多智能体强化学习实训
Multi-agent Reinforcement Learning

Lecture 1:
Dynamic Programming

教师：张寅 zhangyin98@zju.edu.cn

助教：邓悦 devindeng@zju.edu.cn

王子瑞 ziseoiwong@zju.edu.cn

李奕澄 yichengli@zju.edu.cn

浙江大学计算机学院

预期目标

培养多智能体强化学习方面技术人才：

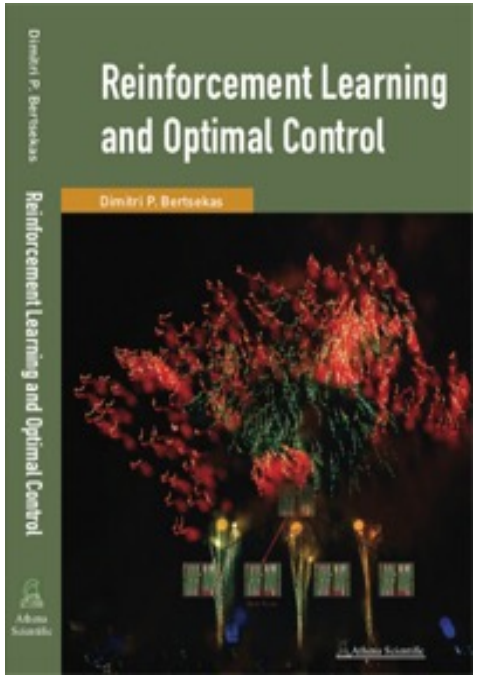
- 掌握强化学习算法思想、当前前沿方法与优化思路。
- 实训基于深度网络的强化学习算法、实现技术。
- 掌握腾讯“开悟”多智能体开放研究平台。
- 在贪食蛇等环境上实训多智能体算法。

课程内容安排

时间	上午	下午
7月1日	第一讲 动态规划理论，Bellman公式，策略评估与策略优化理论	配置强化学习环境，安装gym，完成案例CartPole环境的运行、完成腾讯koh环境部署
7月2日	第二讲 基于表格的Q-learning算法，SARSA算法，eligibility trace的应用	基于gym，测试Q-learning和SARSA算法的表现与区别
7月3日	第三讲 深度学习基础，包含损失函数、梯度回传等知识	安装pytorch，并完成基础的回归、分类任务的网络训练
7月4 日	第四讲 深度强化学习I：基于价值的DQN与基于DQN的算法	基于gym，完成DQN算法实现，并在Freeway等环境测试
7月5日	第五讲 深度强化学习II：基于策略的Reinforce算法、AC算法的讲解、以及A2C、A3C等算法	基于gym，完成基础AC算法的实现
7月8日	第六讲 深度强化学习III：先进算法TRPO、PPO、DDPG等算法原理与实现	基于gym，实现PPO算法
7月9日	第七讲 多智能体强化学习I：基于价值的QMIX算法以及改进算法	在koh 1v1环境中实现PPO算法
7月10日	第八讲 多智能体强化学习II：基于策略的MADDPG、MAPPO算法实现	实现QMIX算法讲解
7月11日	第九讲 多智能体强化学习研究展望	实现MAPPO算法讲解
7月12日	大作业交流分享	大作业答疑

参考图书


Reinforcement Learning
强化学习
（第2版）


Dimitri P. Bertsekas
Reinforcement Learning and Optimal Control
Dimitri P. Bertsekas
Reinforcement Learning and Optimal Control


TURING
深度强化学习
王树森
黎彧君
张志华
著
内容新颖，涵盖近10年来最重要的DRL方法
图多易懂，原创140多幅彩图解释原理与实现
论文导读，136篇参考文献 = DRL论文阅读清单
系列视频课全网播放量
100万次+
超轻松入门DRL
中国工信出版集团
人民邮电出版社

课程考核

- 平时（50%）
  - 5次平时作业完成情况
  - 参加王者荣耀竞赛
- 期末考核（50%）
  - 4人组队参加循环赛（7月10日-8月18日）
  - 8月17日前提交代码、文档、PPT
  - 8月20日左右线上汇报演示


