# 路网最优路径算法概述

By: 欧阳俊(司谦)
Email: jun.ouyang@alibaba-inc.cn
2015-09-22

# 内容提纲

- 背景
- 社交网
- 路径算法—经典Dijkstra
- 路径算法—优化策略
- 路径算法—性能评价
- 路径算法—历史与现状
- 未来挑战

# 路径算法—标题关键字

**路网特点**

1 每个节点的度数不高
2 边有等级的划分
3 路网更新相对缓慢

**最优定义**

最优的相对性:
路人甲更偏好高速路，路人乙更偏好畅通路，而路人丙更偏好沿途风景好的路

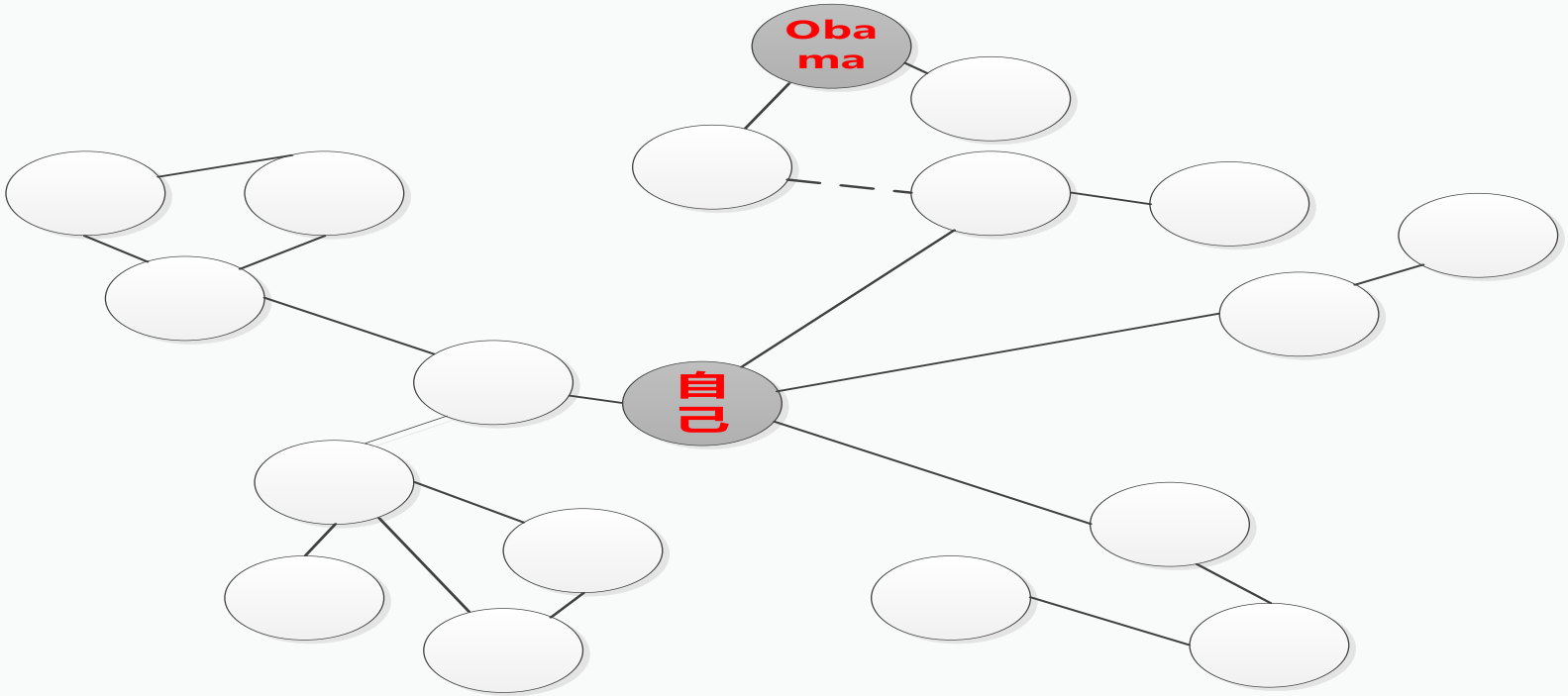
**介绍内容**

道路代价模型 → 决定质量的优劣
**路径算法** → 决定响应的快慢

# 路径算法—社交网　1/3

## 问题：

现有紧急事情，我想联系上奥巴马，捷径呢？



# 路径算法—社交网　2/3

## 方法：

1 首先我会直接联系家人、朋友或同事，有没认识奥巴马的？

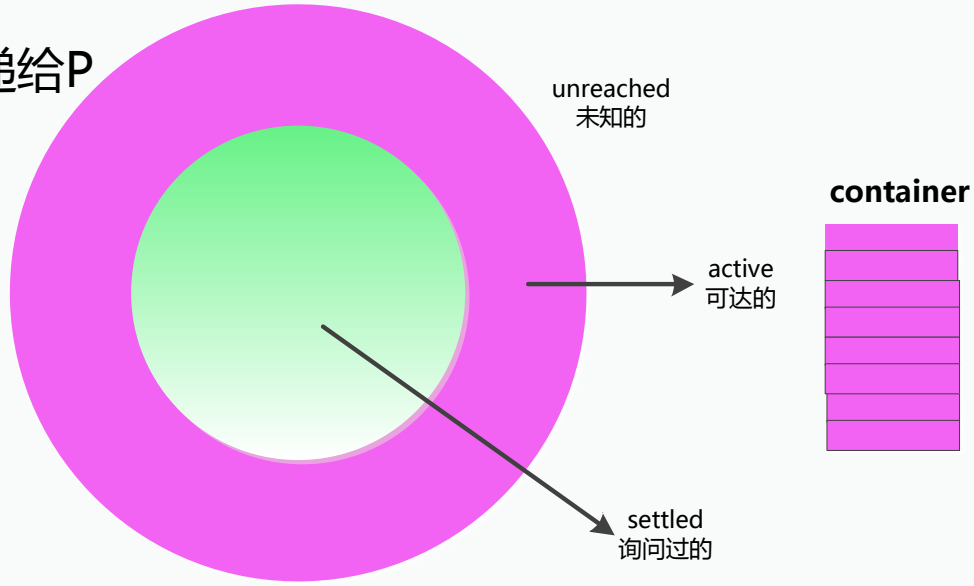
2 然后我拜托家人、朋友联系家人的朋友或朋友的朋友，有没认识奥巴马的？

3 。。。一直循环把问题传递、抛给我间接可以联系的人，直到联系奥巴马

# 路径算法—社交网　3/3

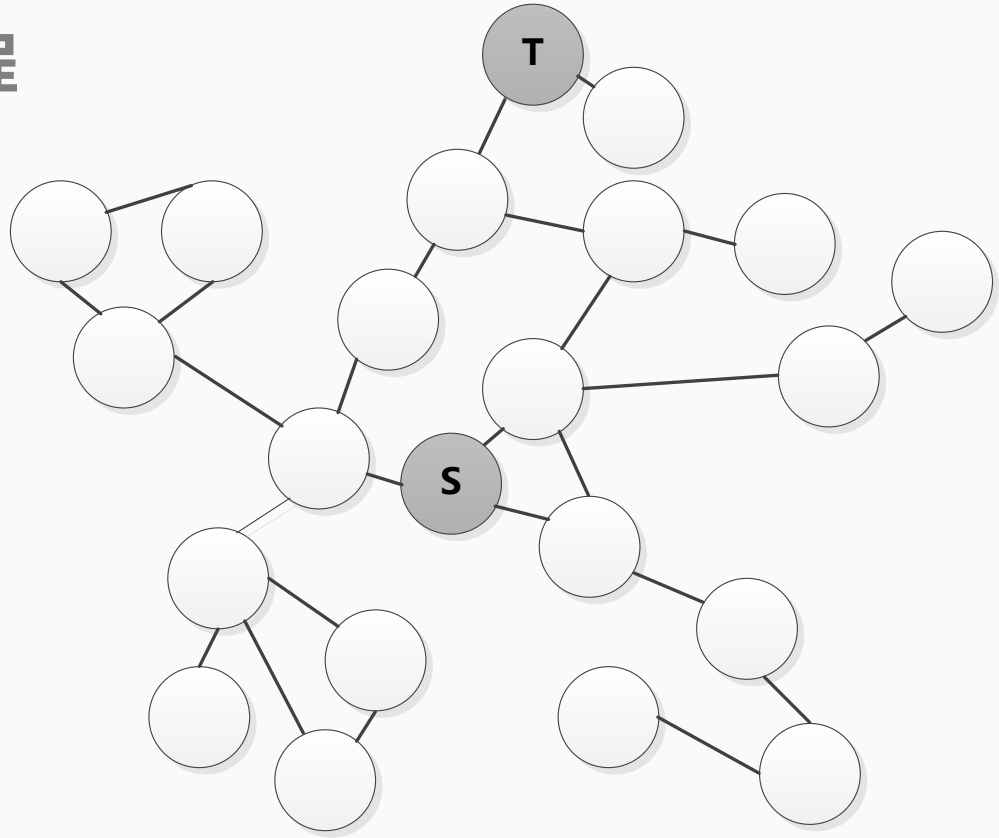
## 步骤(**节点涂色**)：

```
foreach(拿取当前红圈里面最亲近的节点P)
{
    1 节点P是否是奥巴马？
    2 把节点P标记为settled
    3 如果是，问题解决退出
    4 如果不是，问题直接传递给P
}
```



