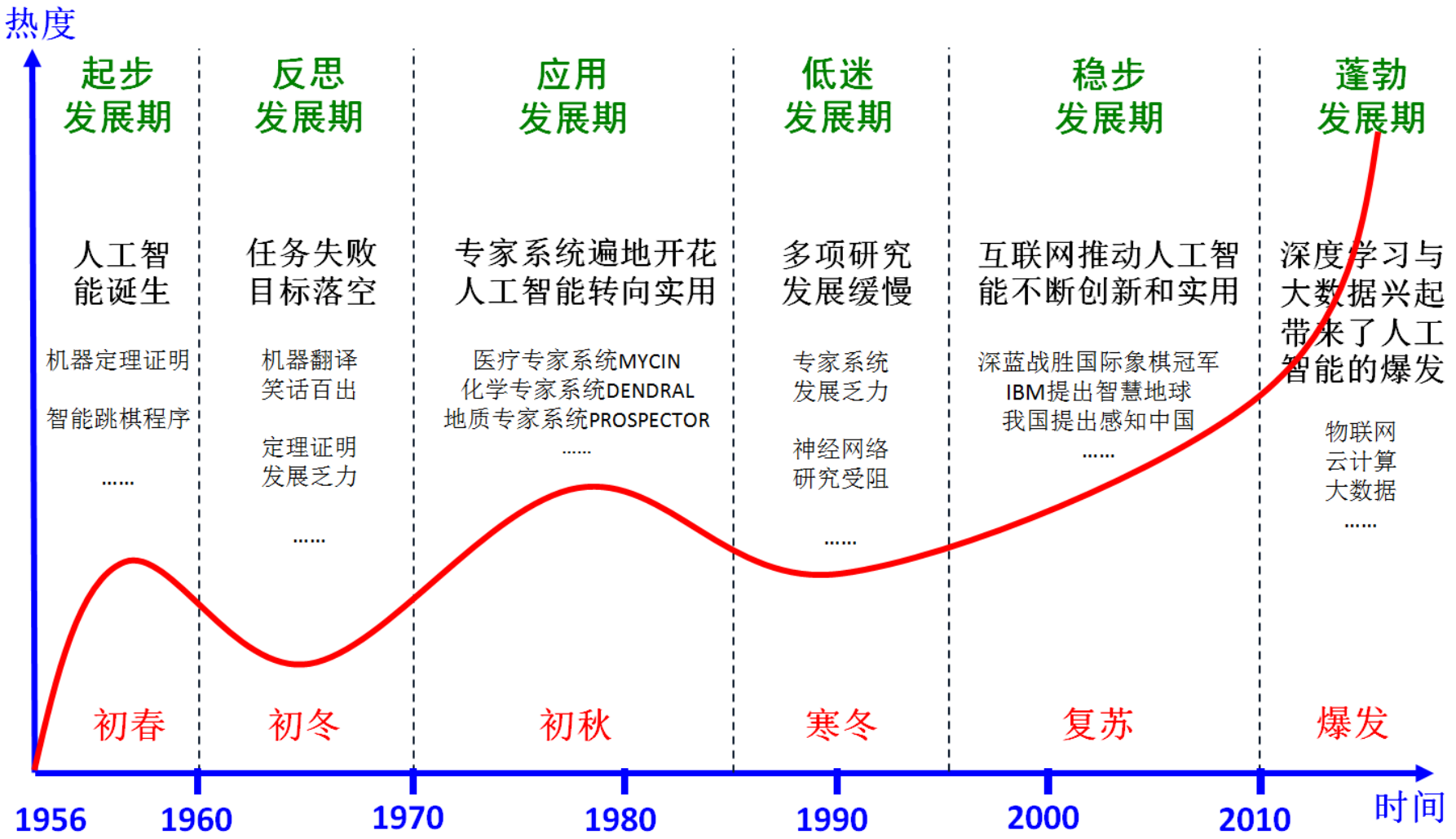
《机器学习》课件

2.2 增强学习

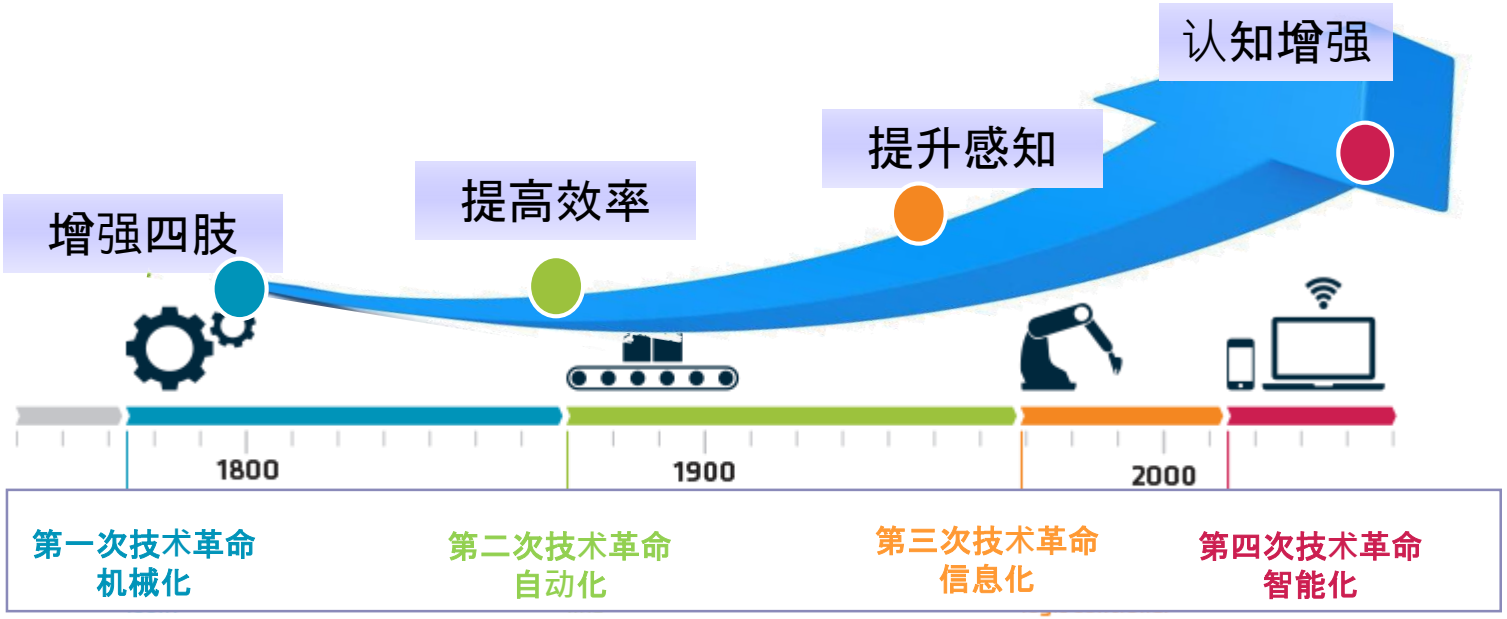
背景

人工智能发展历程


热度
起步发展期
反思发展期
应用发展期
低迷发展期
稳步发展期
蓬勃发展期
人工智能诞生
机器定理证明
智能跳棋程序
.....
任务失败目标落空
机器翻译笑话百出
定理证明发展乏力
.....
专家系统遍地开花人工智能转向实用
医疗专家系统MYCIN
化学专家系统DENDRAL
地质专家系统PROSPECTOR
.....
多项研究发展缓慢
专家系统发展乏力
神经网络研究受阻
.....
互联网推动人工智能不断创新和实用
深蓝战胜国际象棋冠军
IBM提出智慧地球
我国提出感知中国
.....
深度学习与大数据兴起带来了人工智能的爆发
物联网
云计算
大数据
.....
初春
初冬
初秋
寒冬
复苏
爆发
1956
1960
1970
1980
1990
2000
2010
时间