6670/7225
682 731 0 270
14:52
36 vs 22
FPS 59
36ms
8 4 8
全队霸体
控制
净化

课程大纲

- **课程概述**
- 动态规划
- MDP建模
- 策略评估与优化

深度强化学习历史

…

2015
DQN
Double DQN
Dueling DQN
GAE
TRPO
DDPG
PER

2016
AC
A3C
A2C
ACKTR
R2D2

2017
C51
Rainbow
ES
PPO
HER
MADDPG
COMA
Alpha Zero
IMPALA

2018
SAC
TD3
QMIX
Alpha Star
OpenAI Five

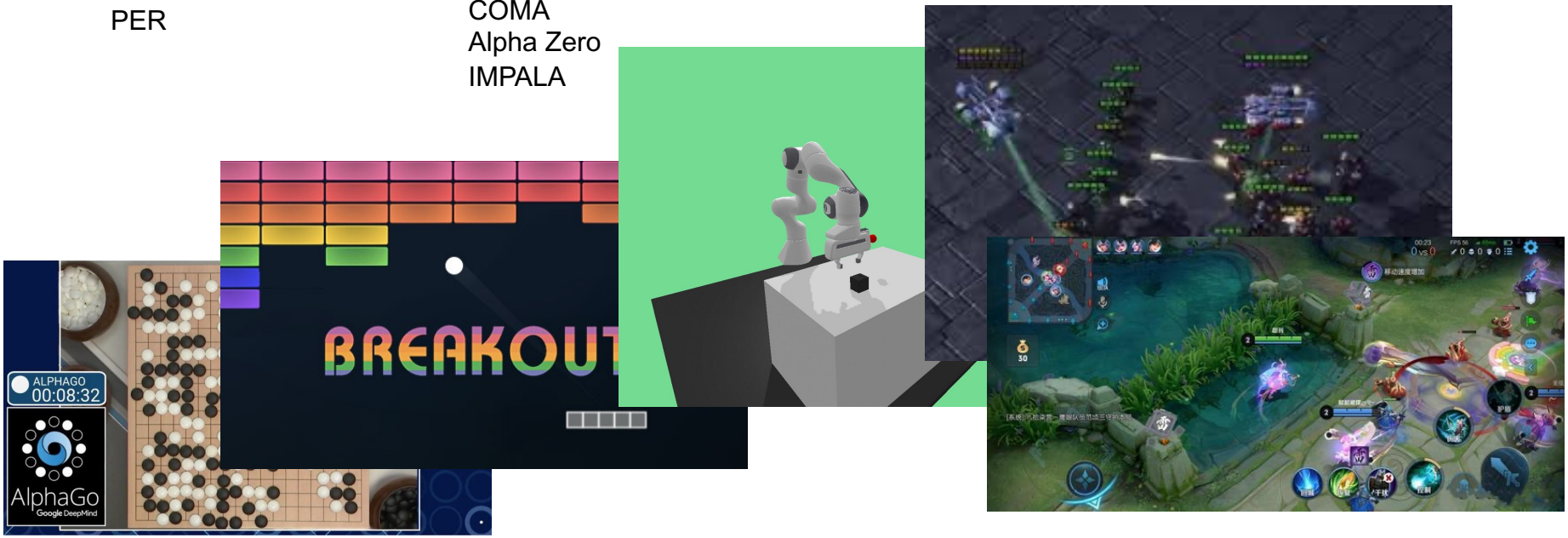
2019
ROMA
Muzero
SEED-RL
Maven

2020
Agent-57
WQMIX
Jue Wu
SCC
DOP
QPLEX

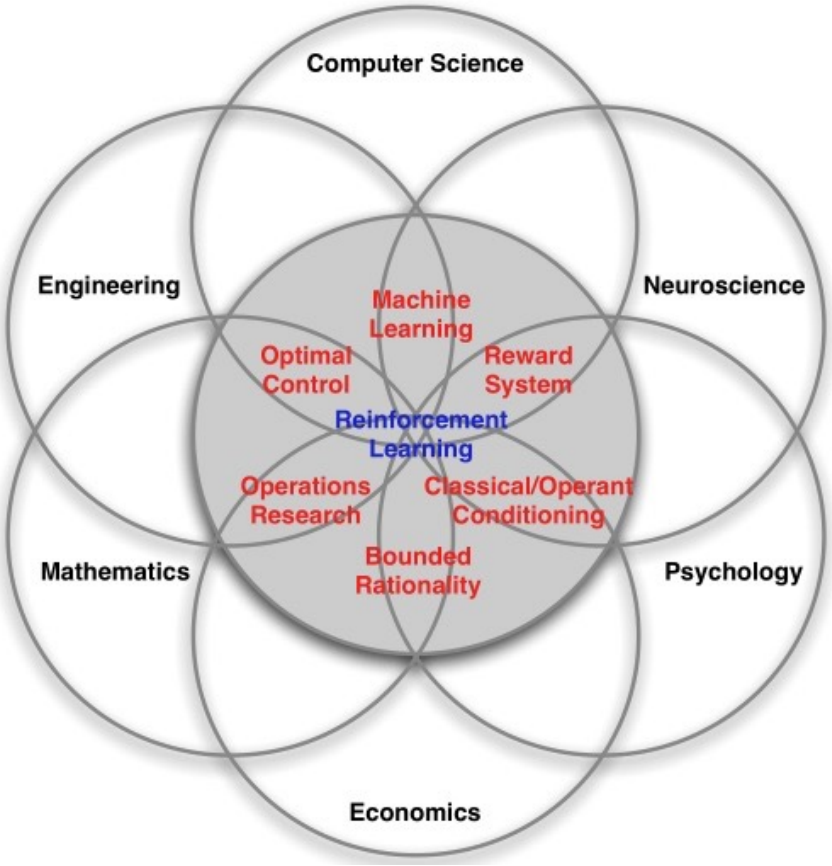