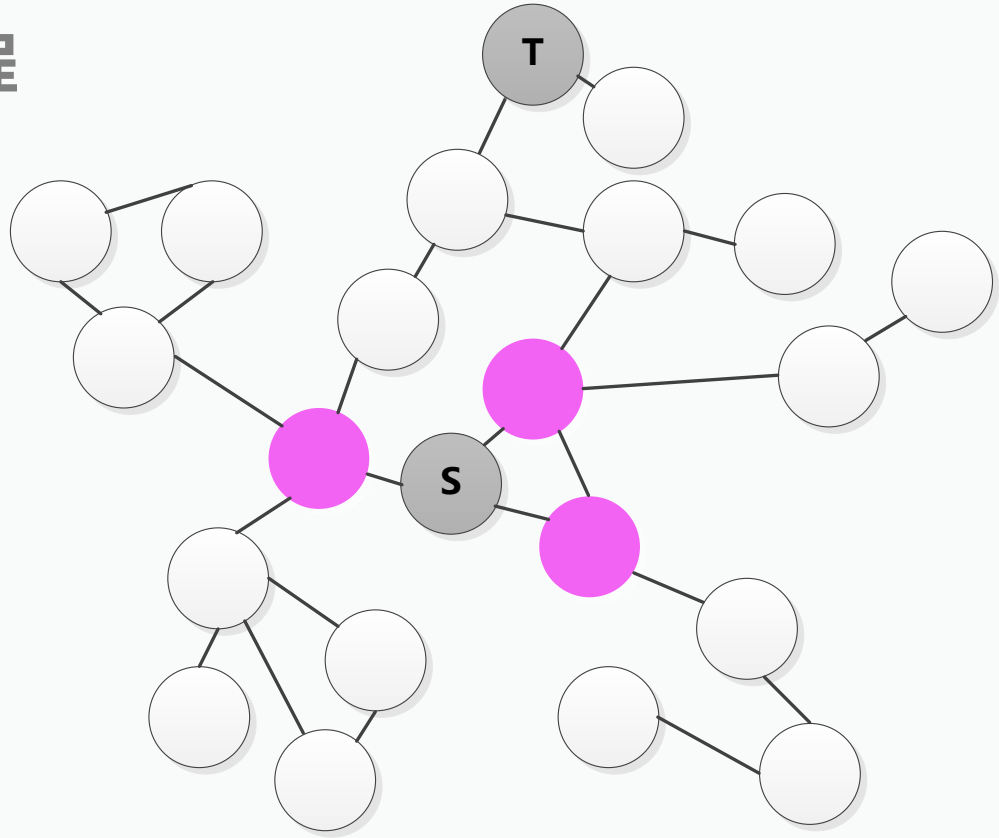
# 路径算法—Dijkstra 0/14

## 搜索过程



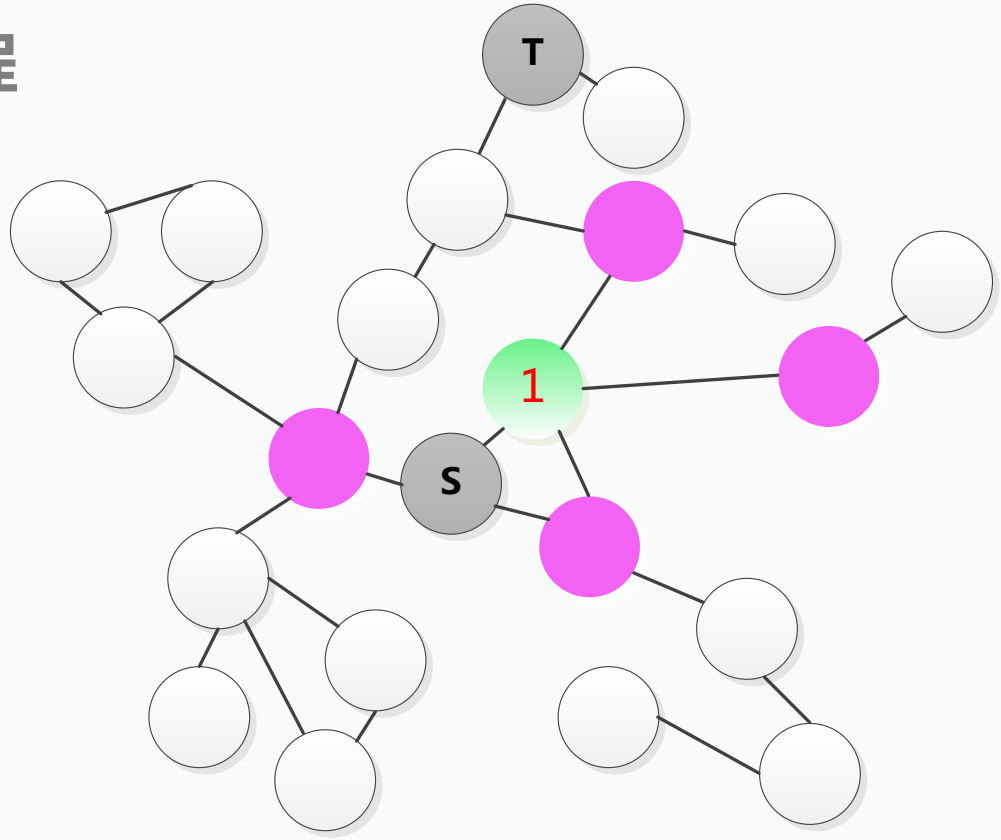
# 路径算法—Dijkstra 1/14

## 搜索过程



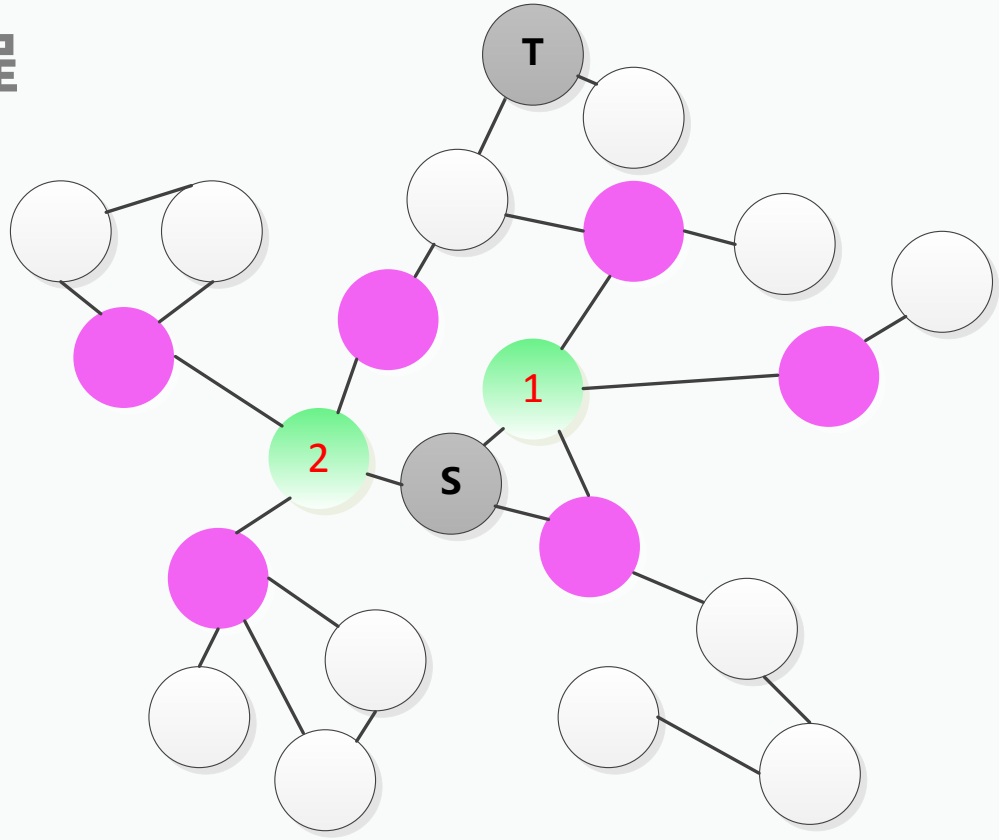
# 路径算法—Dijkstra 2/14

## 搜索过程



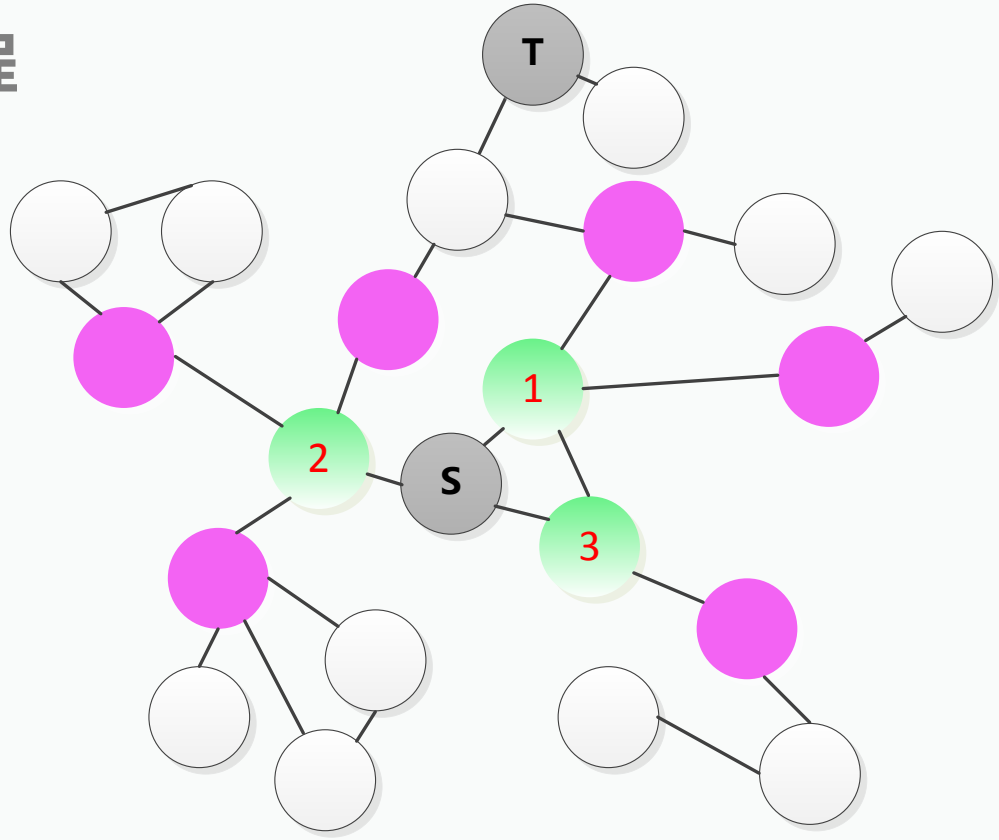
# 路径算法—Dijkstra 3/14

## 搜索过程



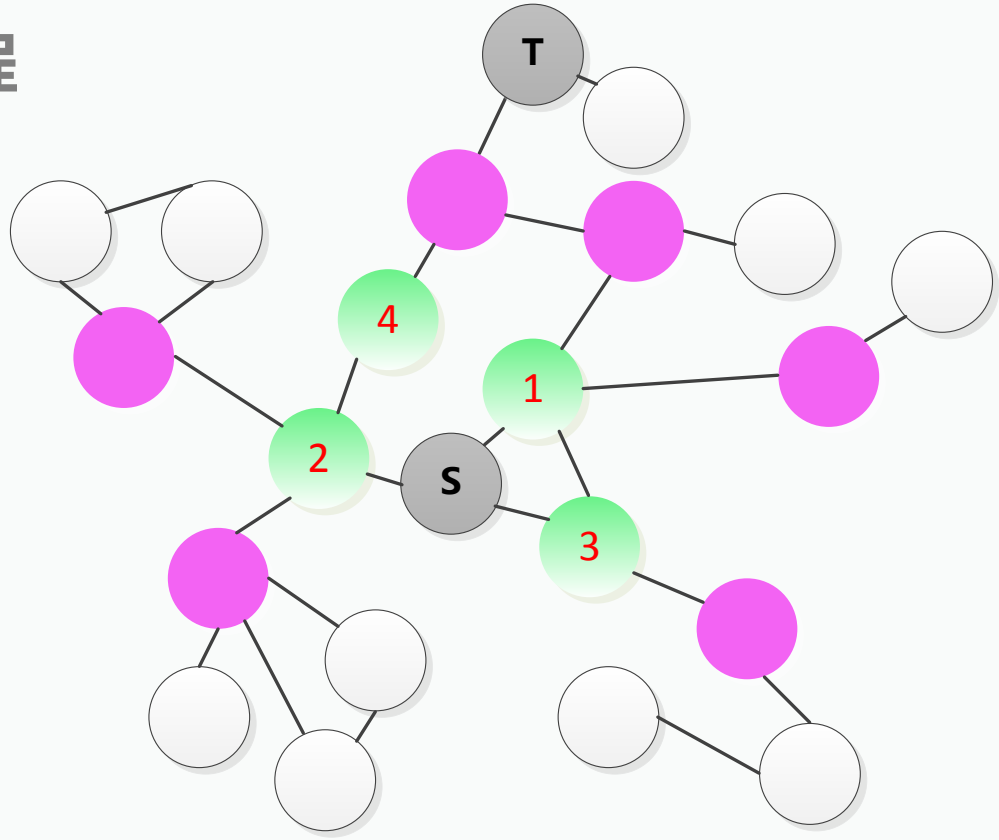