背景

- 人工智能-- 第四次技术革命的基石



- 人工智能将在提升机器认知能力基础上，全面整合人类本身智能以及人类过去文明成果，从而提升乃至解放人类认知。

背景

- 人工智能发展最大特点：将渗入到人类社会的各个领域，重塑产业格局以及社会结构

国防

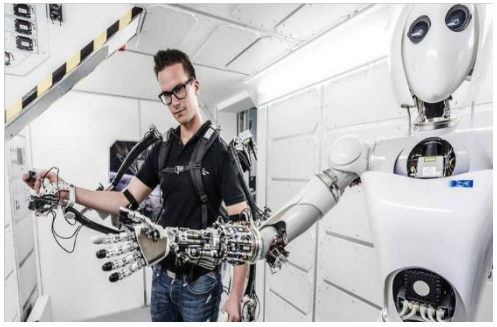
工业

农业

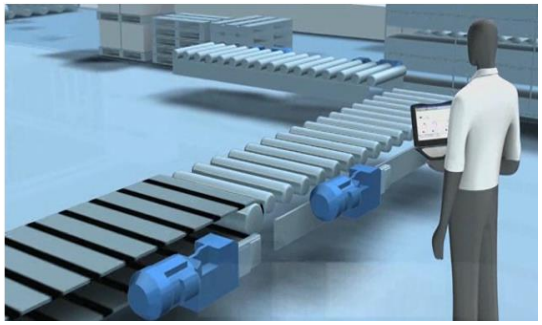
医疗

教育

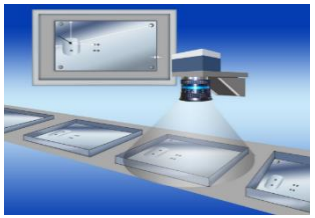
人机协作

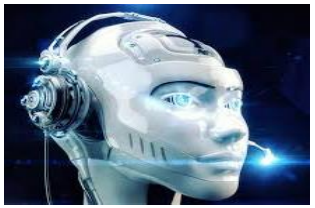


人机决策



2017年5月，谷歌公司的人工智能软件“Alpha Go”战胜目前围棋世界排名第一的柯洁



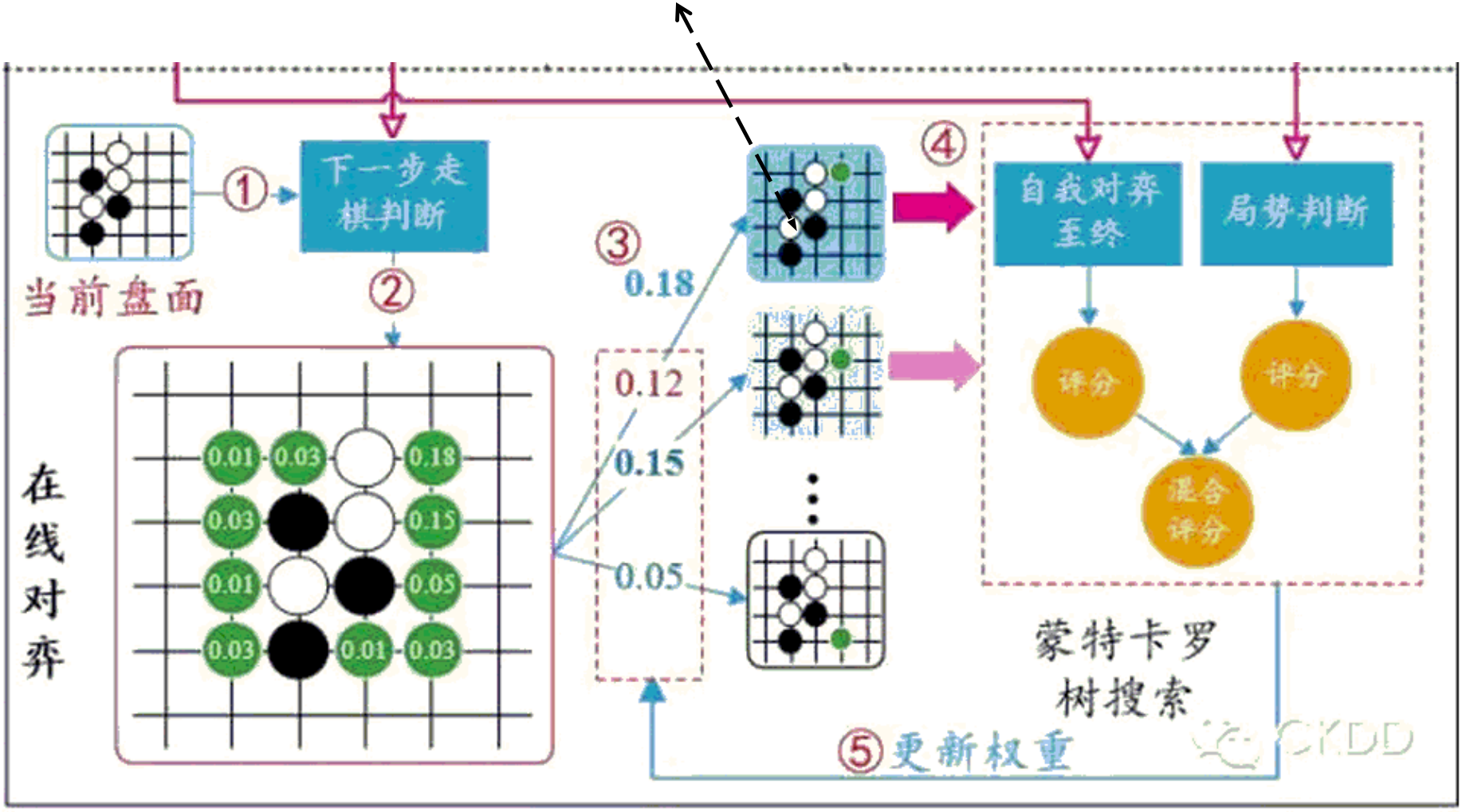


人工智能领域的里程碑

AlphaGo技术的三大核心内容

- 增强学习：Q学习
  - 重点、难点、掌握机器人、无人机路径规划中的应用
- 深度学习
  - 难点、后续课程介绍
- 蒙特卡洛搜索树
  - 了解如何结合Q学习实现AlphaGo，上节课内容