2021
Douzero
RODE
Awd3
FOP
RITT

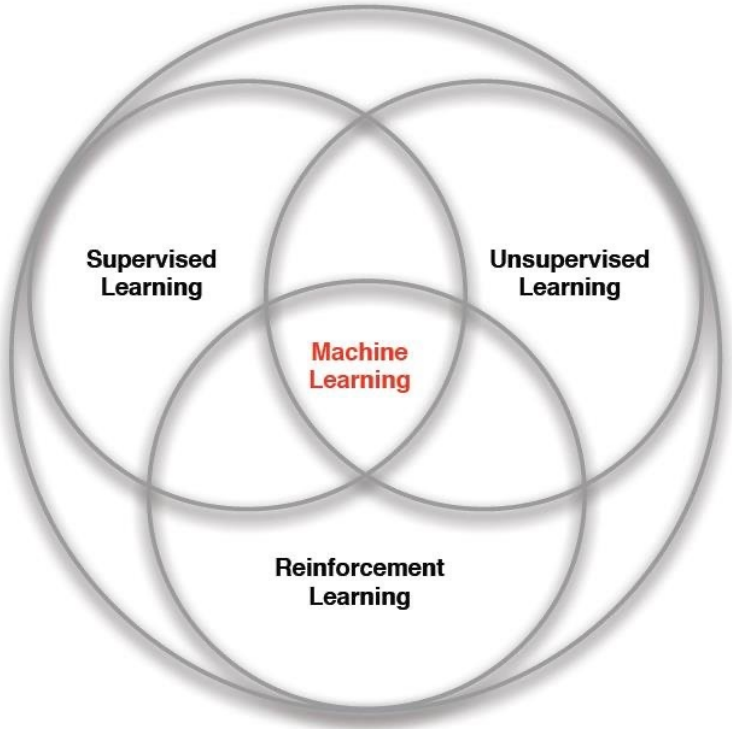
2022
MAPPO
MACPF

…



课程定位


Computer Science
Engineering
Neuroscience
Machine Learning
Optimal Control
Reward System
Reinforcement Learning
Operations Research
Classical/Operant Conditioning
Bounded Rationality
Mathematics
Psychology
Economics

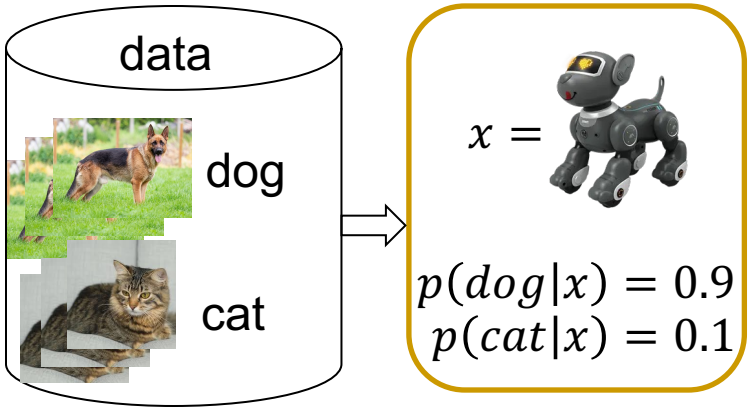

Supervised Learning
Unsupervised Learning
Machine Learning
Reinforcement Learning