# 路径算法—Dijkstra 4/14

## 搜索过程



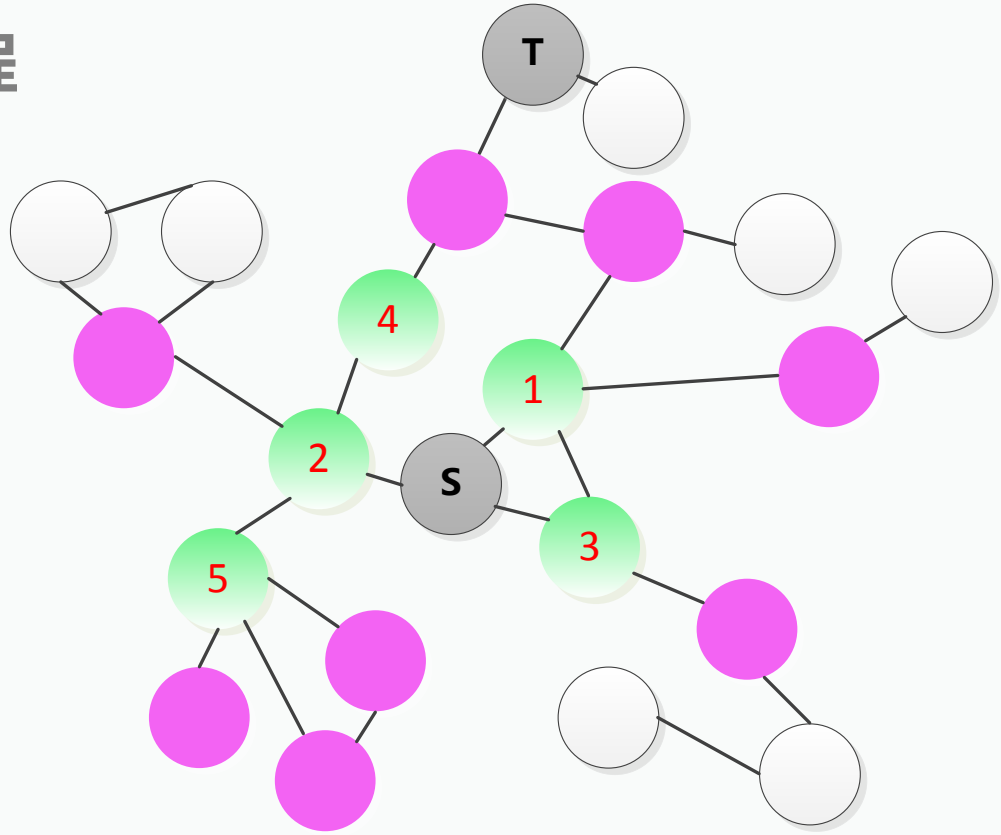
# 路径算法—Dijkstra 5/14

## 搜索过程



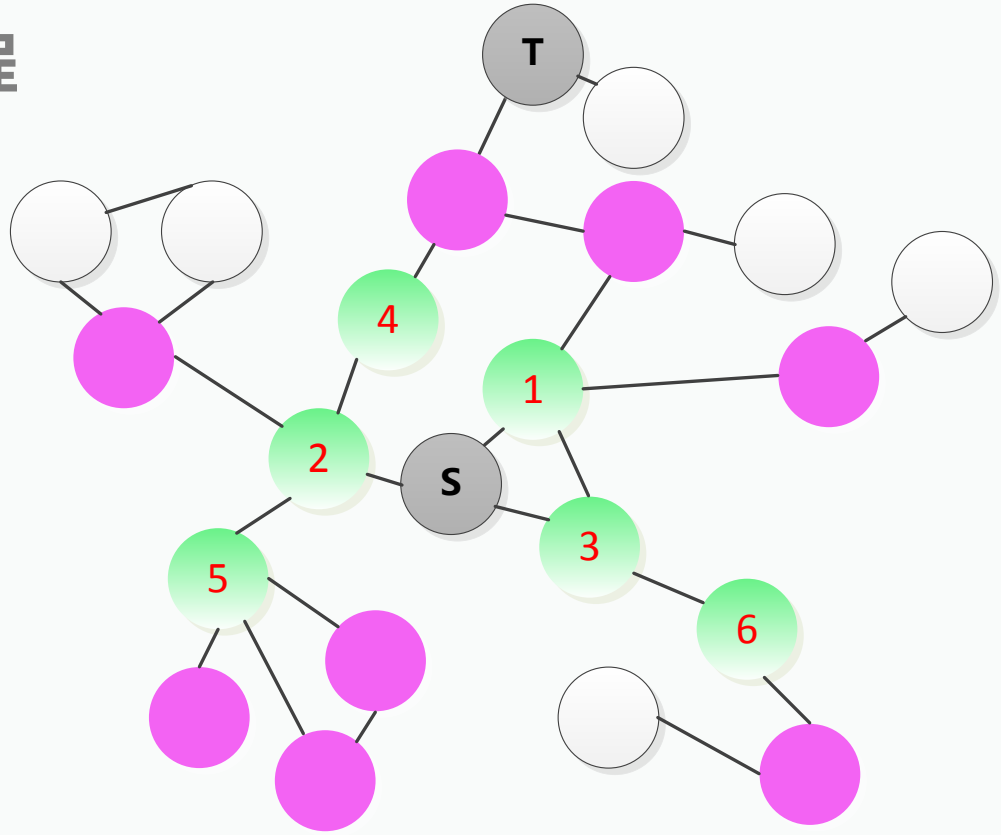
# 路径算法—Dijkstra 6/14

## 搜索过程



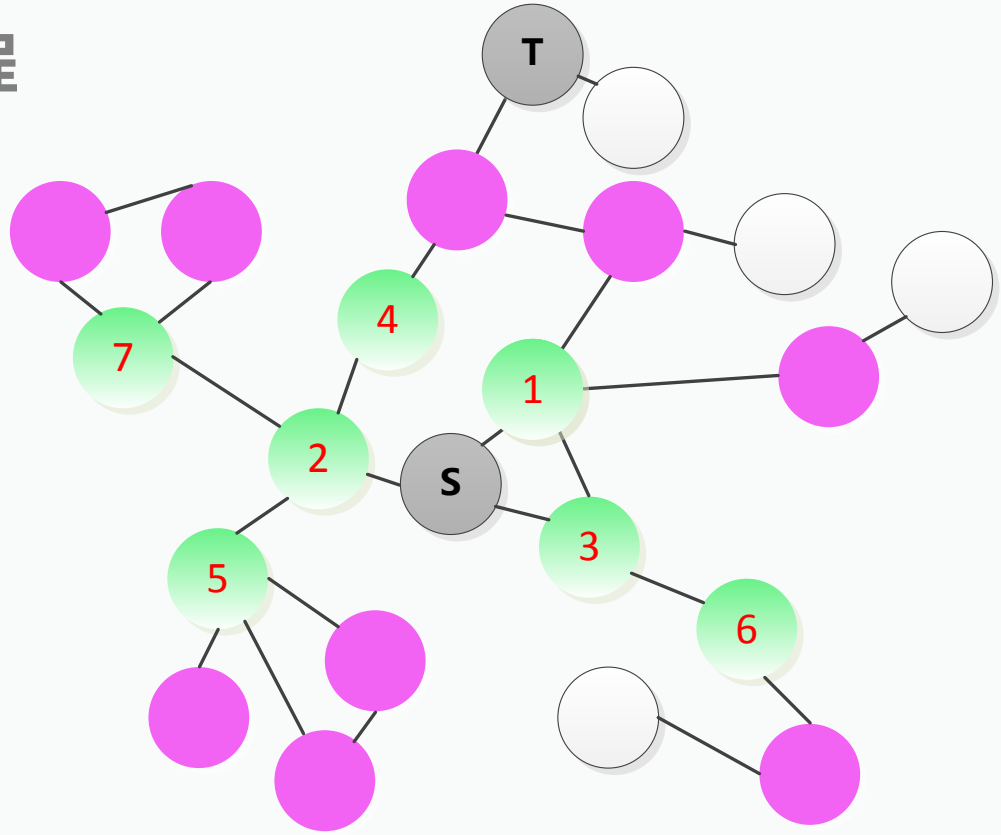
# 路径算法—Dijkstra 7/14

## 搜索过程



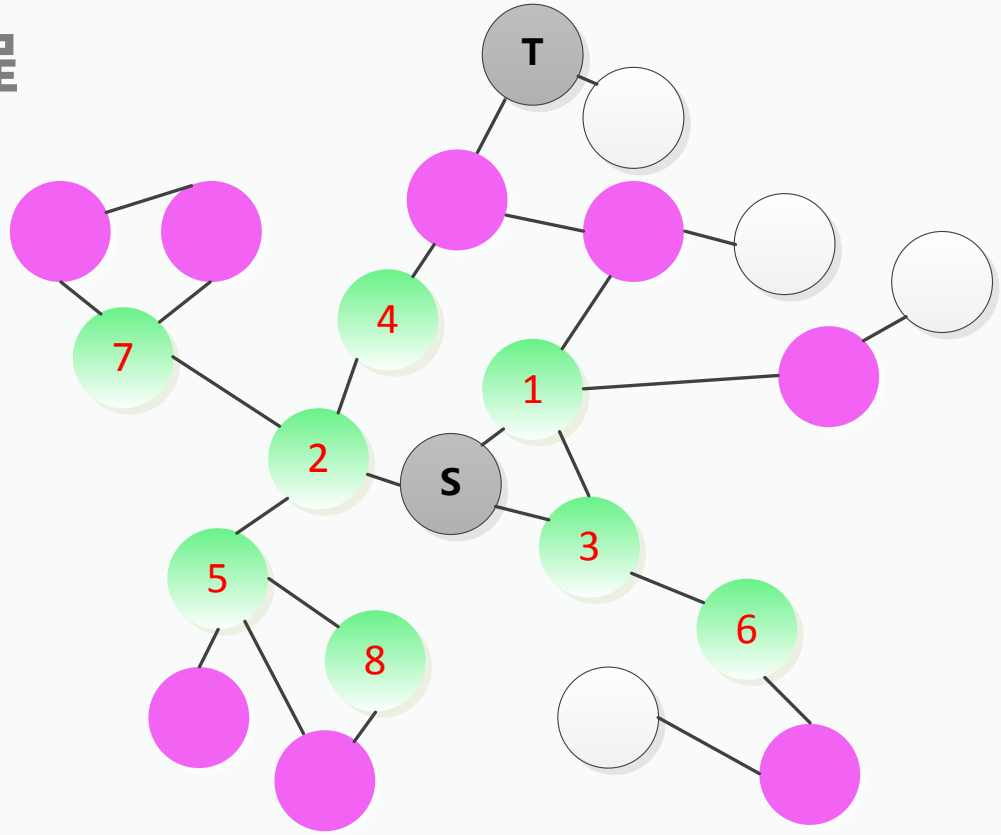
# 路径算法—Dijkstra 8/14

## 搜索过程



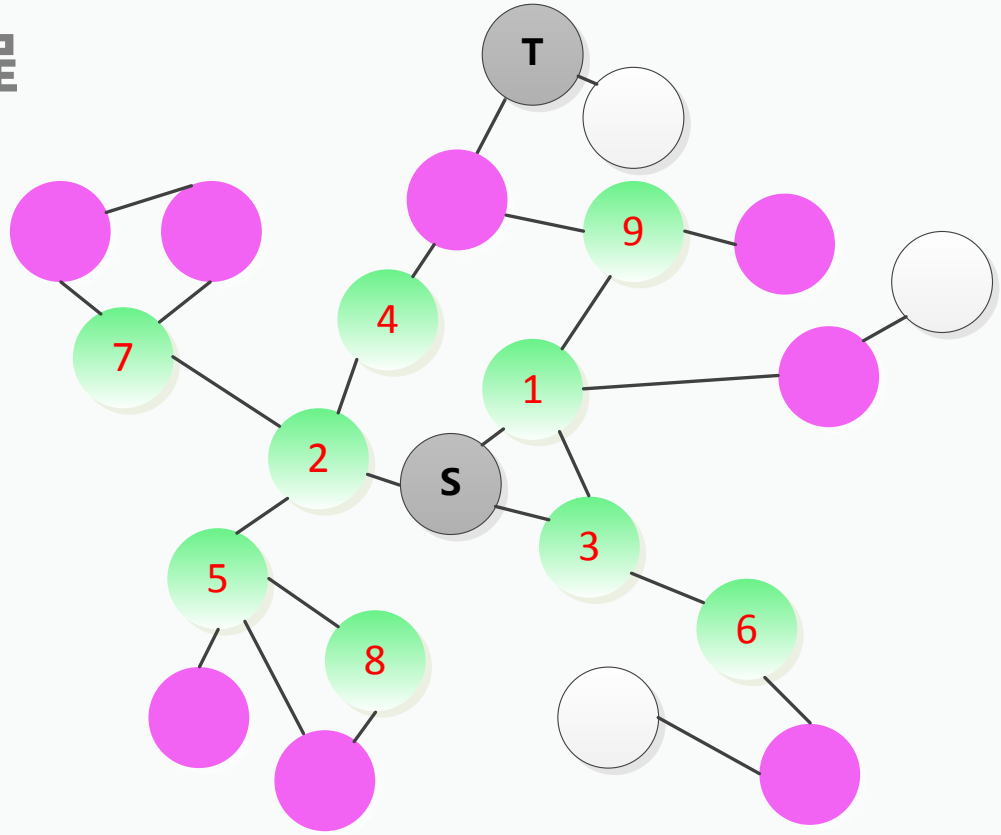