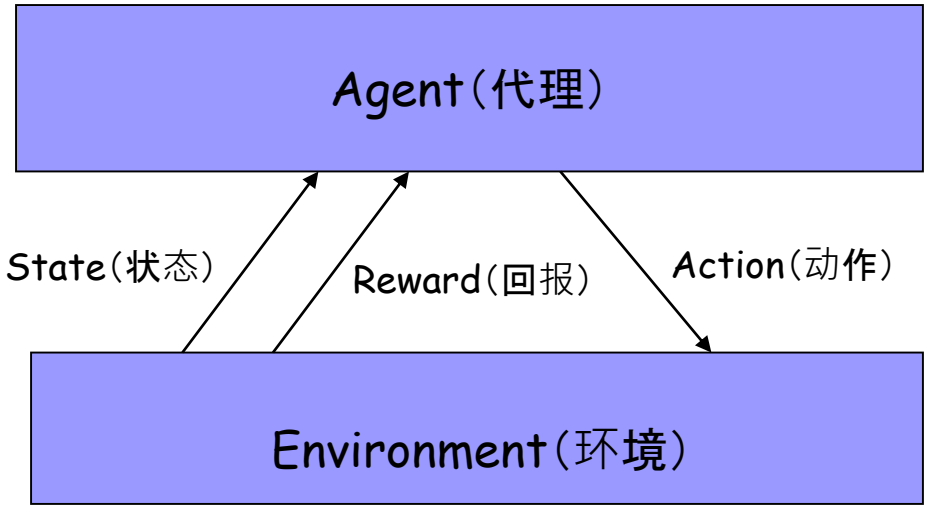
增强学习：学习如何下棋

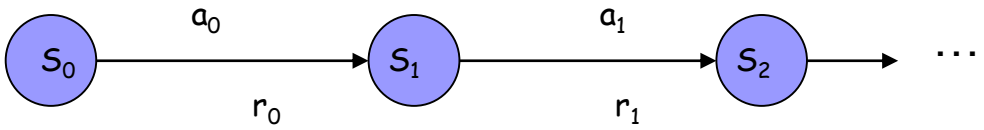


值，则在蒙特卡罗树上进一步展开下一级别的搜索（如图2所示）。

Q-学习通用算法
不限于围棋

Q学习-增强学习算法

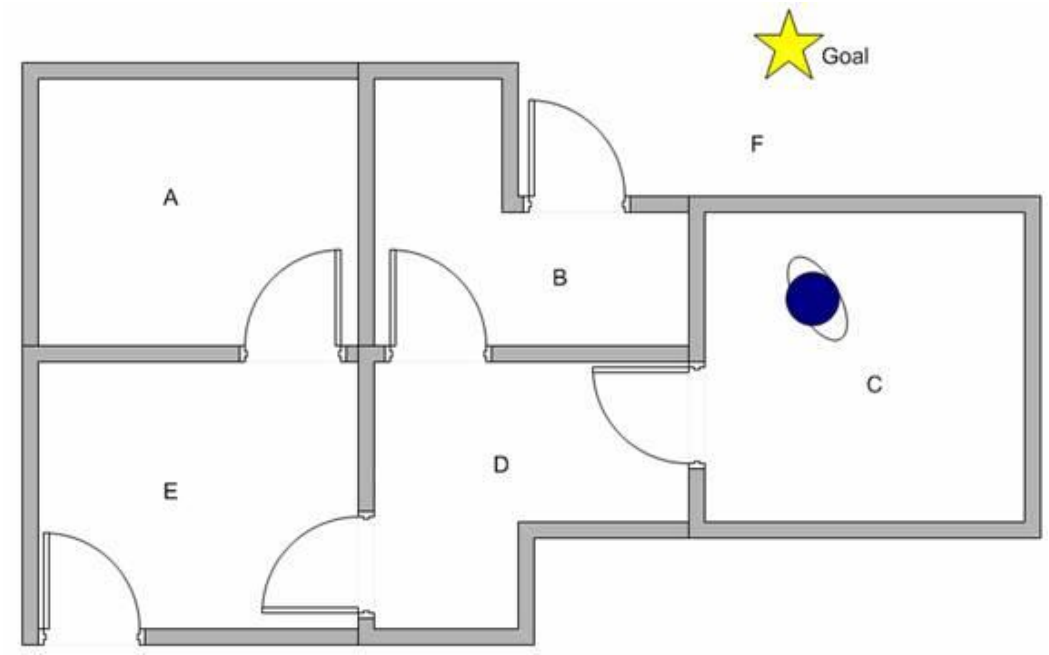




