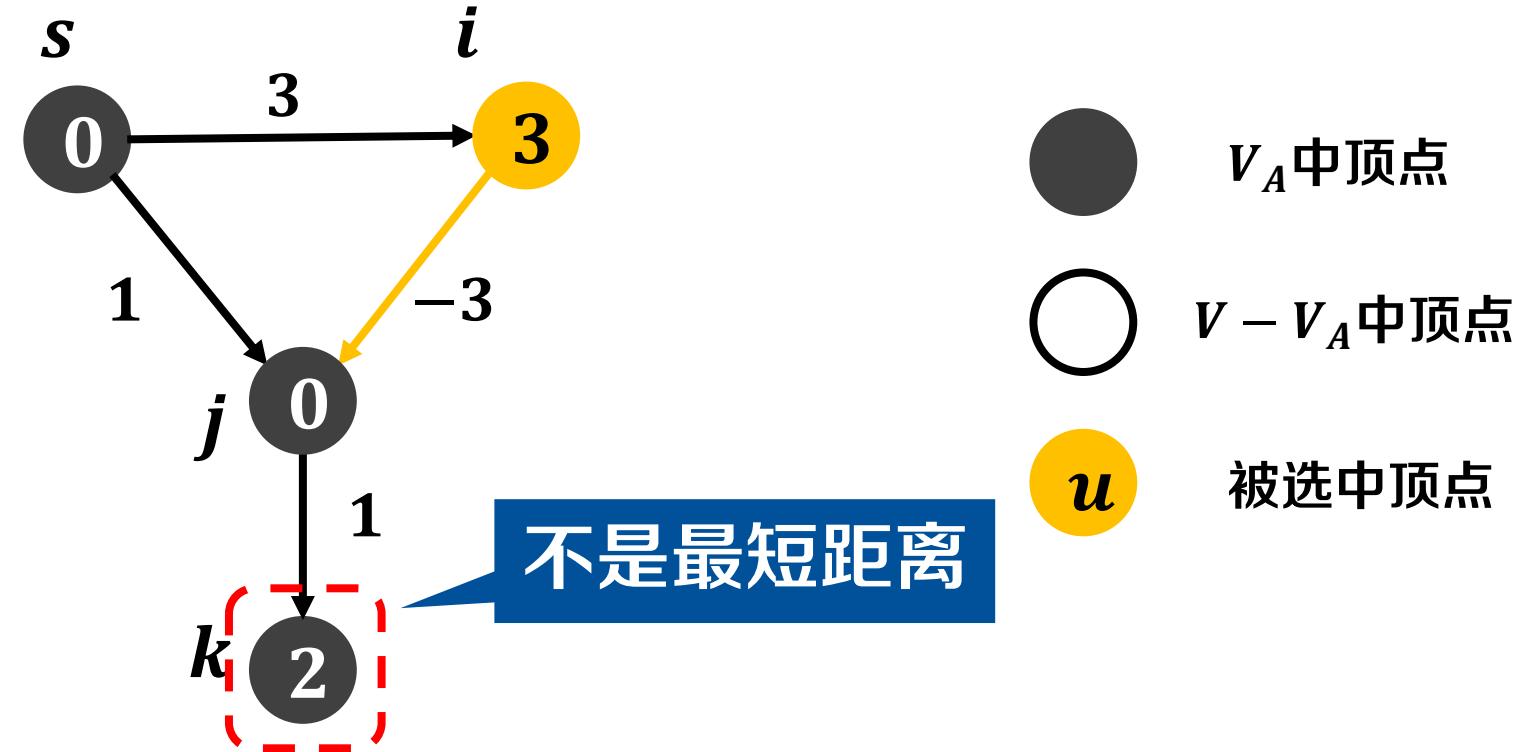


Dijkstra算法适用范围：边权为正的图

# 问题背景



- 图中存在**负权边**, Dijkstra算法不再适用

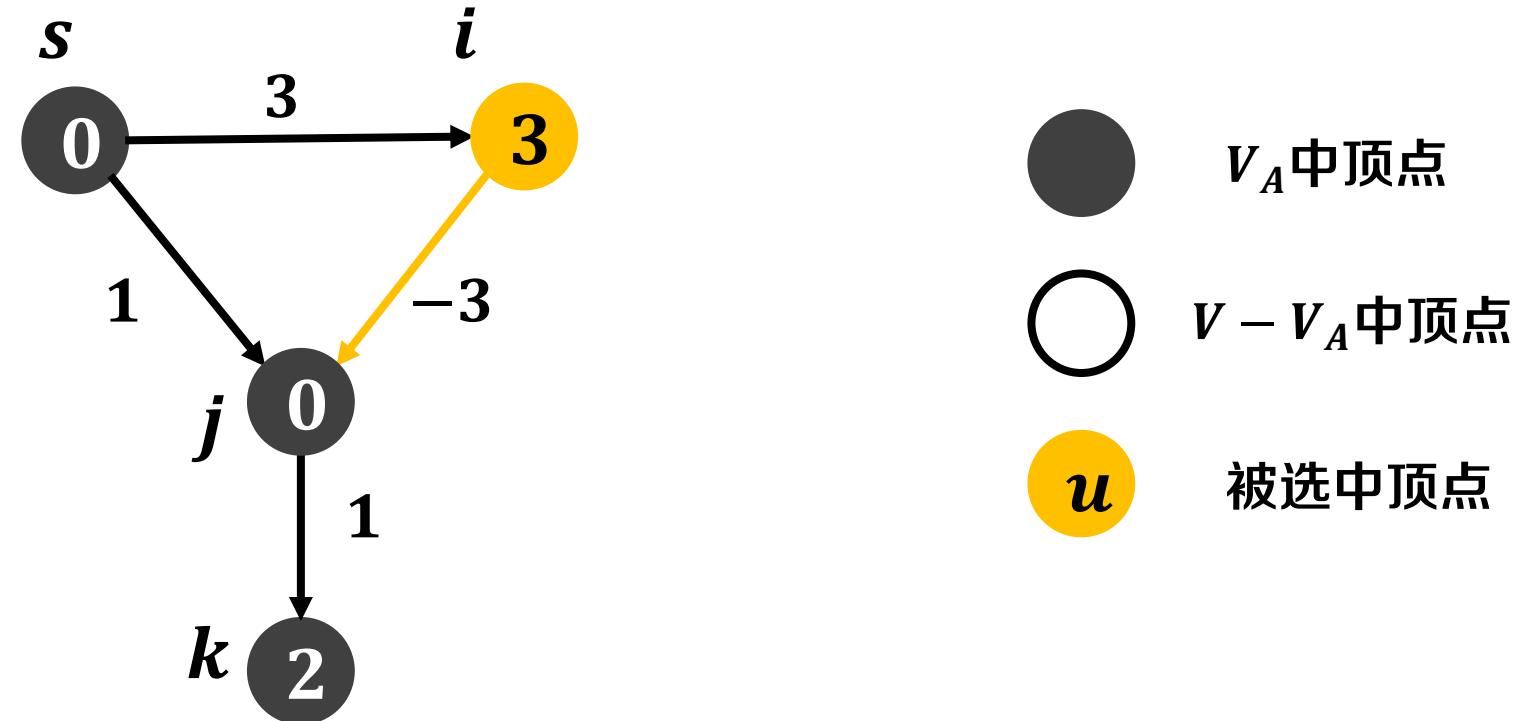


Dijkstra算法：到黑色顶点的最短路应该已经计算出

# 问题背景



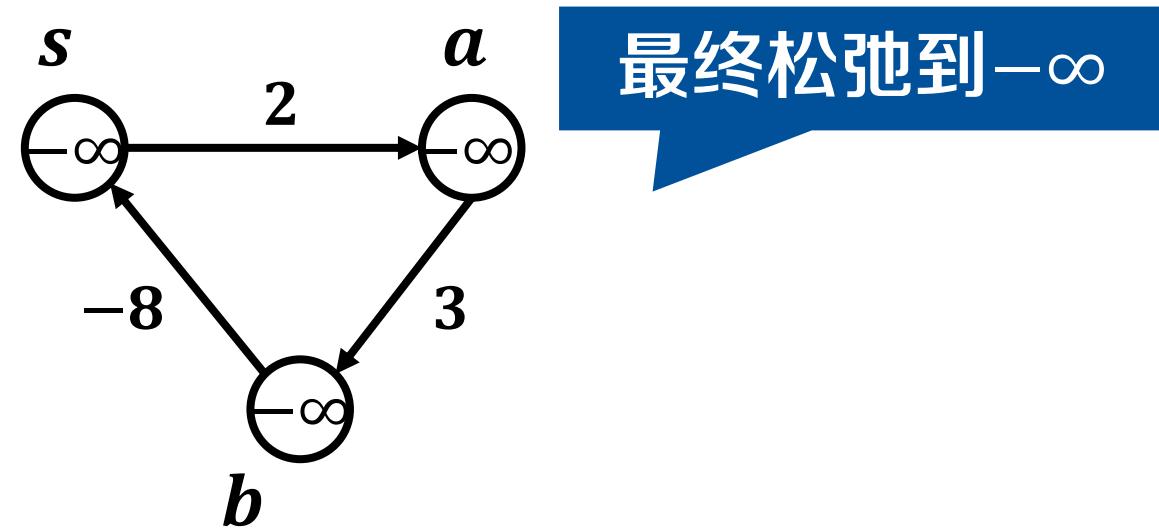
- 图中存在**负权边**, Dijkstra算法不再适用



问题：图中存在负权边时，是否存在单源最短路径？

# 问题背景

- 图中存在负权边时，是否存在单源最短路径？
  - 如果源点s可达**负环**，则难以定义最短路径



若源点s无可达负环，则存在源点s的单源最短路径



## 单源最短路径问题

### Single Source Shortest Paths Problem

#### 输入

- 带权图  $G = \langle V, E, W \rangle$
- 源点编号  $s$

#### 输出

- 源点  $s$  到所有其他顶点  $t$  的最短距离  $\delta(s, t)$  和最短路径  $\langle s, \dots, t \rangle$
- 或存在源点  $s$  可达的负环