# 路径算法—Dijkstra 9/14

## 搜索过程



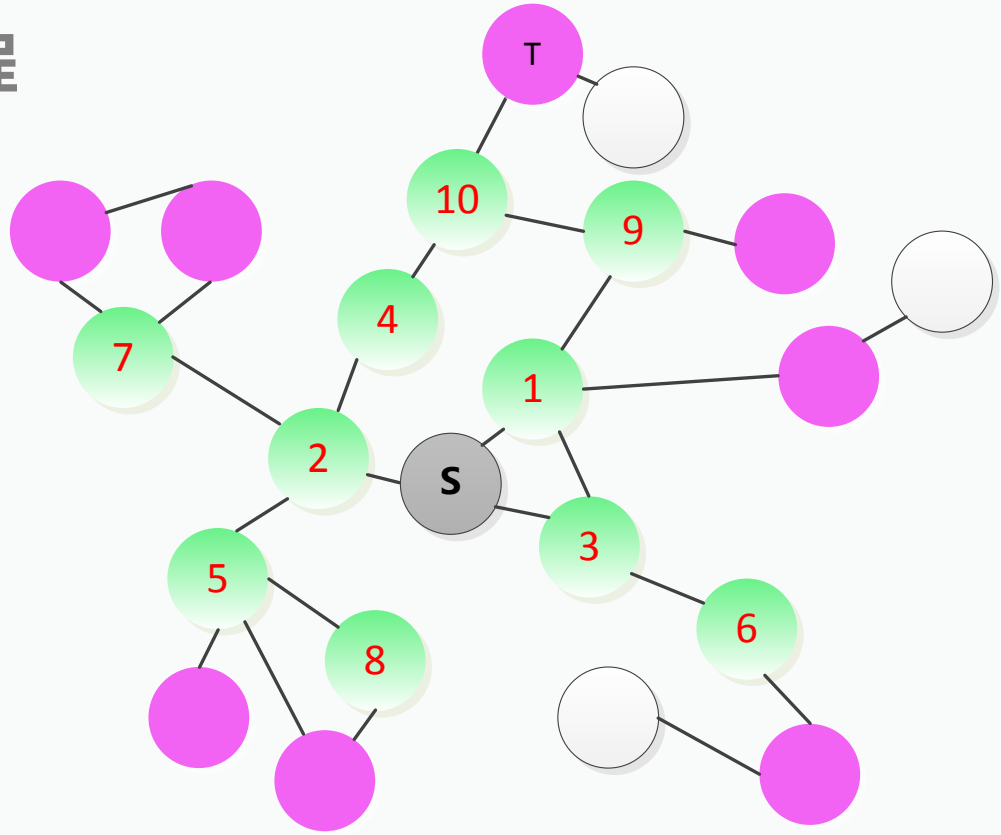
# 路径算法—Dijkstra 10/14

## 搜索过程



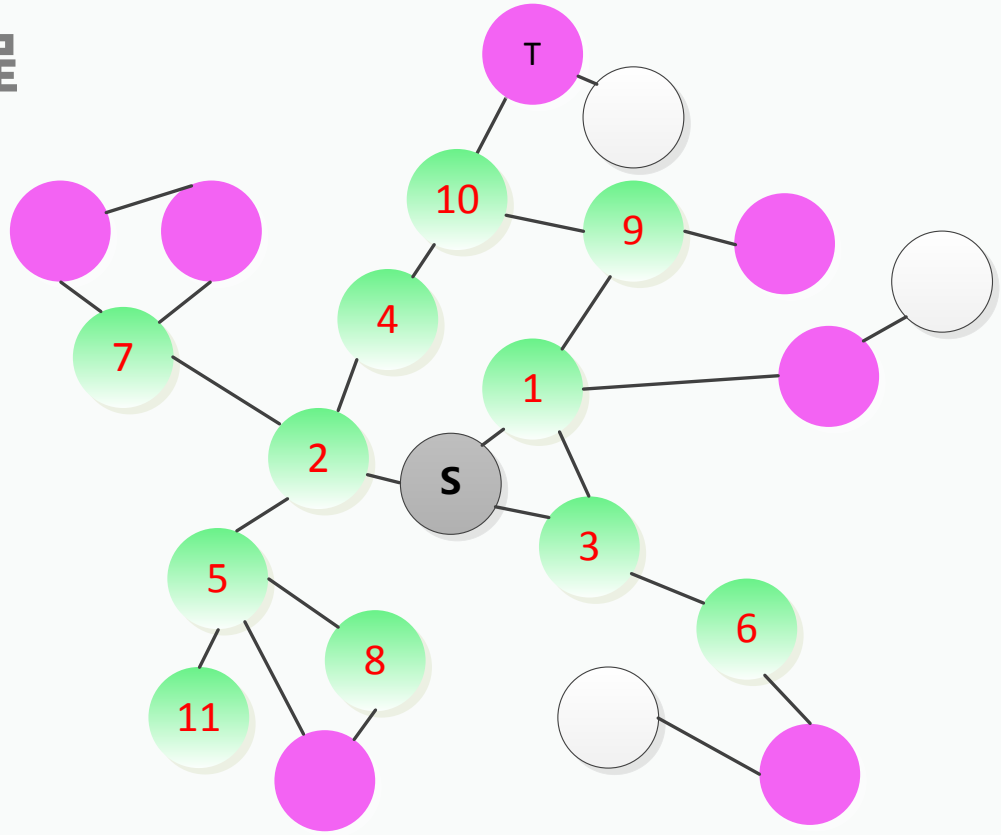
# 路径算法—Dijkstra 11/14

## 搜索过程



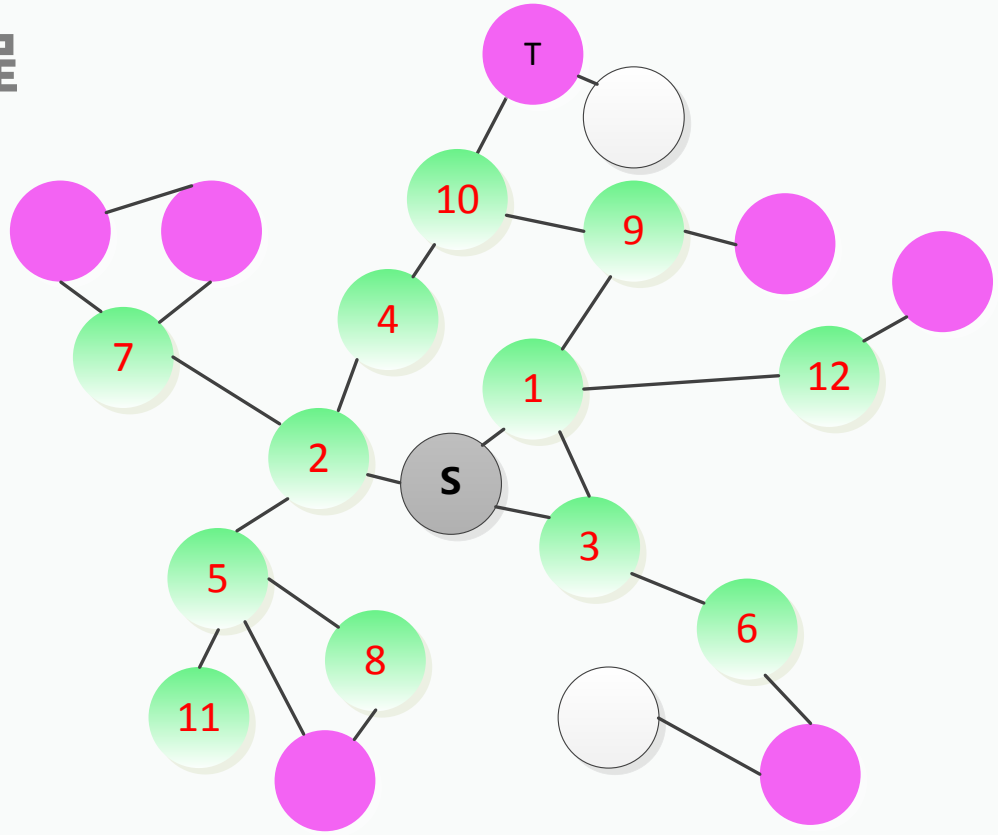
# 路径算法—Dijkstra 12/14

## 搜索过程



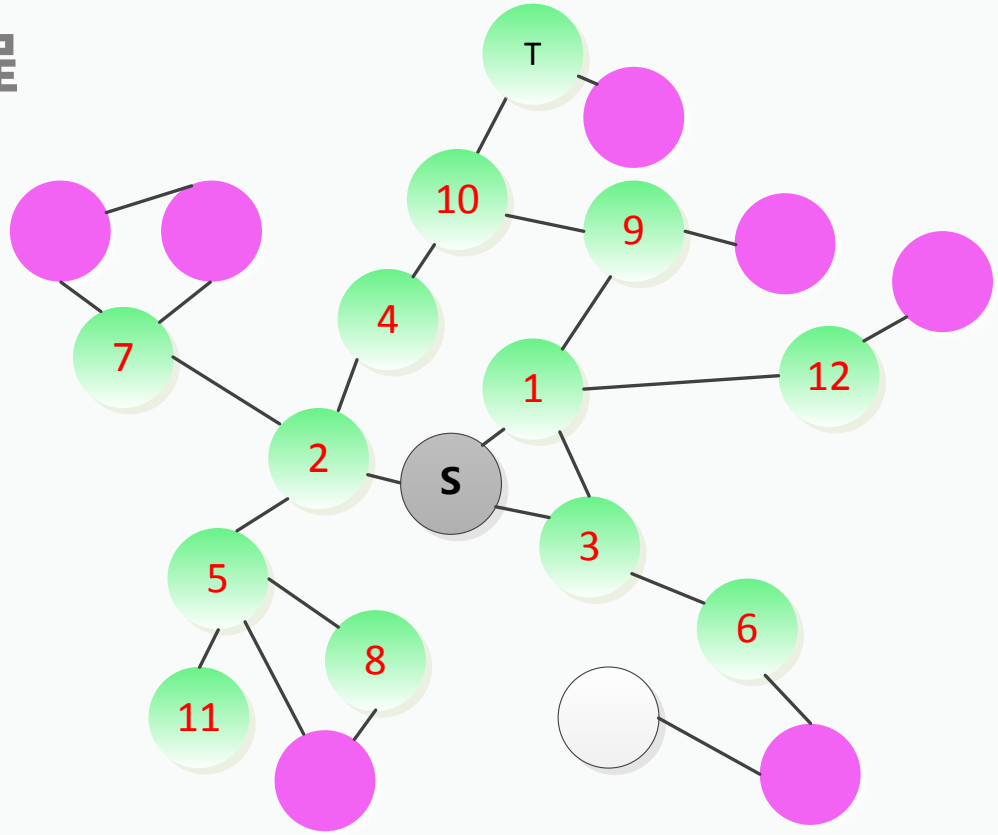
# 路径算法—Dijkstra 13/14

## 搜索过程



# 路径算法—Dijkstra 14/14

## 搜索过程



# 