强化学习

- **一种数学形式的基于学习的决策方法**
- **从经验中学习决策和控制的方法**

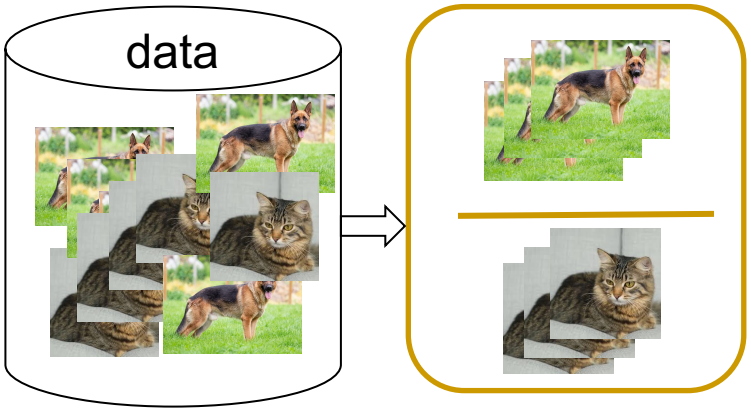
有监督学习

- 给定数据集 $D = \{(x_i, y_i)\}$
- 学习通过x预测y，$f(x) = y$
- 要求数据独立同分布
- 训练集的每一个样本都有事实标签



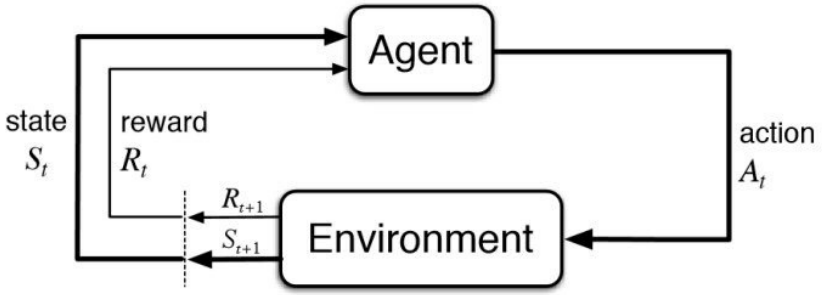
无监督学习

- 给定数据集 $D = \{(x_i)\}$
- 学习通过x预测y，$f(x) = y$
- 训练集样本没有事实标签（监督信号）



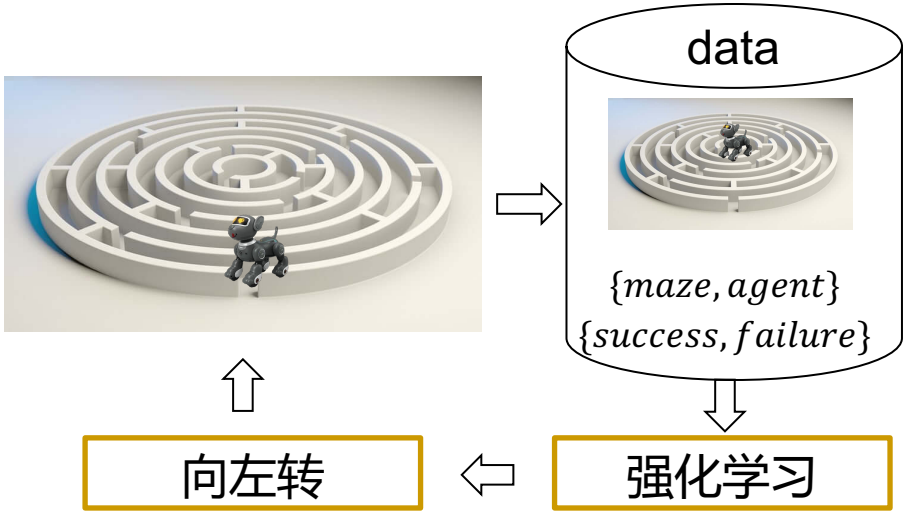
强化学习

- **一种数学形式的基于学习的决策方法**
- **从经验中学习决策和控制的方法**



强化学习

- 多轮决策
- 数据不独立，前状态影响后状态
- 事实标签不可知，只能获取决策收益或最终成功与否



- 输入：每一个时间步的状态s_t
- 输出：每一个时间步的行为a_t
- 数据：$(s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_T, a_T, r_T)$
- 目标：学习策略$\pi_\theta: s_t \to a_t$，使得最大化累计奖励 $\sum_t r_t$