**挑战1：图中存在负权边时，如何求解单源最短路径？**

**挑战2：图中存在负权边时，如何发现源点可达负环？**



问题背景

算法思想

算法实例

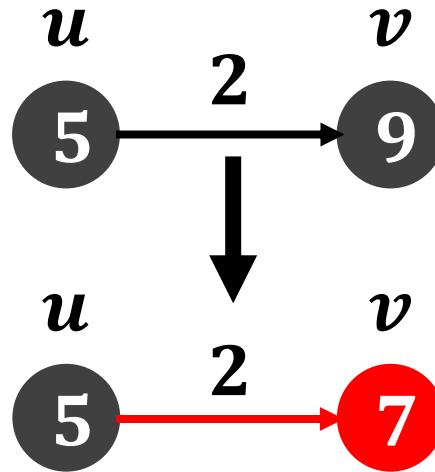
算法分析

算法性质

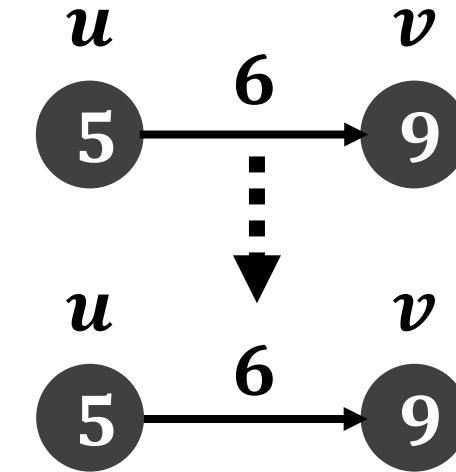
# 算法思想



- Dijkstra算法通过**松弛操作**迭代更新最短距离



松弛成功

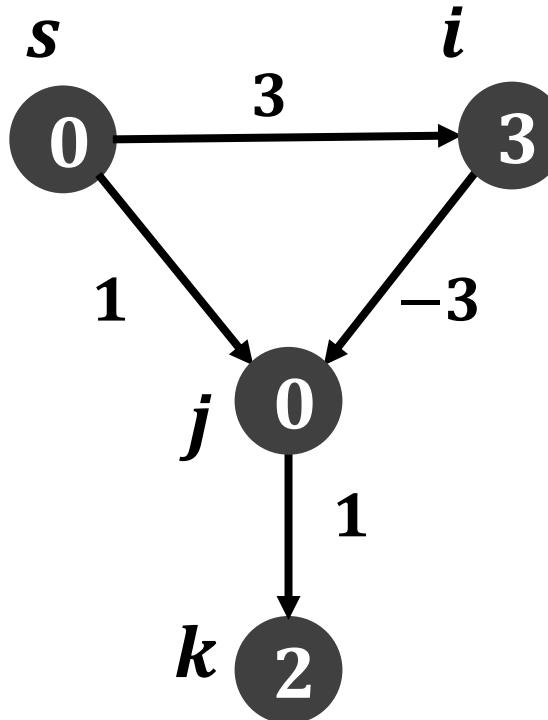


松弛失败

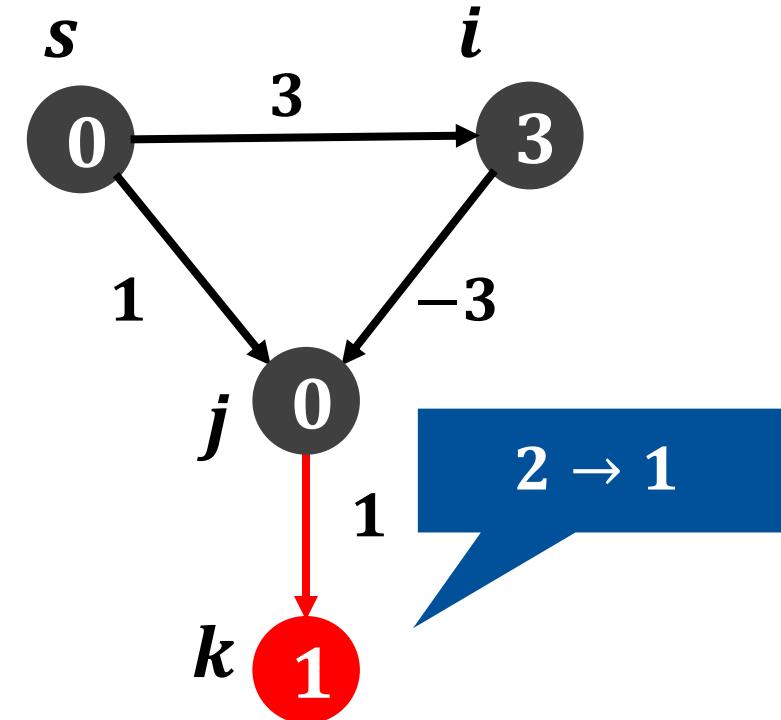
# 算法思想



- 存在负权边时，需要比Dijkstra算法**更多次数的松弛操作**



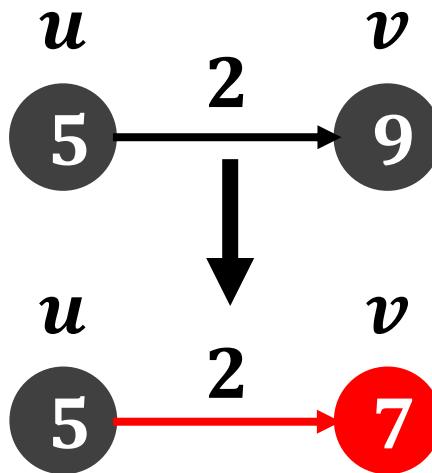
Dijkstra算法



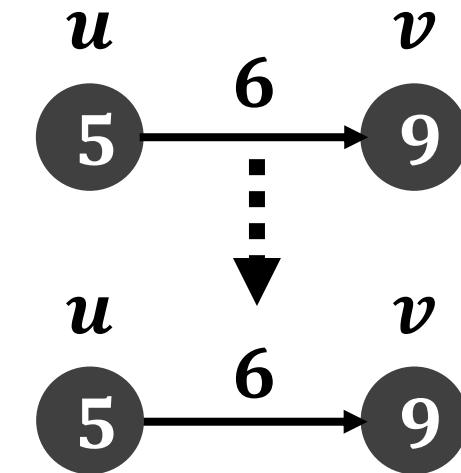
目标算法

问题：图中存在负权边时，如何利用松弛操作求解单源最短路？

- Bellman-Ford算法
  - 解决挑战1：图中存在负权边时，如何求解单源最短路径？
    - 每轮对所有边进行松弛，持续迭代 $|V| - 1$ 轮



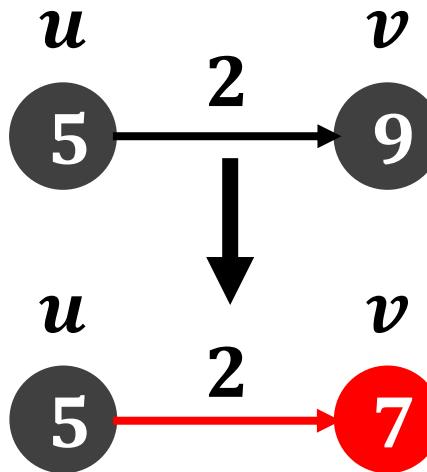
松弛成功



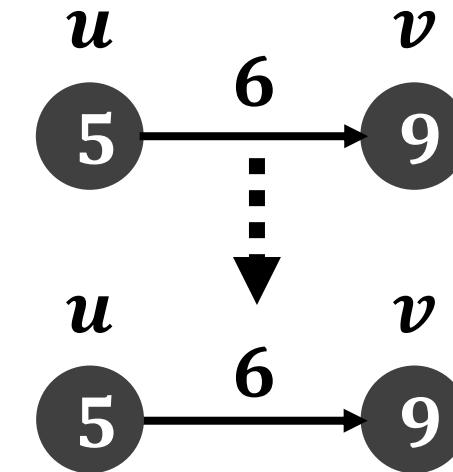
松弛失败

## Bellman-Ford算法

- 解决挑战1：图中存在负权边时，如何求解单源最短路径？
  - 每轮对所有边进行松弛，持续迭代 $|V| - 1$ 轮
- 解决挑战2：图中存在负权边时，如何发现源点可达负环？
  - 若第 $|V|$ 轮仍松弛成功，存在源点 $s$ 可达的负环