以回报为核心的通用算法：Agent在所处环境总是基于回报最大选择合适的动作，从而完成与环境的交互，完成特定任务。

Q学习算法

- 机器人路径规划

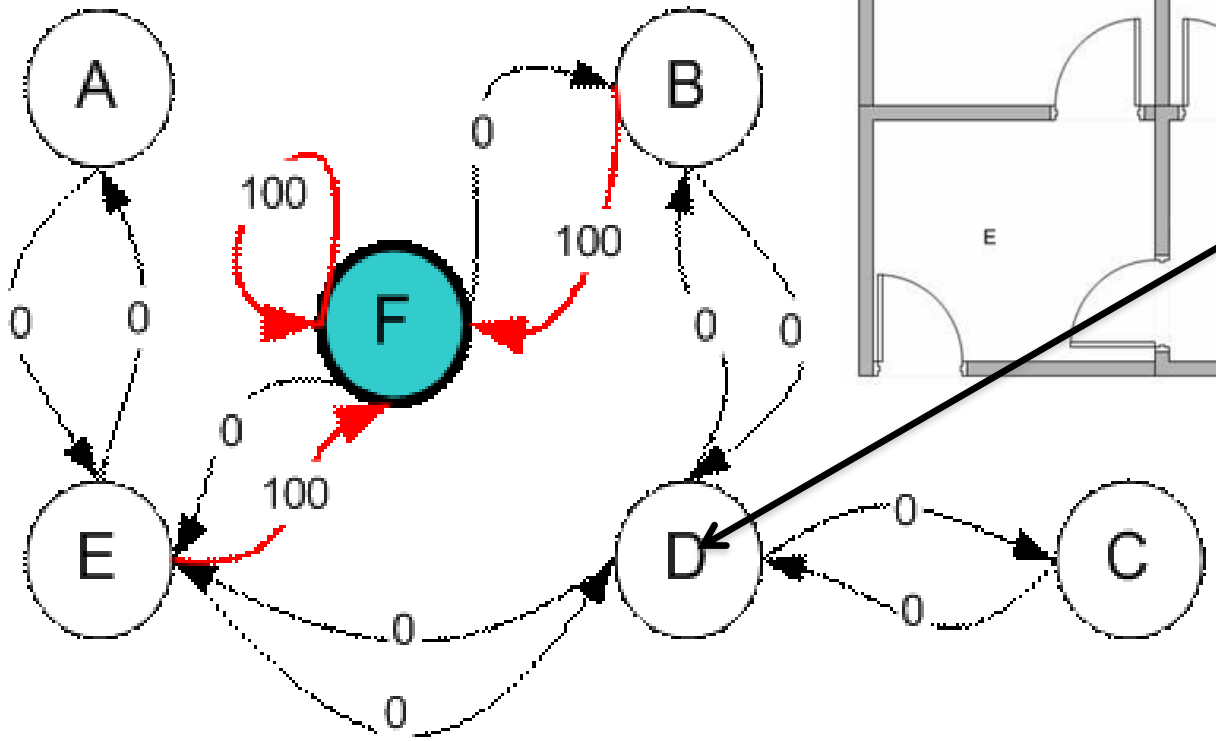


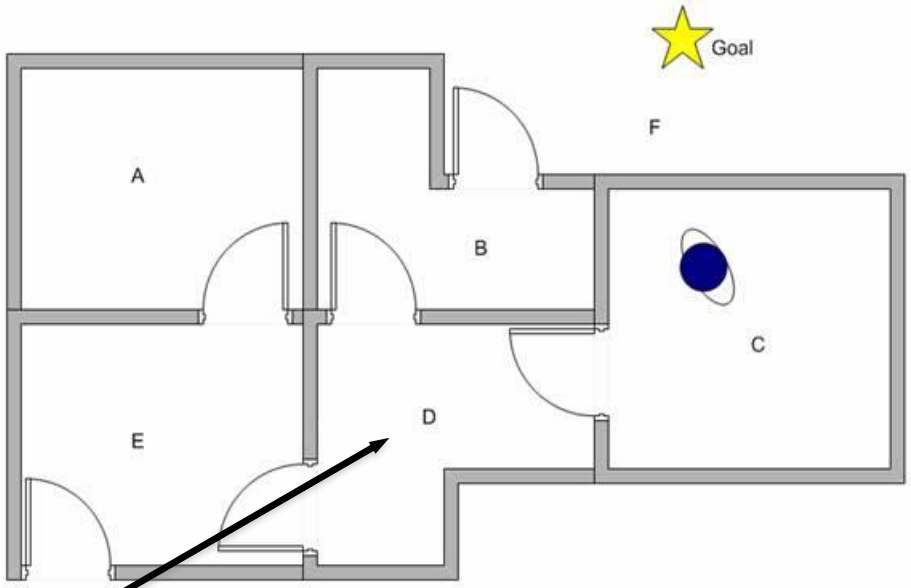
A to E: rooms, F: outside building (target).