路径算法—Dijkstra

## 搜索现象

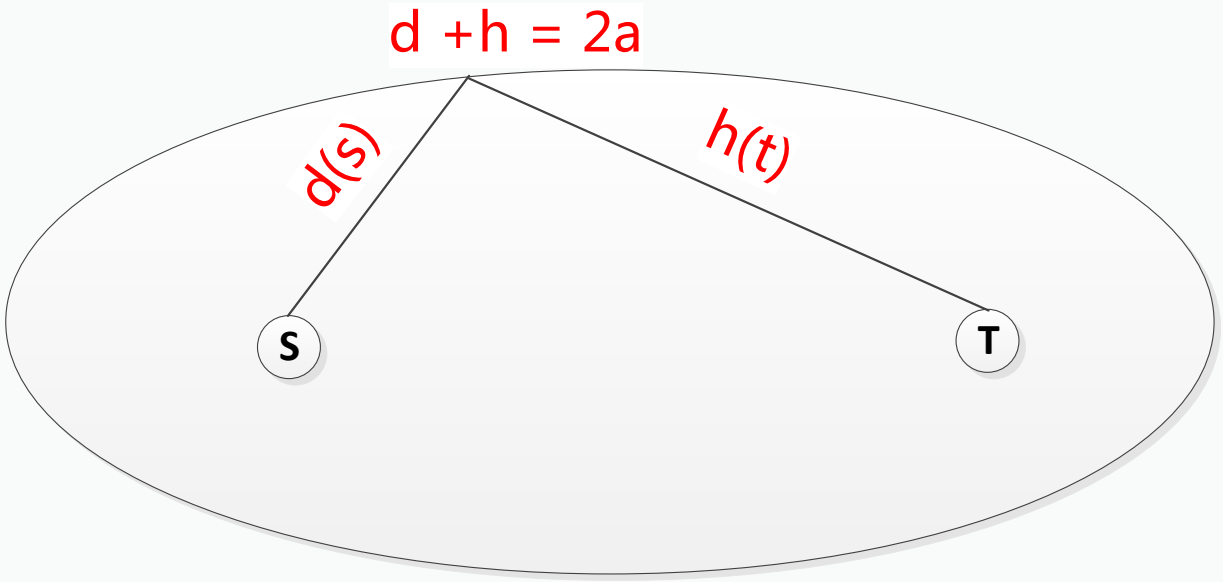
从起始点开始，由近至远，逐层向外扩展，直到搜索到目的点为止

## 存在问题

存在“南辕北辙”的现象，会走“冤枉徒劳”的路

# 路径优化策略—缩小搜索范围

## 椭圆定义

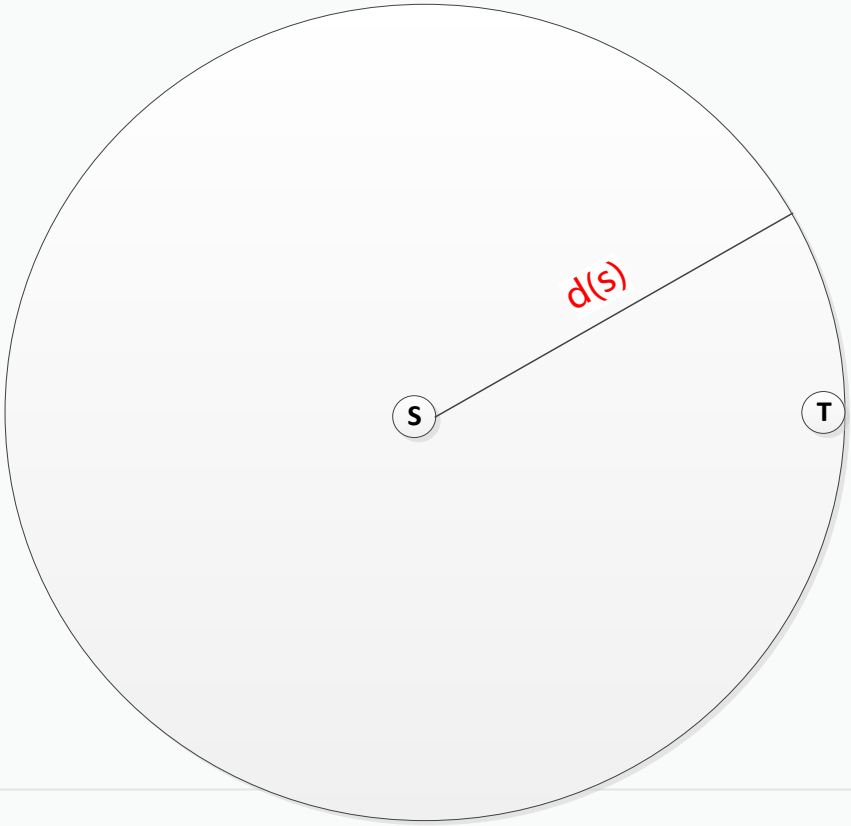


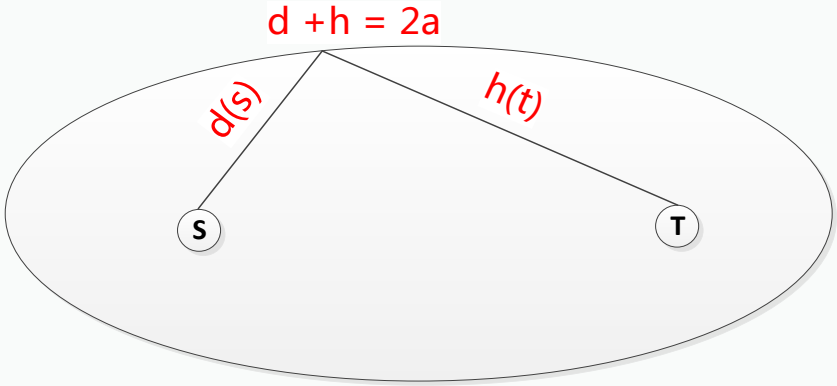
# 路径优化策略—Dijkstra vs A*

## 搜索目标函数

Dijkstra: d(s)

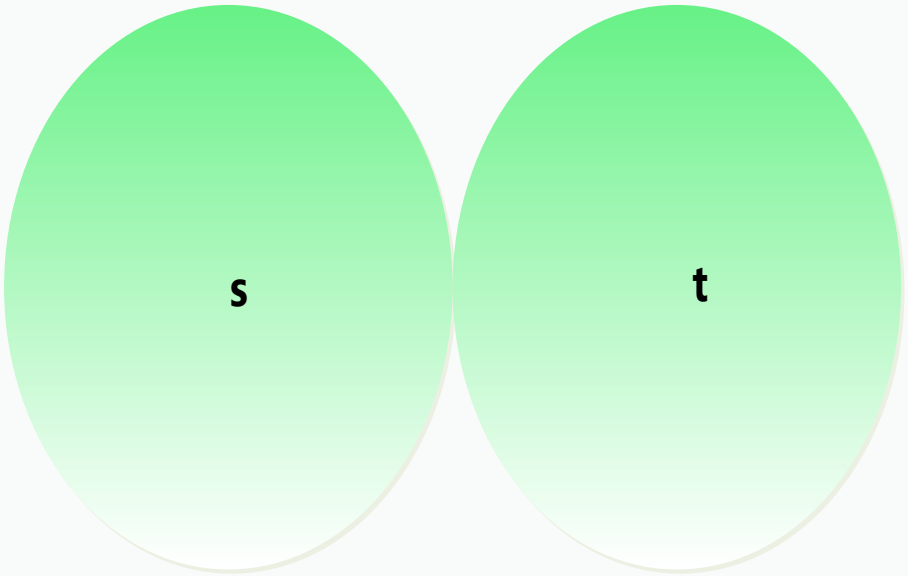
A*: d(s) + h(t)
constraint??