松弛成功



松弛失败



问题背景

算法思想

算法实例

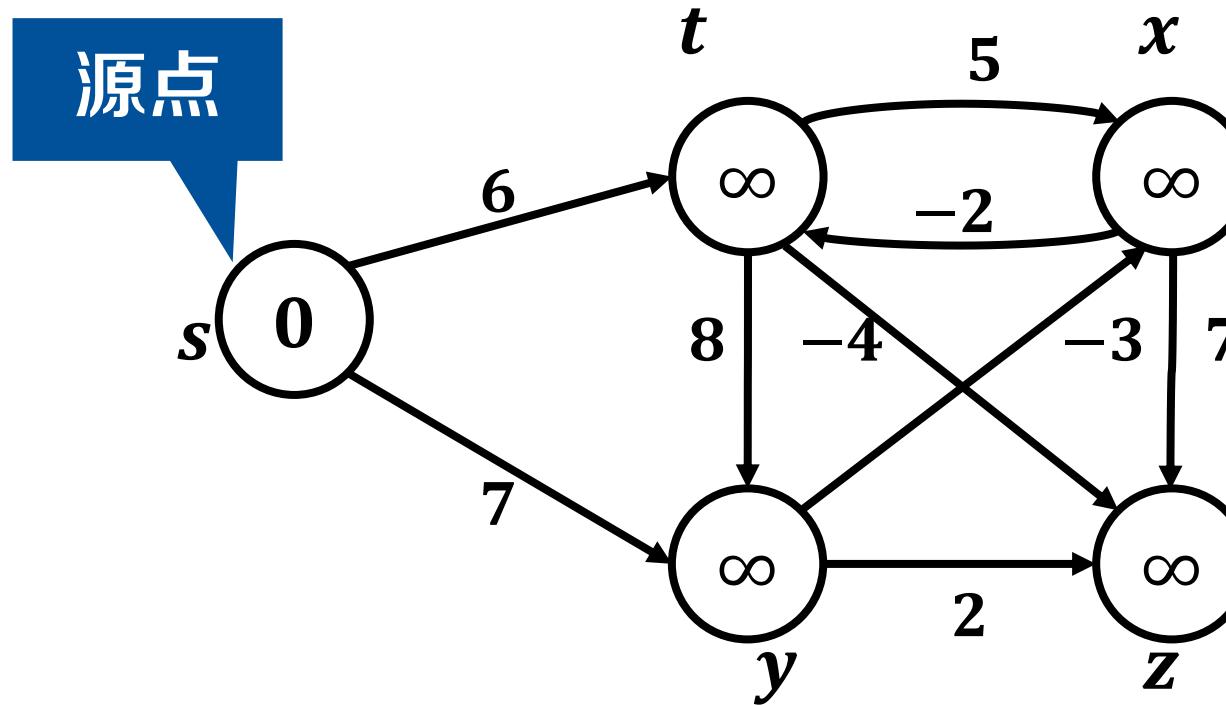
算法分析

算法性质

# 算法实例



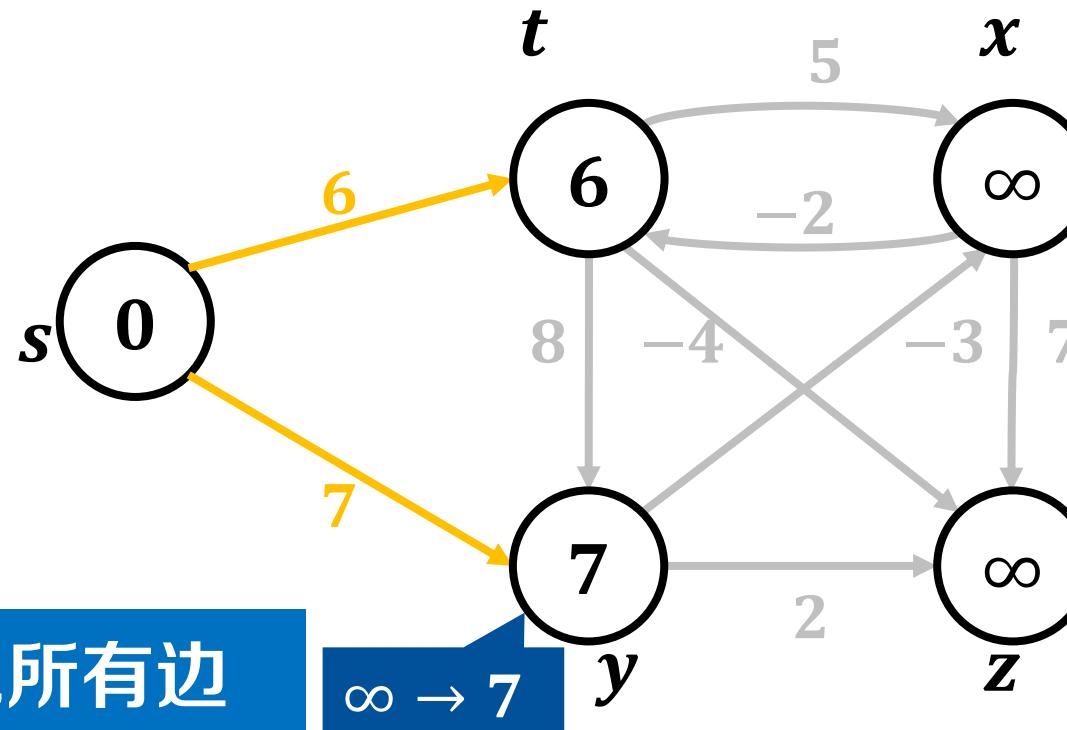
$V$	$s$	$t$	$x$	$y$	$z$
$pred$	N	N	N	N	N
$dist$	0	$\infty$	$\infty$	$\infty$	$\infty$



# 算法实例



$V$	$s$	$t$	$x$	$y$	$z$
$pred$	N	$s$	N	$s$	N
$dist$	0	6	$\infty$	7	$\infty$



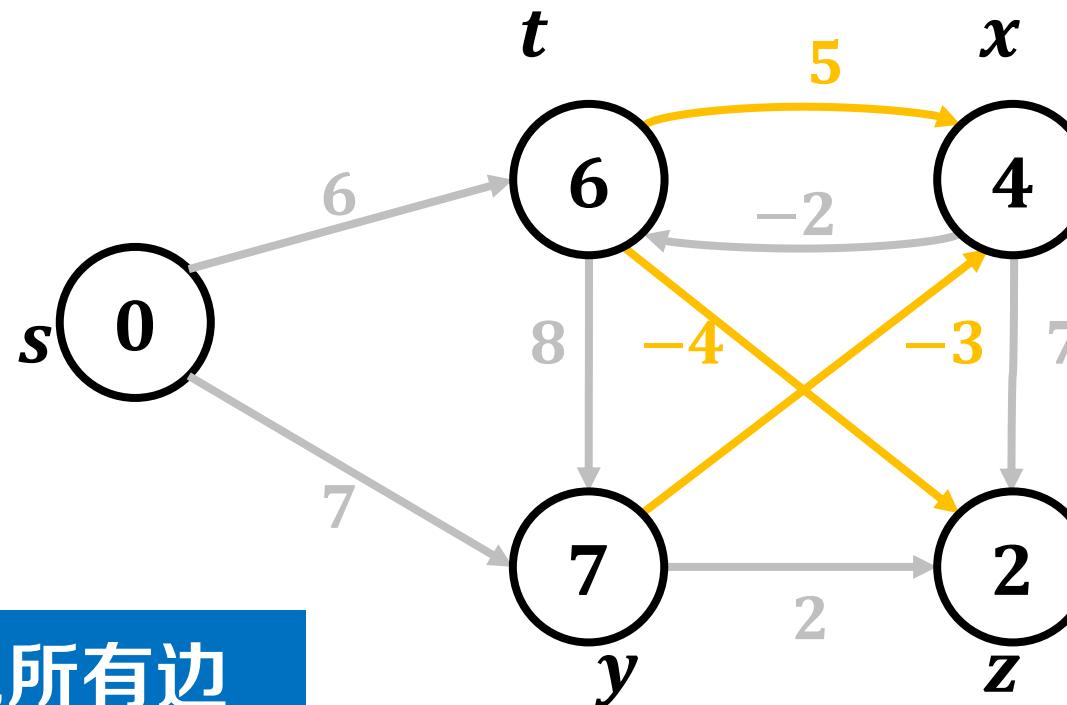
松弛顺序：  $(x, z), (t, x), (x, t), (t, z), (y, x), (y, z), (t, y), (s, t), (s, y)$