任务:处于任何一个房间的机器人都能够找到一条最优路径走到室外

环境建模图表示

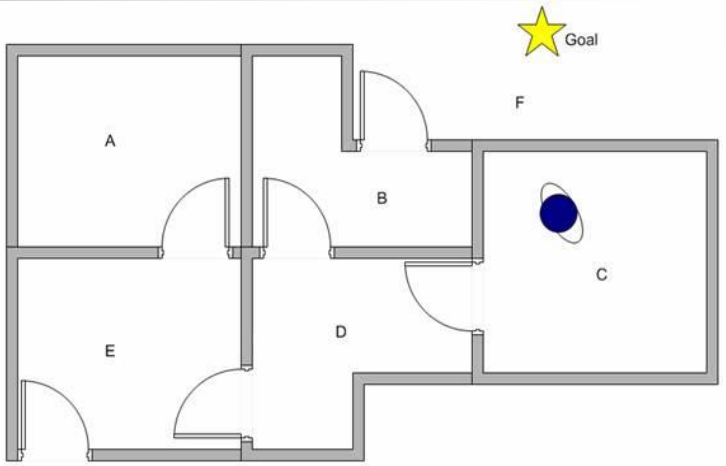
- 图表示，节点表示房间号，边表示动作，它的值表示回报，例如E
- 基于回报进行建模



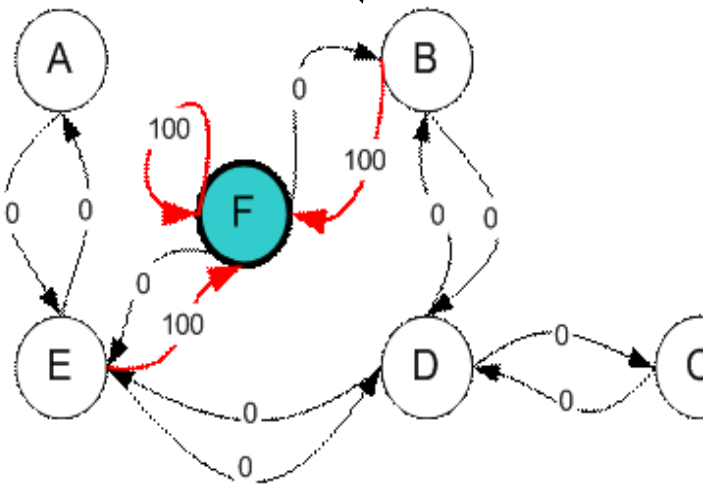


问题：如何存储？

- 环境建模矩阵表示
  - 回报矩阵
  - 图表示等价形式
- 建立状态和行为对应关系



$$\mathbf{R} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} state\backslash action & A & B & C & D & E & F \end{array} \\ \begin{bmatrix} - & - & - & - & 0 & - \\ - & - & - & 0 & - & 100 \\ - & - & - & 0 & - & - \\ - & 0 & 0 & - & 0 & - \\ 0 & - & - & 0 & - & 100 \\ - & 0 & - & - & 0 & 100 \end{bmatrix} \end{array}$$