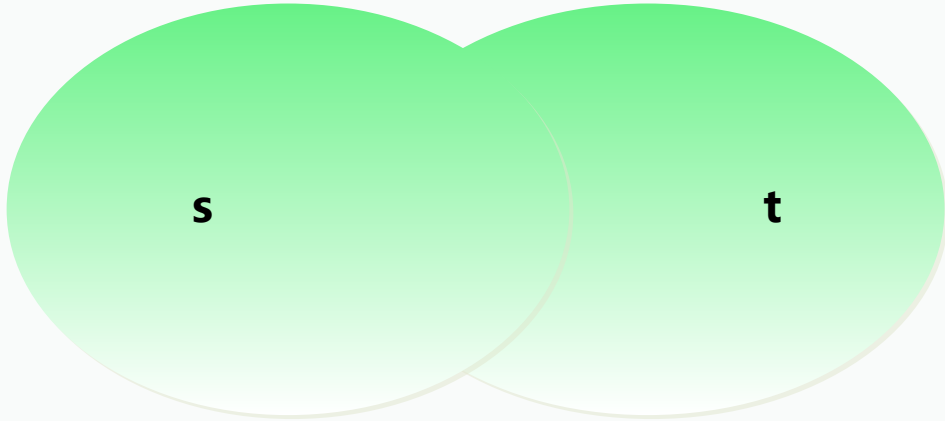



# 路径优化策略—双向搜索

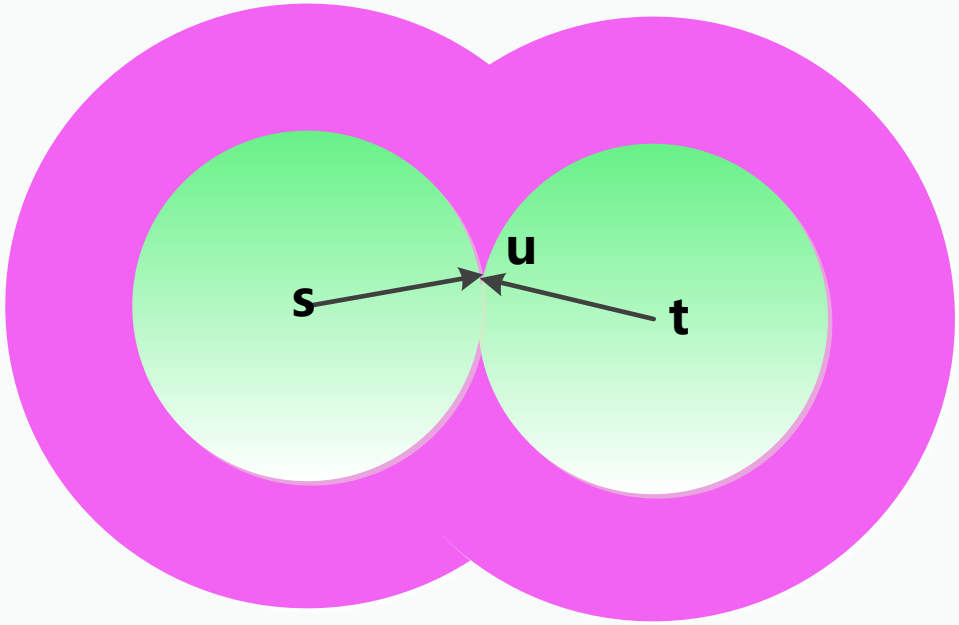
Dijkstra

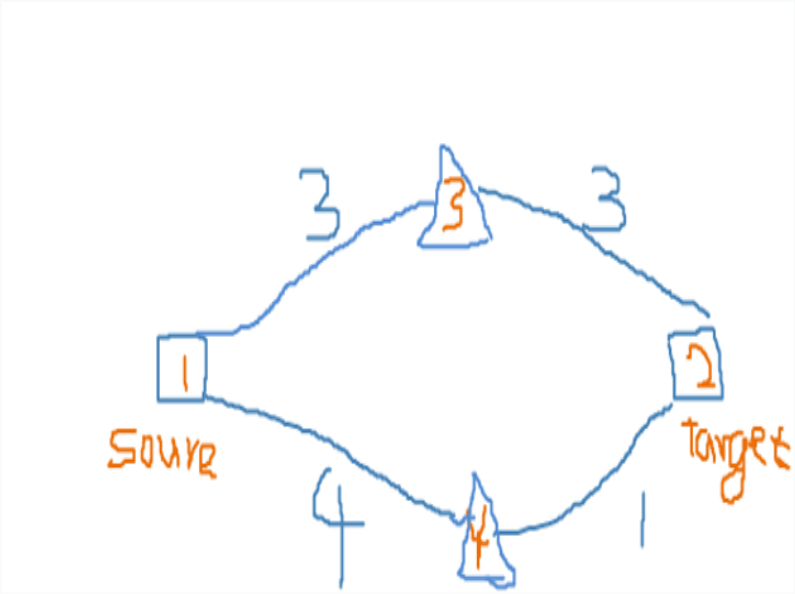



A*



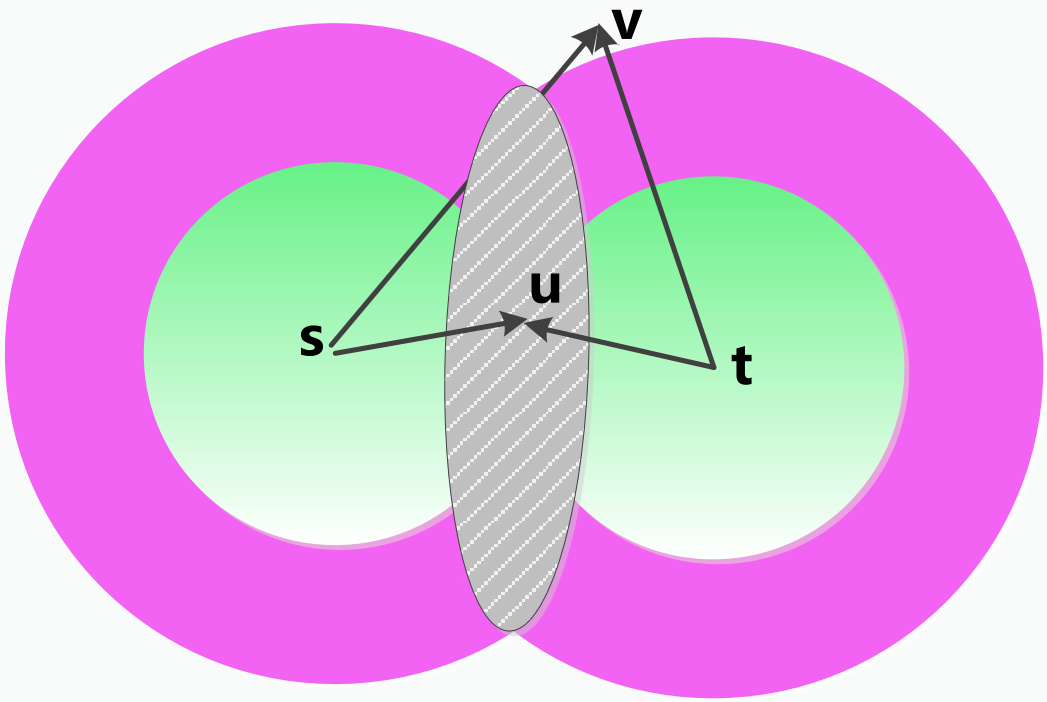
# 双向搜索—终止条件 1/3



# 双向搜索—终止条件 2/3

## 保守(conservative)




初次相遇后，最优meet点一定在两个粉色圆交集的阴影部分

# 双向搜索—终止条件 3/3

## 理论(aggressive)

Bi-Dijkstra

**s(u) + t(v) > min_cost(candidate_path)**

1 s(u): 表示当前正向搜索红色圆弹出的最小值
2 t(v): 表示当前反向搜索红色圆弹出的最小值
3 candidate_path: 表示候选路径中代价最小的

Bi-A*

**s(u) + t(v) > min_cost(candidate_path) + distance(s,t)/2**

1 distance(s,t) 表示起点与终点的估价函数(球面距离)

# 分层思想—稀疏路网密度

## 驾驶心理

**例子**—两个跨省的遥远村庄

驾驶过程
1 近距离：乡村小道、县城主干道
2 中距离：省道、高速
3 远距离：国道、高速

**问题**

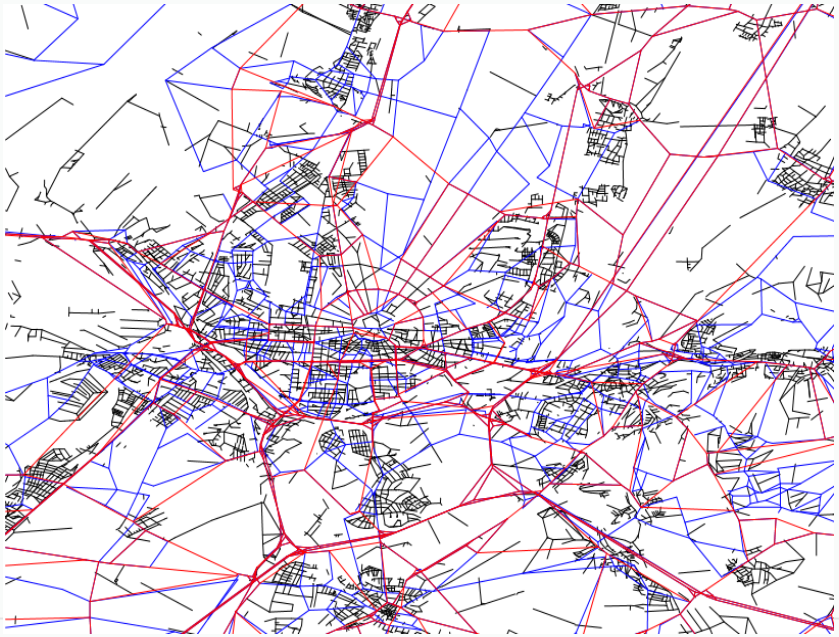
1 “中距离、更远距离”如何量化
2 “道路通行能力”如何量化

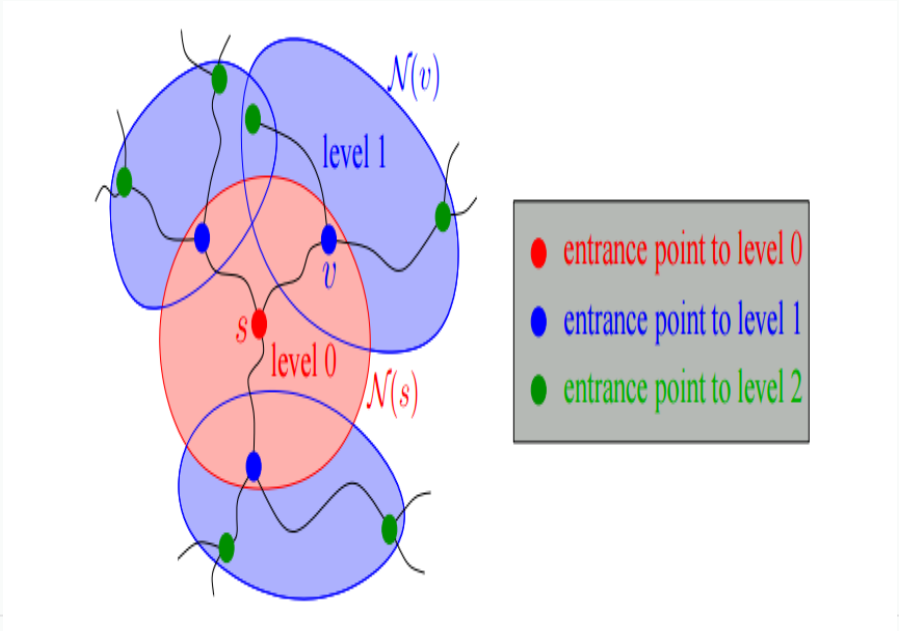
**反例**: 跨省或者跨县之间的道路联通性

# 分层思想—Highway Hierarchies(HHs)

## 干了两件事情：

1 量化所有边的不同层级(**level1**,**level2**,**level3..**)
2 确定每个节点在不同各自的层级(level1,**level2**,**level3..**)　对应的邻域半径(r1,**r2**,**r3**)...