- 松弛失败
- 松弛成功

# 算法实例



$V$	$s$	$t$	$x$	$y$	$z$
$pred$	N	$s$	$y$	$s$	$t$
$dist$	0	6	4	7	2



第2轮：松弛所有边

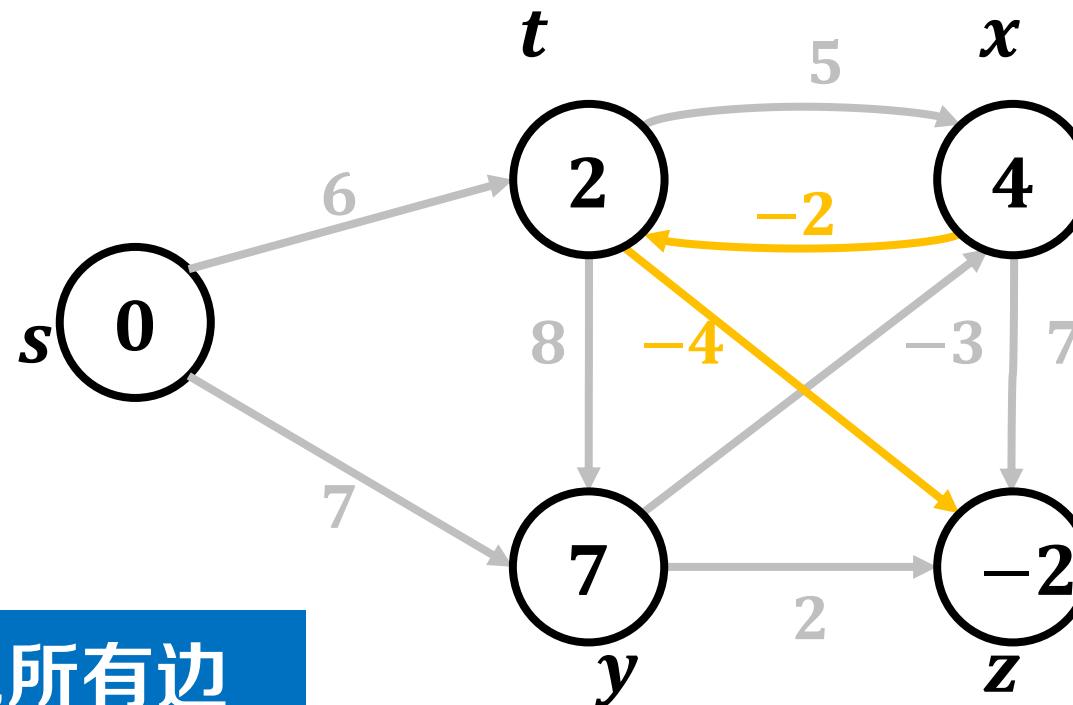
松弛顺序：  $(x, z)$ ,  $(t, x)$ ,  
 $(x, t)$ ,  $(t, z)$ ,  $(y, x)$ ,  $(y, z)$ ,  
 $(t, y)$ ,  $(s, t)$ ,  $(s, y)$

- 松弛失败
- 松弛成功

# 算法实例



$V$	$s$	$t$	$x$	$y$	$z$
$pred$	N	$x$	$y$	$s$	$t$
$dist$	0	2	4	7	-2



松弛顺序:  $(x, z)$ ,  $(t, x)$ ,  
 $(x, t)$ ,  $(t, z)$ ,  $(y, x)$ ,  $(y, z)$ ,  
 $(t, y)$ ,  $(s, t)$ ,  $(s, y)$