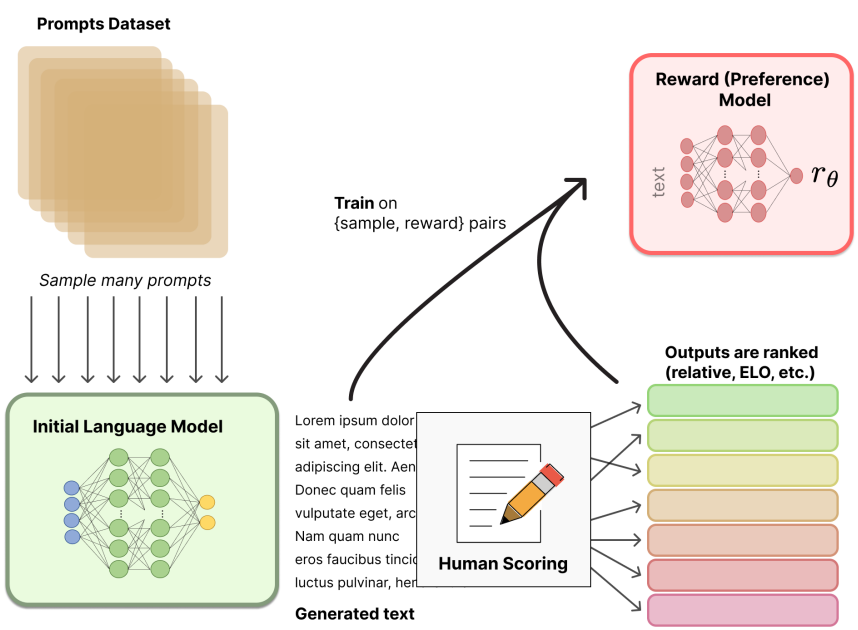
强化学习

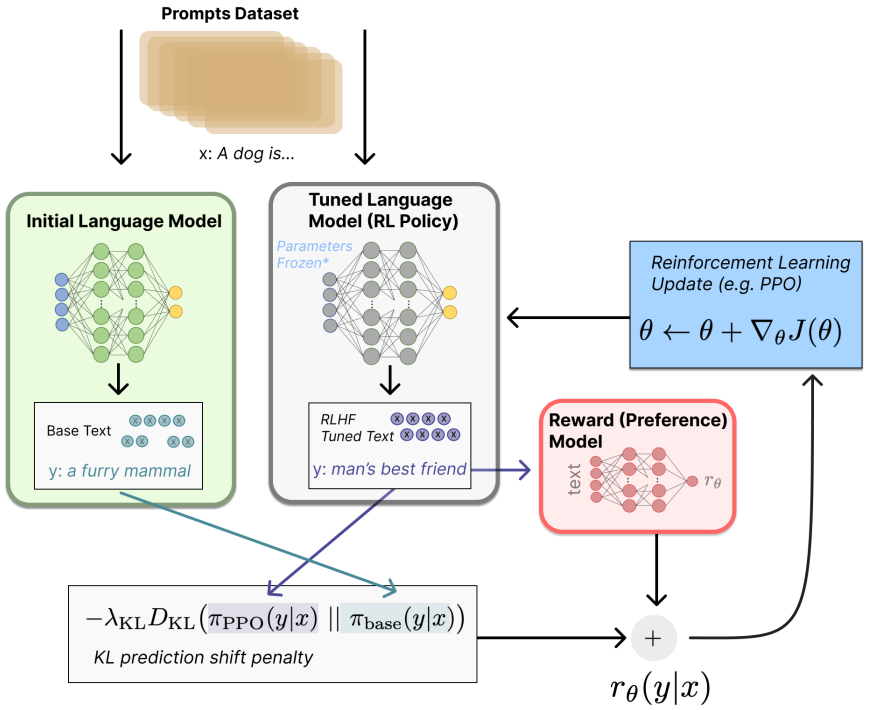
- **强化学习可以发现新的解决问题方法**

监督学习：
- 人工智能可以像人一样绘画

强化学习：
- 人工智能可以发现之前没发现的博弈策略

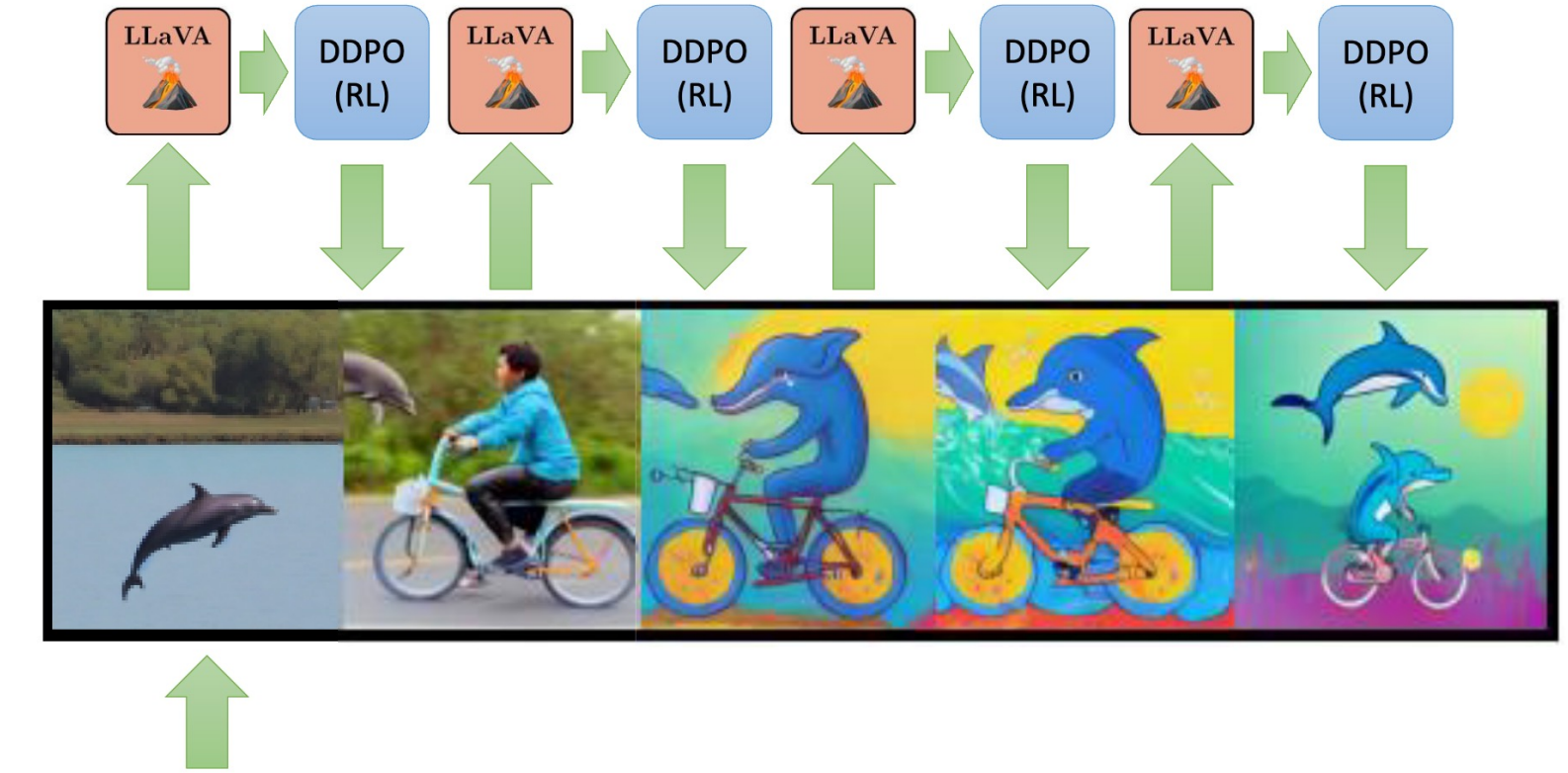
- 强化学习与大语言模型


Prompts Dataset
Sample many prompts
Initial Language Model
Train on
{sample, reward} pairs
Reward (Preference) Model
text
r_θ
Outputs are ranked (relative, ELO, etc.)
Lorem ipsum dolor
sit amet, consectet
adipiscing elit. Aen
Donec quam felis
vulputate eget, arc
Nam quam nunc
eros faucibus tincid
luctus pulvinar, her
Human Scoring
Generated text


Prompts Dataset
x: A dog is...
Initial Language Model
Base Text
y: a furry mammal
Tuned Language Model (RL Policy)
Parameters Frozen*
RLHF Tuned Text
y: man's best friend
Reinforcement Learning Update (e.g. PPO)
$\theta \leftarrow \theta + \nabla_\theta J(\theta)$
Reward (Preference) Model
text
r_θ
$-\lambda_{\mathrm{KL}} D_{\mathrm{KL}}(\pi_{\mathrm{PPO}}(y|x) \,||\, \pi_{\mathrm{base}}(y|x))$
KL prediction shift penalty
+
$r_\theta(y|x)$