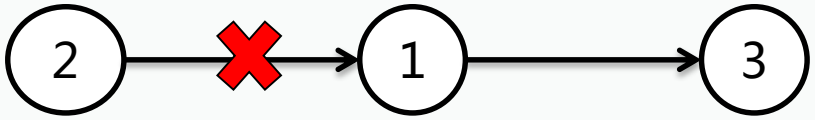




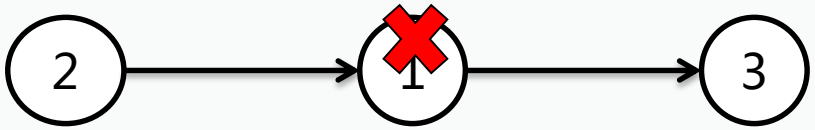
# 路径优化策略—稀疏路网密度

## 两种方法：

1 干掉边(Highway Upgrade ) Highway Hierarchies(HH)

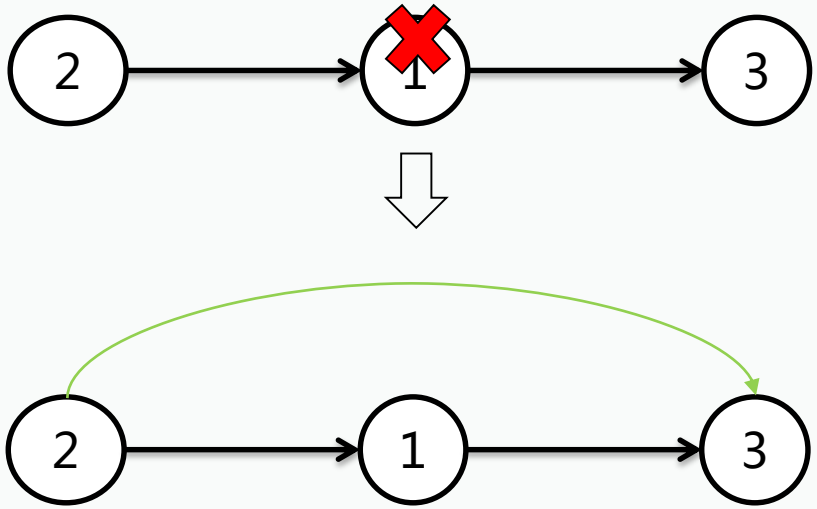



2 干掉节点(Node Contraction) Contraction Hierarchies(CH)



# 分层思想—Contraction Hierarchies(CH)　1/2

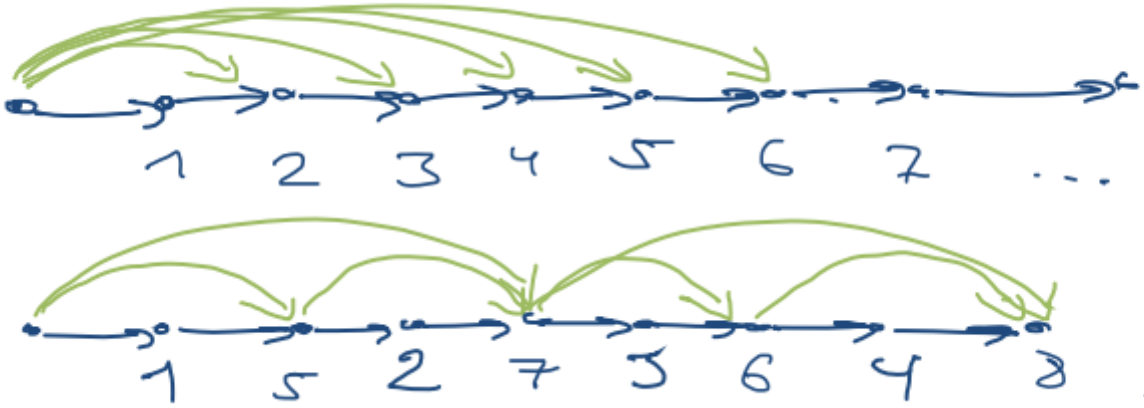
## 1 node contraction




保持拓扑的完整性，增加相应逻辑边（Shortcut）

# 分层思想—Contraction Hierarchies(CH)　2/2

## 2 node order



# 路径优化策略—预存储与实时取

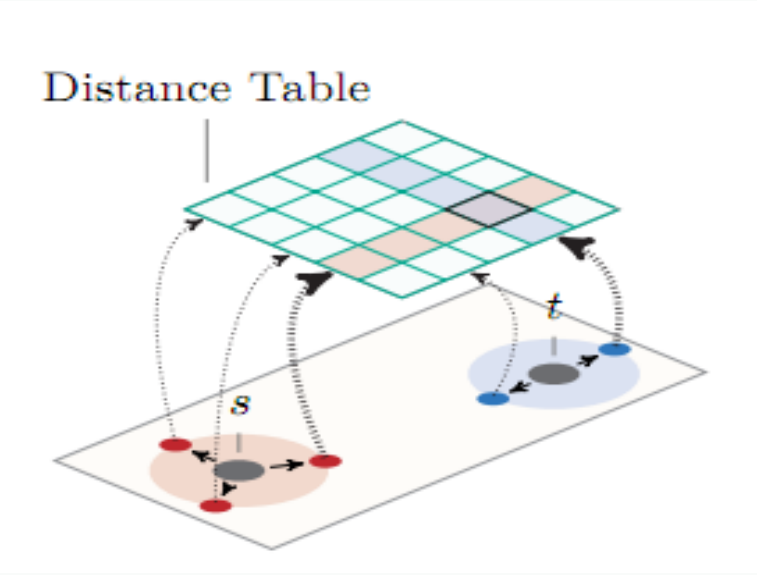
**极端做法**

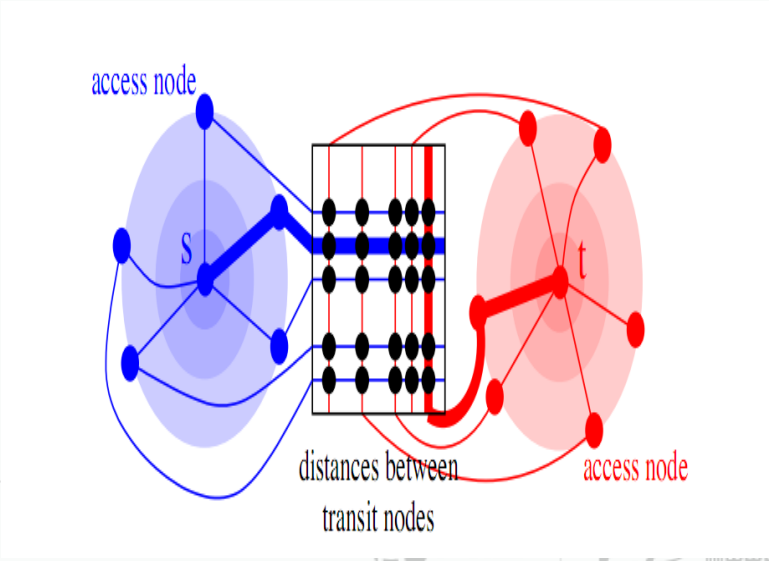
把所有的“节点对”的最优路径结果预计算搁在一张内存表里；
然后实时查询

**可行性**

1 空间：全国路网规模千万级，预存储空间规模>1000万*1000万
2 时间：路网路况实时更新，预存储表需要实时更新

## Transit Node Routing



# 路径算法优化策略总结—Synergy

## 1 双向搜索-bidirectional

## 2 缩小搜索空间范围—A*

a 直线距离

b ALT (A*, landmarks and triangle inequality)

c 多了些约束或剪枝条件

## 3 稀疏路网密度—分层思想（Hierarchy）

HH、CH

## 4 预存储和实时取—TimeTable

Transit Node Routing

# 算法评价

| method | data from | prepro. time [min] | prepro. overh. [B/n.] | query settled nodes | query time [ms] | uses CH | dom. by CH |
|---|---|---|---|---|---|---|---|
| Dijkstra[a] | [Bauer et al. 2010b] | 0 | 0 | 9.11 M | 5 591 | | |
| bidir. Dijkstra[a] | [Bauer et al. 2010b] | 0 | 0 | 4.76 M | 2 713 | | |
| eco. CH | this paper | **8** | 0.6 | 487 | 0.21 | ✓ | |
| aggr. CH | this paper | 27 | -2.1 | 356 | 0.15 | ✓ | |
| ALT-16[b] | [Goldberg et al. 2009] | 13 | 70 | 82 348 | 120.1 | | ✓ |
| ALT-64[a] | [Delling and Wagner 2007] | 68 | 512 | 25 234 | 19.6 | | ✓ |
| AF[c] | [Hilger et al. 2009] | 2 156 | 25 | 1 593 | 1.1 | | ✓ |
| REAL[b] | [Goldberg et al. 2009] | 103 | 36 | 610 | 0.91 | | ✓ |
| HH | [Schultes 2008] | 13 | 48 | 709 | 0.61 | | ✓ |
| HNR | [Schultes 2008] | 15 | 2.4 | 981 | 0.85 | | ✓ |
| SHARC[a] | [Bauer and Delling 2009] | 81 | 14.5 | 654 | 0.29 | | ✓ |
| bidir. SHARC[a] | [Bauer and Delling 2009] | 158 | 21.0 | 125 | 0.065 | | ✓[d] |
| CALT[a] | [Bauer et al. 2010b] | 11 | 15.4 | 1 394 | 1.34 | | ✓ |
| eco. CH+AF[a] | [Bauer et al. 2010b] | 32 | 0.0 | 111 | 0.044 | ✓ | |
| gen. CH+AF[a] | [Bauer et al. 2010b] | 99 | 12 | **45** | 0.017 | ✓ | |
| partial CH[a] | [Bauer et al. 2010b] | 15 | **-2.9** | 965 k | 53.63 | ✓ | |
| TNR | this paper | 46 | 193 | N/A | 0.0033 | ✓[e] | |
| TNR+AF | [Bauer et al. 2010b] | 229 | 321 | N/A | **0.0019** | ✓[e] | |