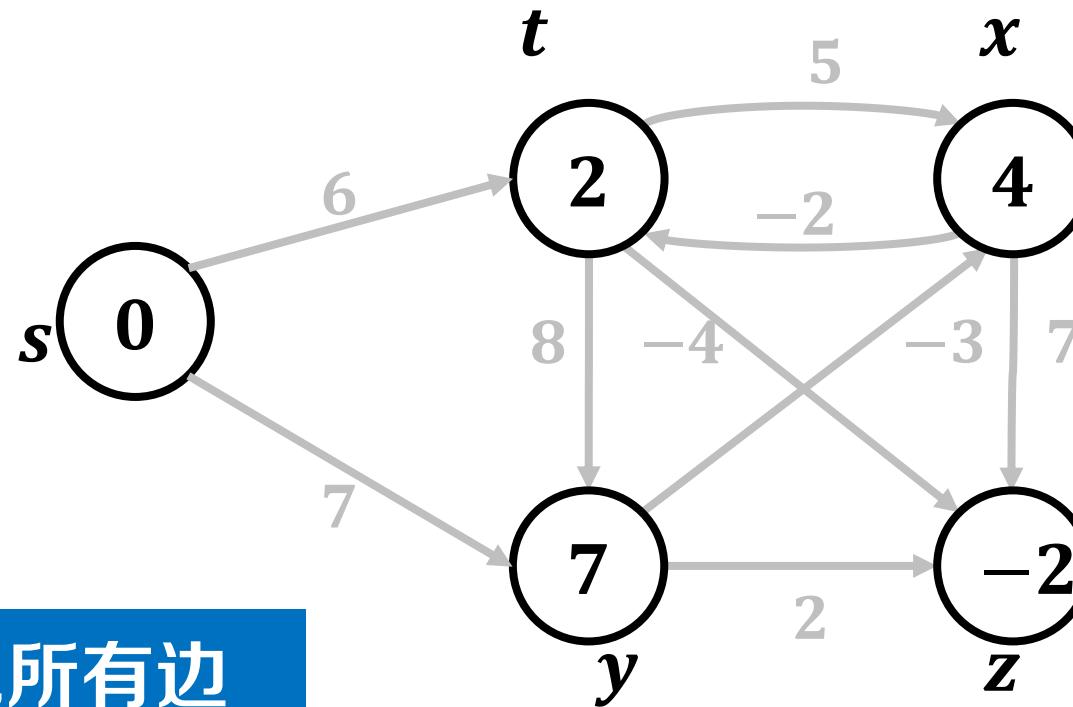
- 松弛失败  
→ 松弛成功

第3轮：松弛所有边

# 算法实例



$V$	$s$	$t$	$x$	$y$	$z$
$pred$	N	$x$	$y$	$s$	$t$
$dist$	0	2	4	7	-2



松弛顺序:  $(x, z), (t, x), (x, t), (t, z), (y, x), (y, z), (t, y), (s, t), (s, y)$

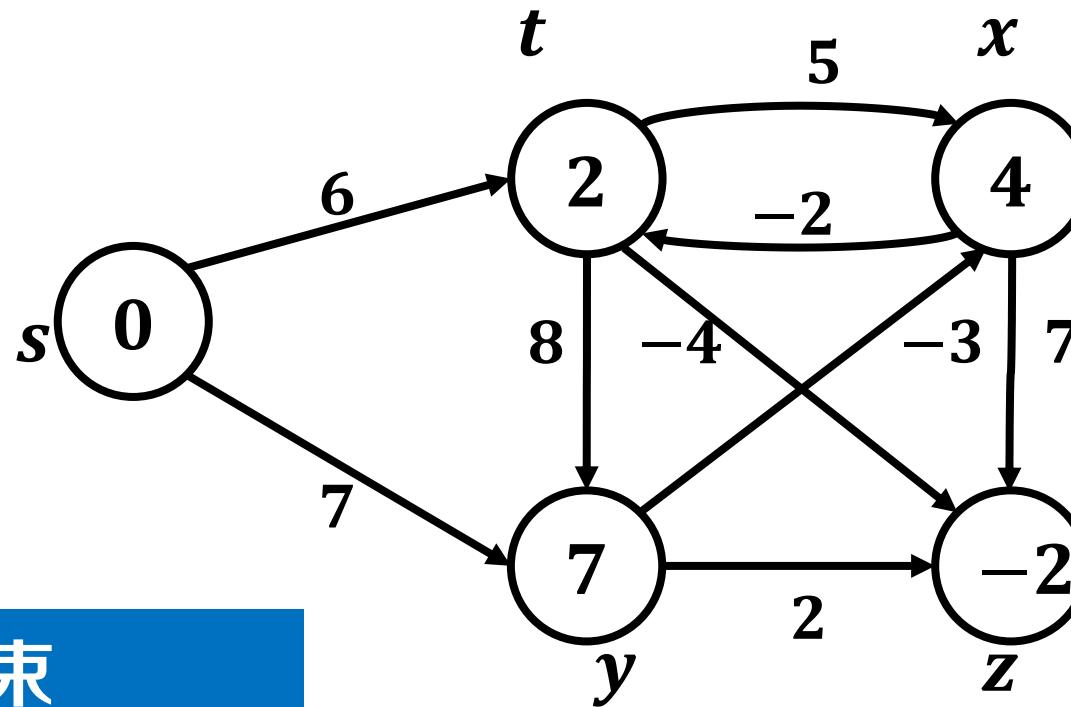
- 松弛失败  
→ 松弛成功

第4轮: 松弛所有边

# 算法实例



$V$	$s$	$t$	$x$	$y$	$z$
$pred$	N	$x$	$y$	$s$	$t$
$dist$	0	2	4	7	-2



松弛顺序:  $(x, z), (t, x), (x, t), (t, z), (y, x), (y, z), (t, y), (s, t), (s, y)$

- 松弛失败
- 松弛成功



问题背景

算法思想

算法实例

算法分析

算法性质



# 伪代码

- Bellman-Ford( $G, s$ )

输入: 图 $G = \langle V, E, W \rangle$ , 源点 $s$

输出: 单源最短路径 $P$

新建一维数组 $dist[1..|V|]$ ,  $pred[1..|V|]$

//初始化

for  $u \in V$  do

$dist[u] \leftarrow \infty$

$pred[u] \leftarrow NULL$

end

$dist[s] \leftarrow 0$



# 伪代码

- Bellman-Ford( $G, s$ )

```
//执行单源最短路径算法
for  $i \leftarrow 1$  to  $|V| - 1$  do
    for  $(u, v) \in E$  do
        if  $dist[u] + w(u, v) < dist[v]$  then
             $dist[v] \leftarrow dist[u] + w(u, v)$ 
             $pred[v] \leftarrow u$ 
        end
    end
end
for  $(u, v) \in E$  do
    if  $dist[u] + w(u, v) < dist[v]$  then
        print 存在负环
        break
    end
end
```



# 时间复杂度分析

- Bellman-Ford( $G, s$ )

输入: 图  $G = \langle V, E, W \rangle$ , 源点  $s$

输出: 单源最短路径  $P$

新建一维数组  $dist[1..|V|]$ ,  $pred[1..|V|]$

//初始化

```
for  $u \in V$  do  
     $dist[u] \leftarrow \infty$   
     $pred[u] \leftarrow NULL$   
end  
 $dist[s] \leftarrow 0$ 
```

$\left. \right\} O(|V|)$



# 时间复杂度分析

## ● Bellman-Ford( $G, s$ )

```
//执行单源最短路径算法
for  $i \leftarrow 1$  to  $|V| - 1$  do
    for  $(u, v) \in E$  do
        if  $dist[u] + w(u, v) < dist[v]$  then
             $dist[v] \leftarrow dist[u] + w(u, v)$ 
             $pred[v] \leftarrow u$ 
        end
    end
end
for  $(u, v) \in E$  do
    if  $dist[u] + w(u, v) < dist[v]$  then
        print 存在负环
        break
    end
end
```

时间复杂度 $O(|E| \cdot |V|)$



问题背景

算法思想

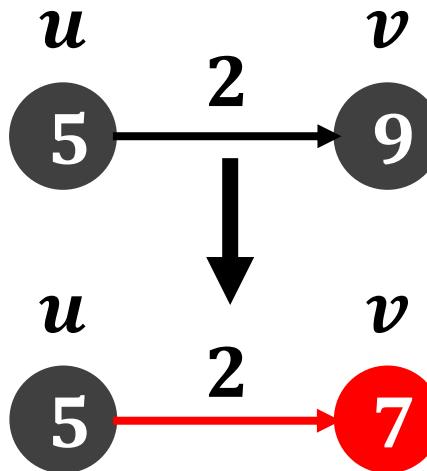
算法实例

算法分析

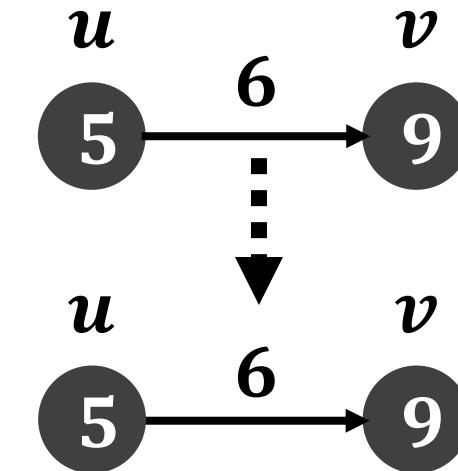
算法性质

- Bellman-Ford算法

- 挑战1：图中存在负权边时，如何求解单源最短路径?
  - 解决方案：每轮对所有边进行松弛，持续迭代 $|V| - 1$ 轮
- 挑战2：图中存在负权边时，如何发现源点可达负环?
  - 解决方案：若第 $|V|$ 轮仍松弛成功，存在源点 $s$ 可达的负环



