A search problem can be defined formally by five components:

- Initial state
- Actions
- Transition model
- Goal test
- Path cost

Solution: a path (i.e., an action sequence) from the initial state to the goal state

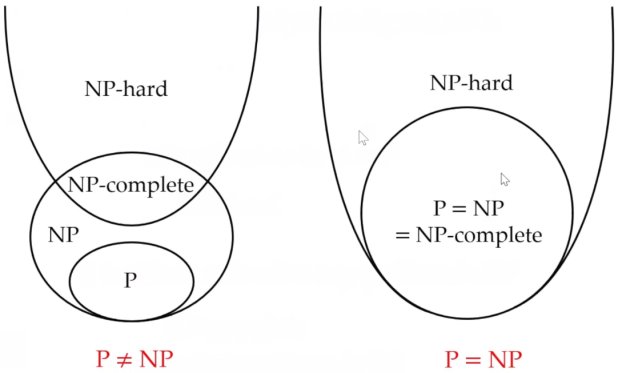
Optimal solution: a path with the lowest cost

- 初始状态
- 动作
- 传递模型
- 目标测试
- 动作代价

旅行商问题

- 带权图，是否有cost至多为k的tour，遍历所有城市一次并返回
- 着色：任意边两个端点不同色，看能不能k种颜色染完

优化问题转换为判断问题：尝试不同的k

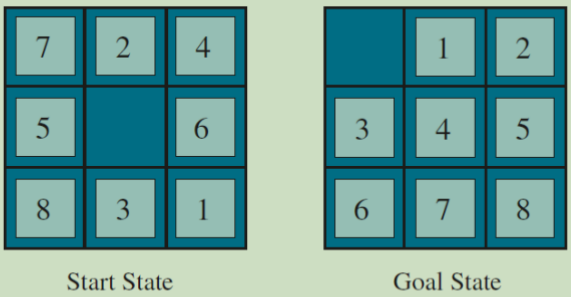


- **P Problem**: 对于任意的输入规模 n，问题都可以在 n 的多项式时间内**得到解决**;
- **NP(Non-deterministic Polynomial) Problem**: 可以在多项式的时间里**验证**一个解的问题;
- **NPC(Non-deterministic Polynomial Complete) Problem**: 满足两个条件:
  - 是一个 NP 问题
  - 所有的 NP 问题都可以约化到它
- **NP-hard Problem**: 满足NPC问题的第 2 条，但**不一定要满足第 1 条。**（NP-Hard问题要比 NPC问题的范围广）

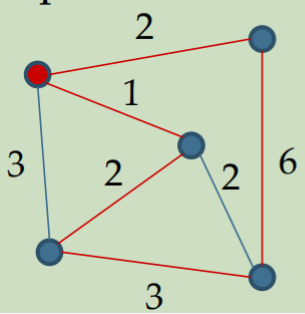
> NP：N是非确定性算法 。猜解，验证解，若不满足则不输出
>
> NP-hard>=NP

> NP-complete是前两个的交集，最难的

- *n*-puzzle: NP-complete



- Travelling salesman problem: NP-hard



Tree-search-frontier:

- 下一步可探索结点集合，为空时失败
- 访问子节点时，包含其他子节点和孙节点

Graph-search-explored

- 下一步可探索结点集合，为空时失败
- 记录已探索节点

> 搜索算法都有tree和graph两个版本
> 完备性，最优性，时间空间复杂度

$f \in O(g)$	$f \leq g$
$f \in o(g)$	$f < g$
$f \in \Omega(g)$	$f \geq g$
$f \in \omega(g)$	$f > g$
$f \in \Theta(g)$	$f = g$