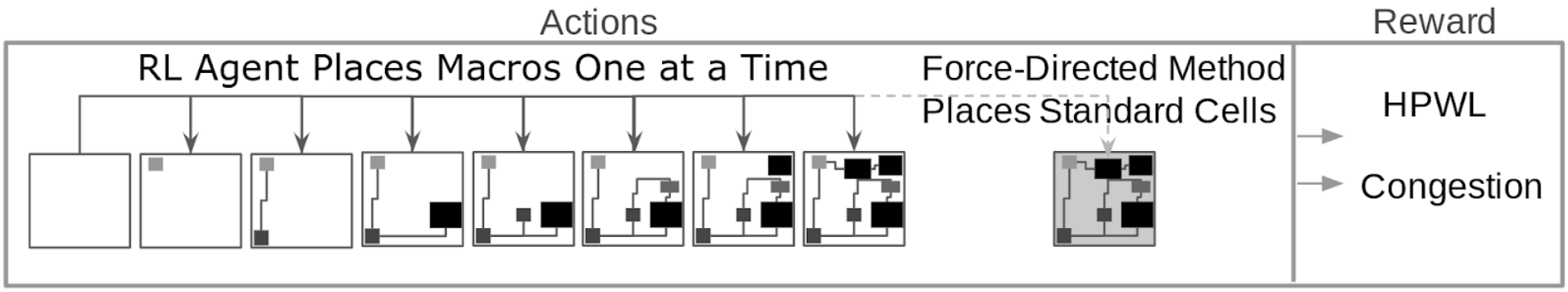
强化学习

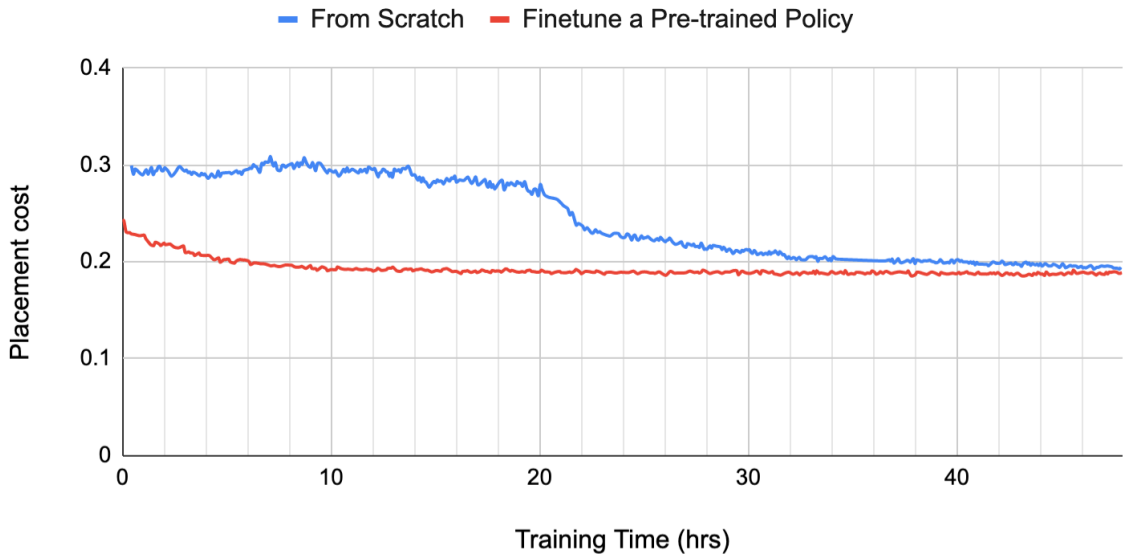
- 强化学习与图片生成



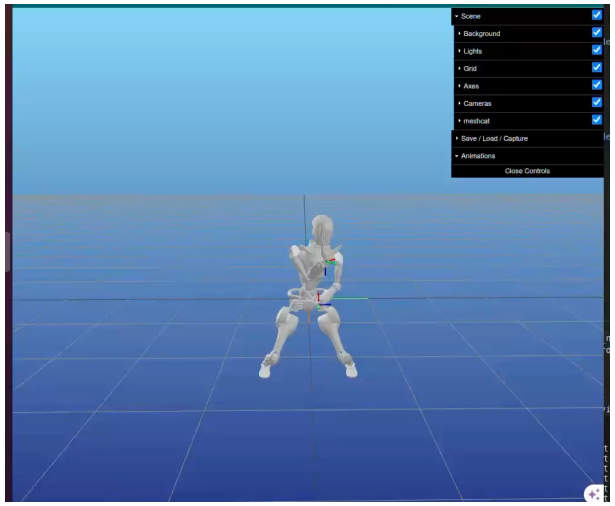
强化学习

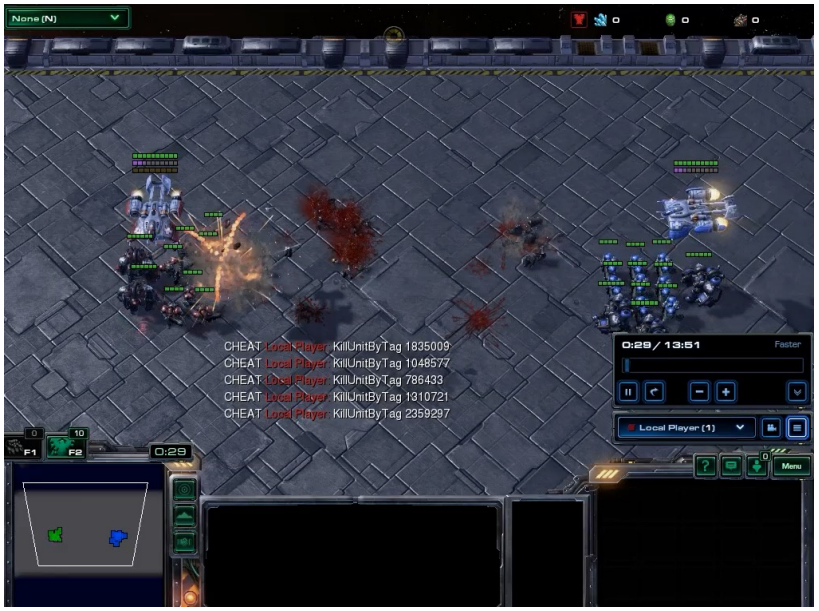
- 强化学习与芯片设计





强化学习


Scene
Background
Lights
Grid
Axes
Cameras
meshcat
Save / Load / Capture
Animations
Close Controls


None (N)
CHEAT Local Player: KillUnitByTag 1835009
CHEAT Local Player: KillUnitByTag 1048577
CHEAT Local Player: KillUnitByTag 786433
CHEAT Local Player: KillUnitByTag 1310721
CHEAT Local Player: KillUnitByTag 2359297
0:29 / 13:51
Faster
Local Player (1)
F1
F2
0:29
Menu