## 效率并非银弹：

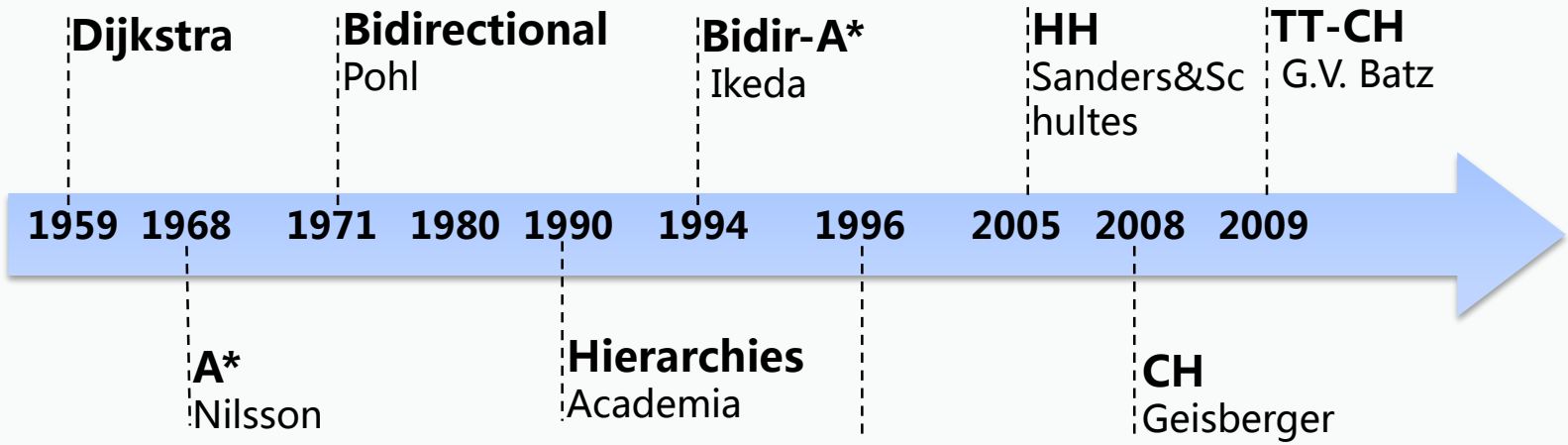
实际产业界：

结合考虑限行、交通管制、动态交通等多种的实际feature

理想的算法:

平衡实时求路的效率与道路代价模型(RCM)的变化

# 路径算法-发展历史



# 路径算法研究单位—KIT

Peter Sanders

http://algo2.iti.kit.edu/routeplanning.php

- Highway Hierarchies — Sanders&Schultes, July 05
- Transit-Node Routing — Sanders&Schultes, July 06
- Highway-Node Routing — Sanders&Schultes, July 06
- Contraction Hierarchies — Geisberger&Sanders&Schultes, July 08

Dorothea Wagner

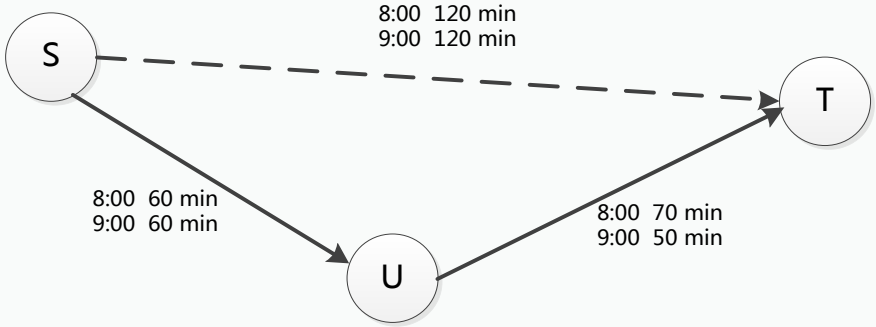
http://i11www.iti.uni-karlsruhe.de/en/projects/route_planning/index

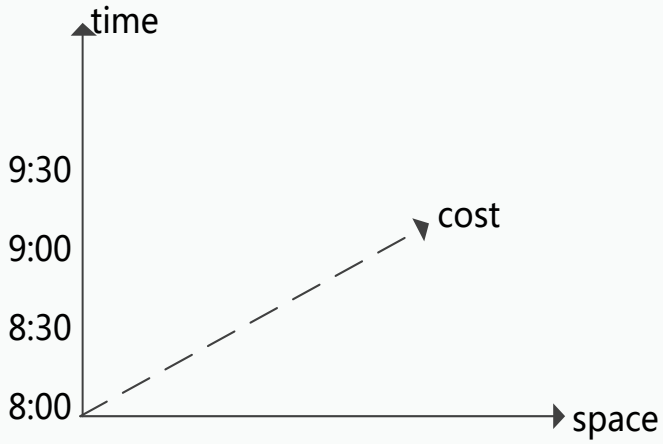
- Multi-Criteria
- Time-dependent Routing
- Public Transportation
- Multi-Modal
- Hierarchical Speed-up
- Separator-based Multi-Level
- Goal-Directed Speed-up
- SHARC
- Geometric Container
- Combinations of Speed-up

# 路径服务未来—挑战

## 1 时间依赖 time-dependent

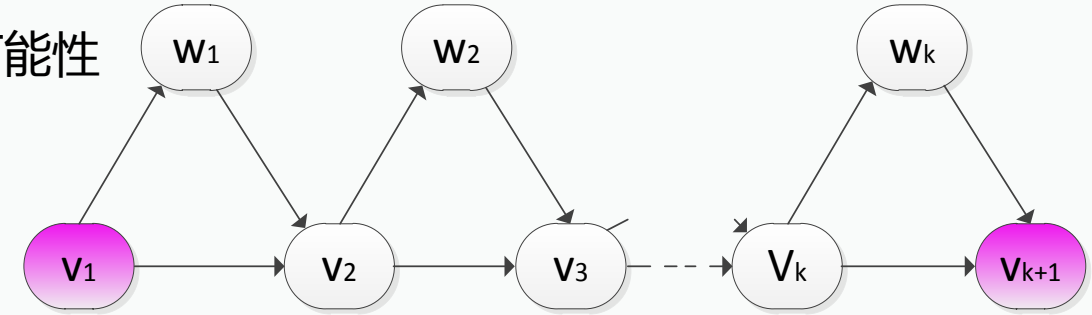
### a 动态路径的时间相对性




### b 问题复杂性

Path($V_1$, $V_{k+1}$): $2^k$ 种可能性



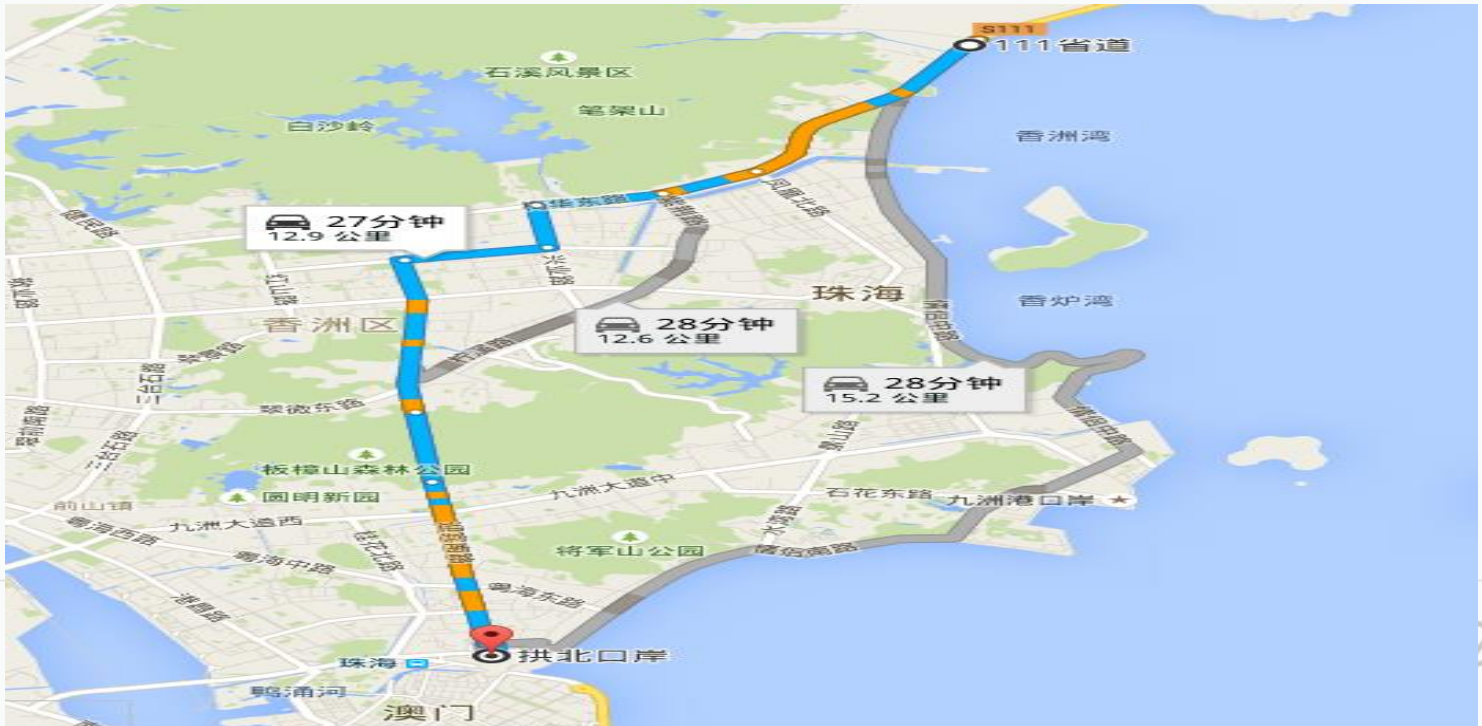
# 路径服务未来—挑战

## 2 定制化路线 personalized route

a 凭什么你给的路线就是我想要的？

b 用户偶尔不按我们给的路线来行驶？

c 用户在任一时间和地点出发时，我们可以知道用户去哪里？



# 参考


1 http://algo2.iti.kit.edu/english/publications.php

2 http://i11www.iti.uni-karlsruhe.de/en/projects/route_planning/index

3 http://ad-wiki.informatik.uni-freiburg.de/teaching/EfficientRoutePlanningSS2012

4 http://www.atatech.org/article/detail/31724/0

5 http://www.atatech.org/article/detail/31815/0

6 http://www.atatech.org/article/detail/31725/0?rnd=1930996854

01


高德 AutoNavi

Alibaba 阿里巴巴集团