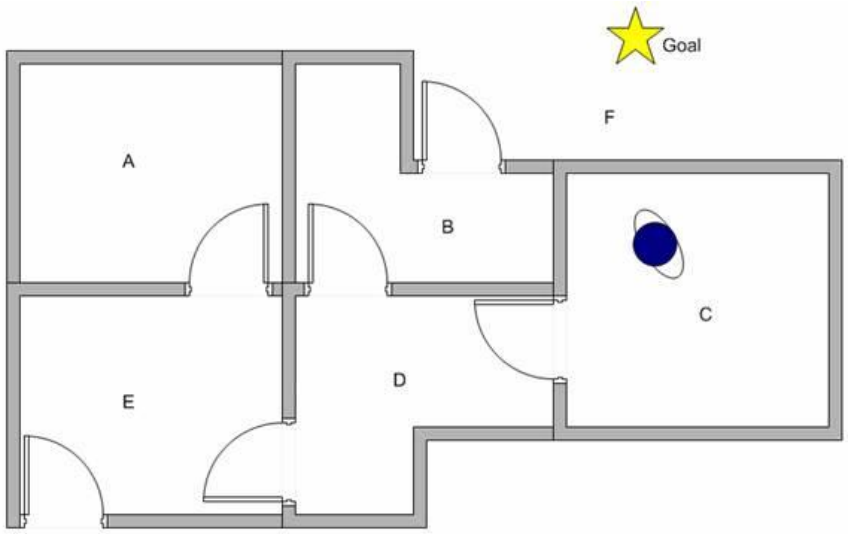
思考：由于元素的值多数为0，无法求解，如何计算更加合理的回报？

Q学习算法核心概念

- Q矩阵进行回报学习
- 更加合理的回报:状态/动作

$$\mathbf{R} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} - & - & - & - & 0 & - \\ - & - & - & 0 & - & 100 \\ - & - & - & 0 & - & - \\ - & 0 & 0 & - & 0 & - \\ 0 & - & - & 0 & - & 100 \\ - & 0 & - & - & 0 & 100 \end{bmatrix} \end{array}$$

$$\mathbf{Q} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$



$$\mathbf{Q}(state, action) = \mathbf{R}(state, action) + \gamma \cdot Max\left[\mathbf{Q}(next\ state, all\ actions)\right]$$

$0 \le \gamma < 1$ 学习率

Q矩阵的计算是一个迭代过程，决定了算法设计；

思考问题:为什么是增强学习？

Q学习算法

- 输入：状态/动作-回报矩阵(R)，目标状态Goal；
- 输出：从任何初始状态到目标状态的最小路径(Q)；

1. 设置学习率γ，环境汇报矩阵R
2. Q = 0；
3. 进行迭代循环:
   - 随机选择初始状态
   - 如果不是目标状态:
   - 选择所有可能的动作，进行测试:
     - 选择Q值最大的状态进行Q值更新，计算公式如下所示:

$$\mathbf{Q}(state, action) = \mathbf{R}(state, action) + \gamma \cdot Max\left[\mathbf{Q}(next\ state, all\ actions)\right]$$

如果Q值稳定了，结束循环，否则继续迭代。

Q学习算法

初始状态选择为B

$$\mathbf{Q} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

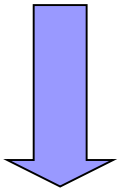
$$\mathbf{R} = \begin{array}{c} state \backslash action \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} - & - & - & - & 0 & - \\ - & - & - & 0 & - & 100 \\ - & - & - & 0 & - & - \\ - & 0 & 0 & - & 0 & - \\ 0 & - & - & 0 & - & 100 \\ - & 0 & - & - & 0 & 100 \end{bmatrix} \end{array}$$

1次迭代

在B状态下，我们随机选择一个动作，根据收益最大，我们选择为F，则有：

$$\mathbf{Q}(state, action) = \mathbf{R}(state, action) + \gamma \cdot Max\left[\mathbf{Q}(next\ state, all\ actions)\right]$$

$$\mathbf{Q}(B,F) = \mathbf{R}(B,F) + 0.8 \cdot Max\left\{\mathbf{Q}(F,B), \mathbf{Q}(F,E), \mathbf{Q}(F,F)\right\} = 100 + 0.8 \cdot 0 = 100$$



$$\mathbf{Q} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

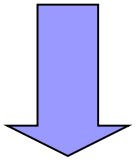
$$\mathbf{R} = \begin{array}{c} state \setminus action \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} - & - & - & - & 0 & - \\ - & - & - & 0 & - & 100 \\ - & - & - & 0 & - & - \\ - & 0 & 0 & - & 0 & - \\ 0 & - & - & 0 & - & 100 \\ - & 0 & - & - & 0 & 100 \end{bmatrix} \end{array}$$

2次迭代

随机选择一个状态D，根据回报最大，选择的动作为B，则可以更新Q如下：

$$\mathbf{Q}(state, action) = \mathbf{R}(state, action) + \gamma \cdot Max\left[\mathbf{Q}(next\ state, all\ actions)\right]$$

$$\mathbf{Q}(D,B) = \mathbf{R}(D,B) + 0.8 \cdot Max\left\{\mathbf{Q}(B,D), \mathbf{Q}(B,F)\right\} = 0 + 0.8 \cdot Max\left\{0, 100\right\} = 80$$



$$\mathbf{Q} = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 80 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

$$\mathbf{R} = \begin{array}{c} state \setminus action \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} - & - & - & - & 0 & - \\ - & - & - & 0 & - & 100 \\ - & - & - & 0 & - & - \\ - & 0 & 0 & - & 0 & - \\ 0 & - & - & 0 & - & 100 \\ - & 0 & - & - & 0 & 100 \end{bmatrix} \end{array}$$