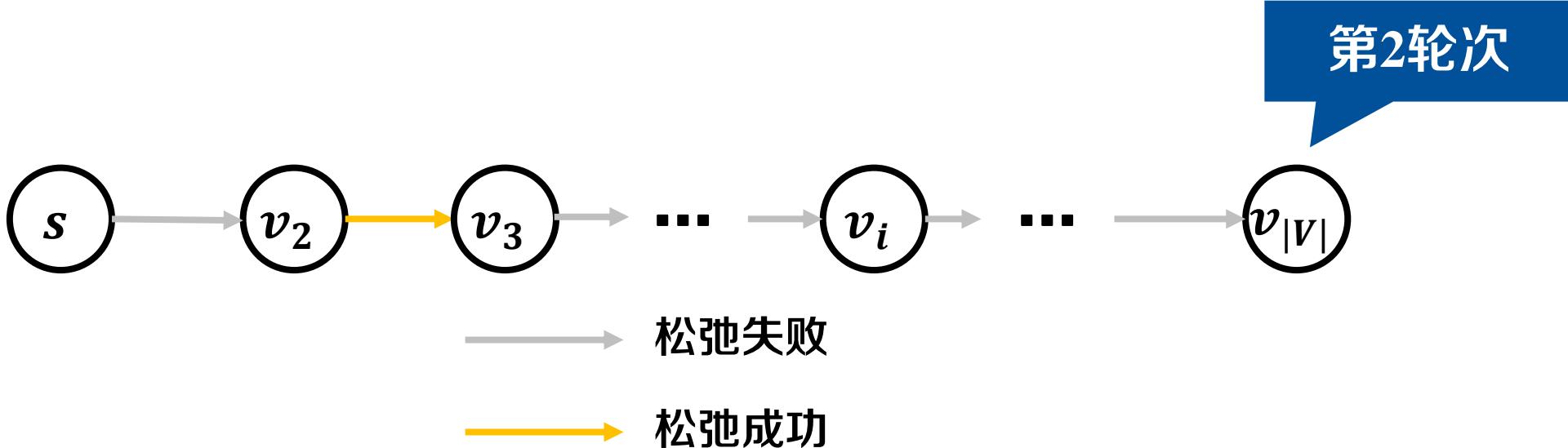
松弛成功



松弛失败

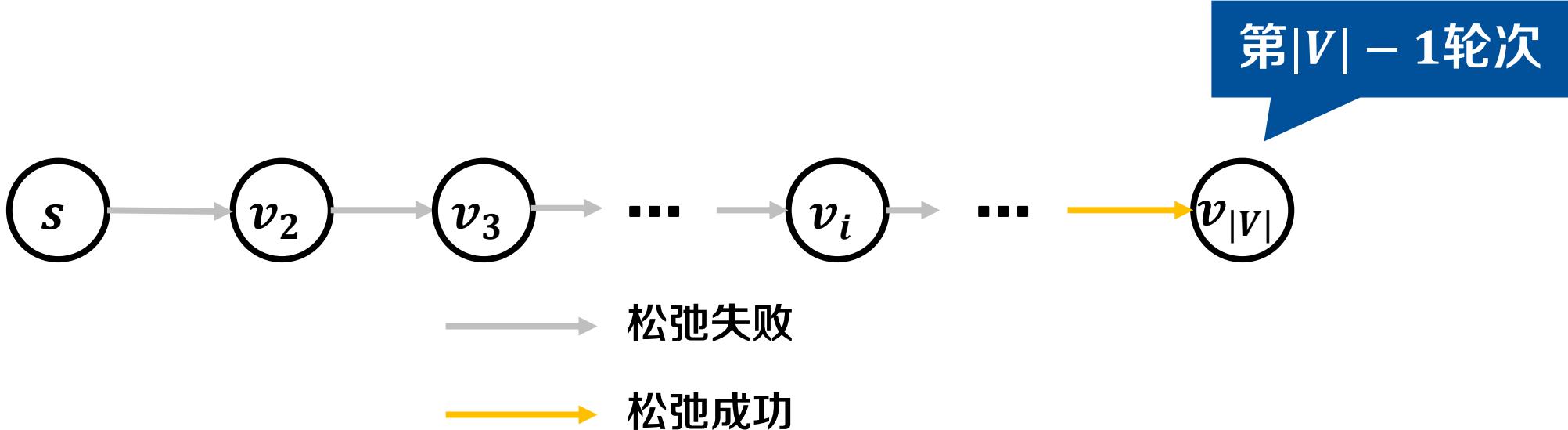
# 算法正确性

- 挑战1：图中存在负权边时，如何求解单源最短路径？
  - 解决方案：每轮对所有边进行松弛，持续迭代 $|V| - 1$ 轮
- 最坏情况
  - 非环路的路径 $< s, v_2, v_3, \dots, v_{|V|} >$ 至多经过 $|V| - 1$ 条边



# 算法正确性

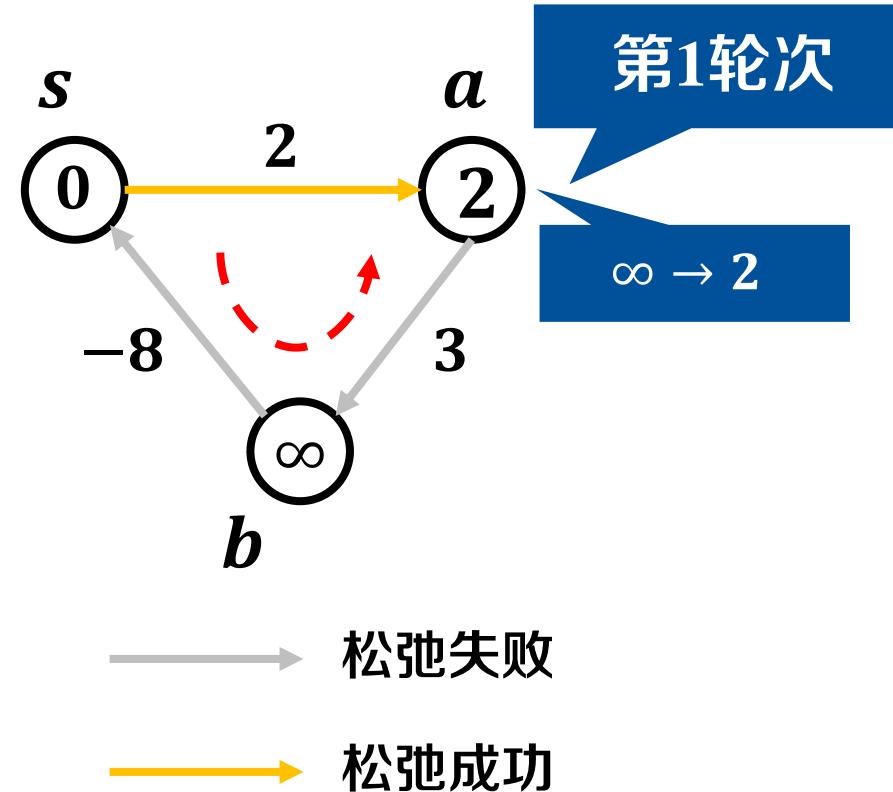
- 挑战1：图中存在负权边时，如何求解单源最短路径？
  - 解决方案：每轮对所有边进行松弛，持续迭代 $|V| - 1$ 轮
- 最坏情况
  - 非环路的路径 $< s, v_2, v_3, \dots, v_{|V|} >$ 至多经过 $|V| - 1$ 条边