多次迭代之后

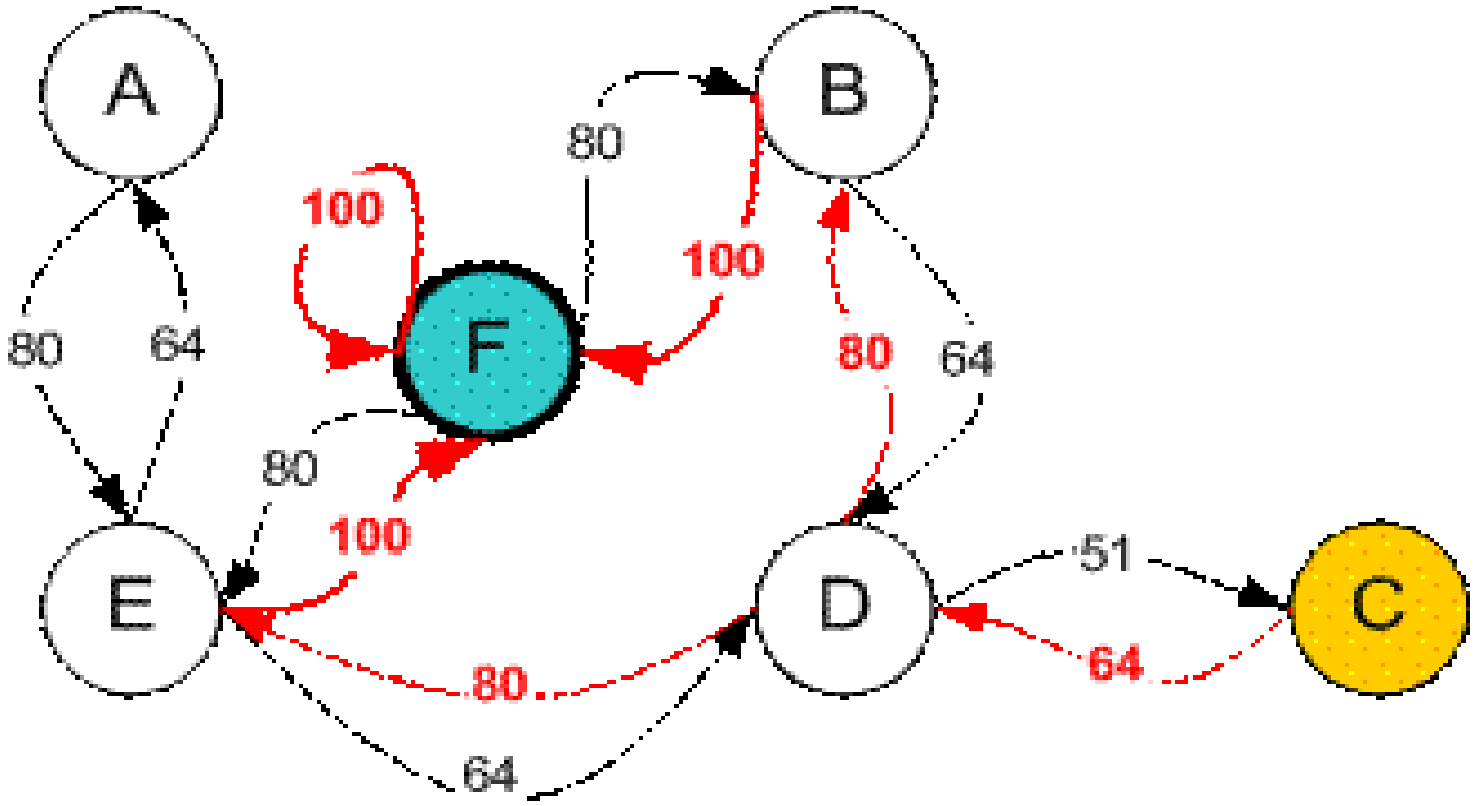
Q 矩阵达到稳定状态

$$\mathbf{Q} = \begin{array}{c} state \backslash action \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} - & - & - & - & 400 & - \\ - & - & - & 320 & - & 500 \\ - & - & - & 320 & - & - \\ - & 400 & 256 & - & 400 & - \\ 320 & - & - & 320 & - & 500 \\ - & 400 & - & - & 400 & 500 \end{bmatrix} \end{array}$$

规范化

$$\tilde{\mathbf{Q}} = \begin{array}{c} state \backslash action \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{bmatrix} - & - & - & - & 80 & - \\ - & - & - & 64 & - & 100 \\ - & - & - & 64 & - & - \\ - & 80 & 51 & - & 80 & - \\ 64 & - & - & 64 & - & 100 \\ - & 80 & - & - & 80 & 100 \end{bmatrix} \end{array}$$

基于Q矩阵，可以从任何一个状态开始，寻优找到路径：



请同学回答：从C开始，机器人如何走出去房间

课堂练习

- 给出R矩阵 –输入
- 计算Q 矩阵-输出