最坏情况下进行 $|V| - 1$ 轮松弛操作，可以保证求得单源最短路径

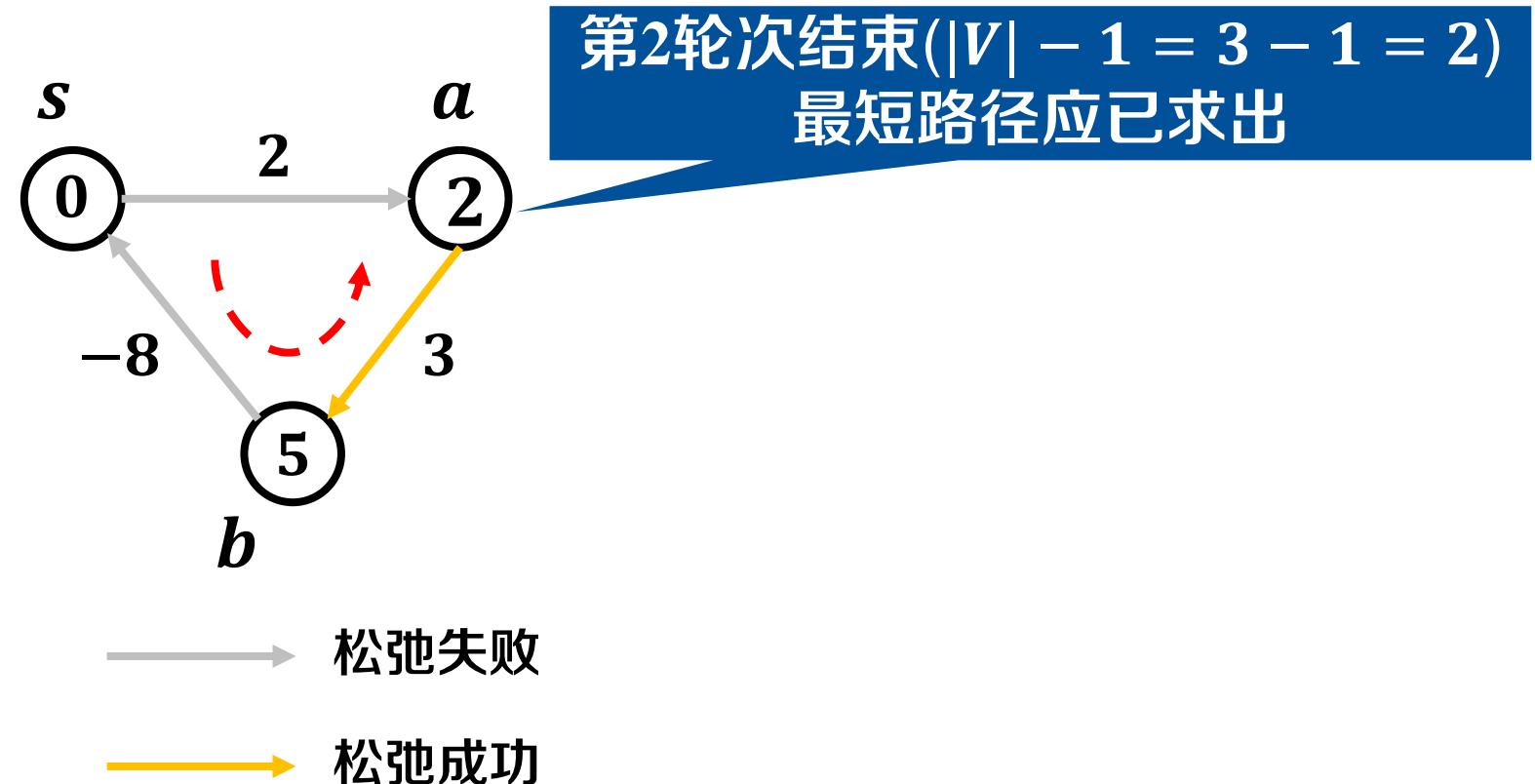
# 算法正确性

- 挑战2：图中存在负权边时，如何发现源点可达负环？
  - 解决方案：若第 $|V|$ 轮仍松弛成功，存在源点 $s$ 可达的负环
- 若源点 $s$ 可达负环，可松弛成功无限次



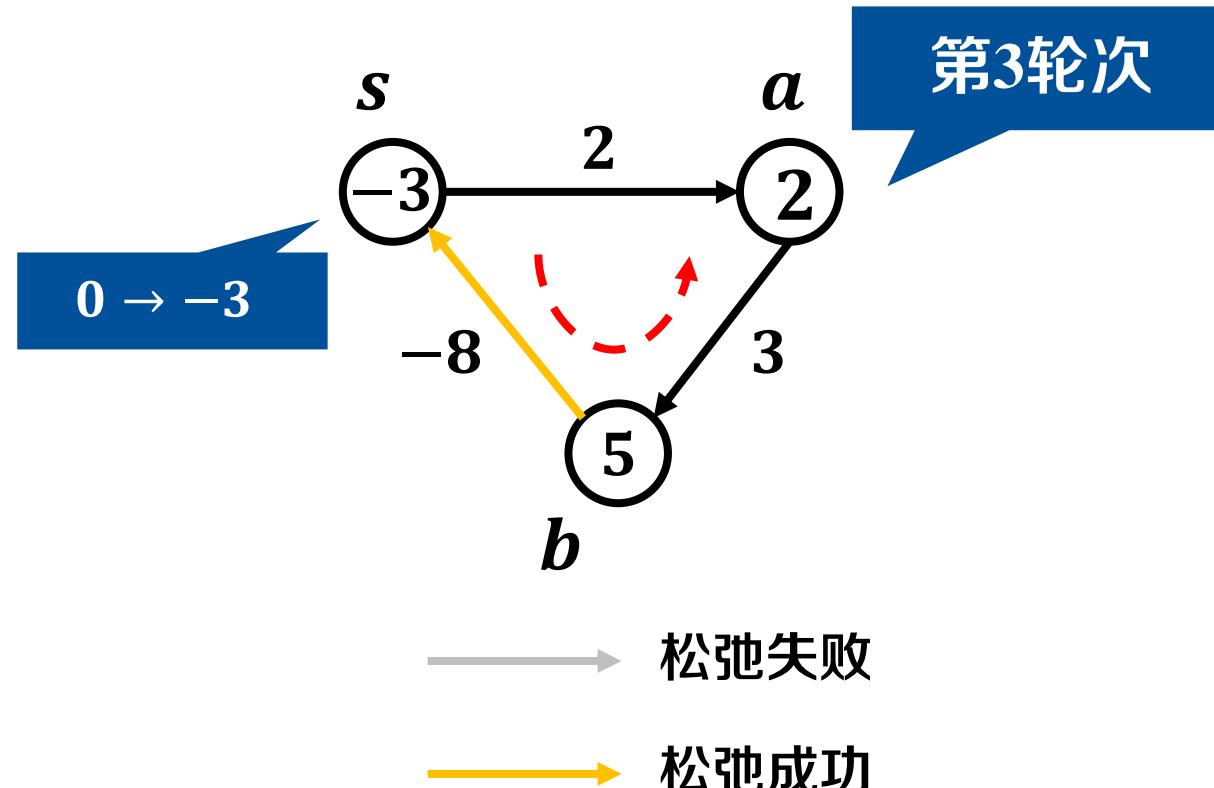
# 算法正确性

- 挑战2：图中存在负权边时，如何发现源点可达负环？
  - 解决方案：若第 $|V|$ 轮仍松弛成功，存在源点 $s$ 可达的负环
- 若源点 $s$ 可达负环，可松弛成功无限次



# 算法正确性

- 挑战2：图中存在负权边时，如何发现源点可达负环？
  - 解决方案：若第 $|V|$ 轮仍松弛成功，存在源点 $s$ 可达的负环
- 若源点 $s$ 可达负环，可松弛成功无限次



第 $|V|$ 轮仍松弛成功的原因：存在源点可达的负环

# 小结



	广度优先搜索	Dijkstra算法	Bellman-Ford算法
适用范围	无权图	带权图 (所有边权为正)	带权图
松弛次数	--	$ E $ 次	$ V  \cdot  E $ 次
数据结构	队列	优先队列	--
运行时间	$O( V  +  E )$	$O( E  \cdot \log V )$	$O( E  \cdot  V )$



---

謝謝