MARIO
000000
×00
WORLD
1-2
TIME
400

课程大纲

腾讯开悟

- 课程概述
- **动态规划**
- MDP建模
- 策略评估与优化

动态规划

腾讯开悟

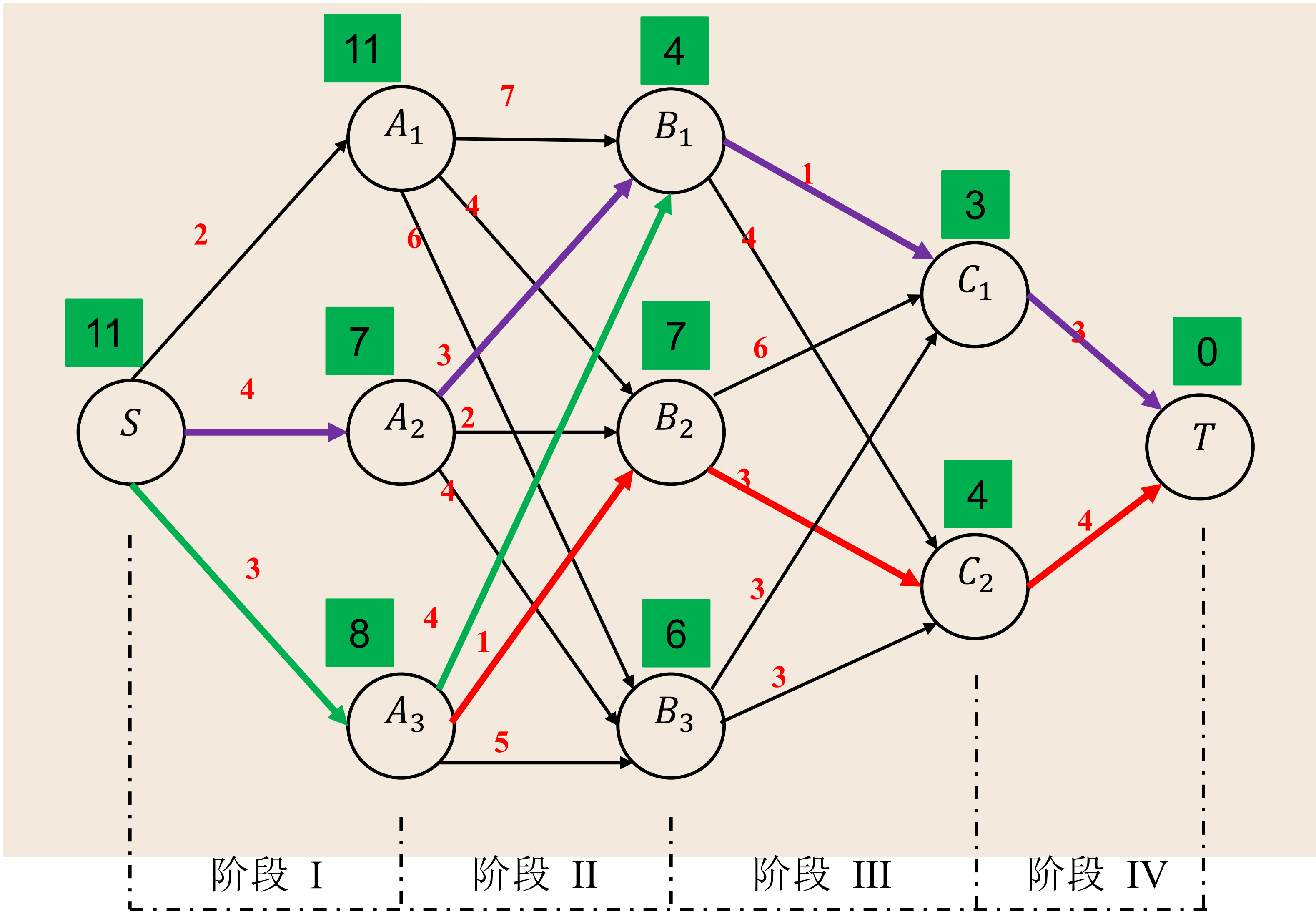
- 动态规划将复杂的多阶段决策问题分解为一系列简单的、离散的单阶段决策问题，采用顺序求解方法，通过求解一系列小问题达到求解整个问题的目的。

- 动态规划的各个决策阶段不但要考虑本阶段的决策目标，还要兼顾整个决策过程的整体目标，从而实现整体的最优决策。

动态规划

- 通常多阶段决策过程的发展是通过状态的一系列变换来实现的。而适合于用动态规划的方法求解的问题是具备无后效性的决策过程。
- 无后效性即马尔科夫性（Markov Property），系统从某个阶段往后的发展只取决于当前状态，而与以前经历的状态与决策无关。
- 当前的状态是后面过程发展的初始条件。

最短路径案例



11
4
3
0
11
7
7
8
6
4
A_1
A_2
A_3
B_1
B_2
B_3
C_1
C_2
S
T
2
4
3
7
4
6
3
2
4
4
1
5
1
4
6
3
3
3
3
4
阶段 I
阶段 II
阶段 III
阶段 IV

动态规划

- 动态规划问题的主要要素为：
  - 一个状态取决于控制的离散时间动态系统。这里我们假设有n个状态，分别表示为1, 2, …n，加终止状态0。在状态i的时候，控制行为可以从有限集$U(i)$中获取。同时在状态i，采取行为u，进入状态j的概率可以表示为$p_{i,j}(u)$。
  - 一个根据状态和行为所决定的可累计的奖励。在第k步决策中，我们设定$\alpha^k g(i,u,j)$为奖励，其中g是奖励函数，α是惩罚长远奖励的折扣因子。

动态规划

- 动态规划得到结果是策略（policies），即$\pi = (\mu_0, \mu_1, \ldots)$，其中每一个$\mu_k$都是一个从状态到行为控制的映射$\mu_k(i) \in U(i)$。如果策略$\pi$已经确定，状态$i_k$序列即为马尔科夫链：$P(i_{k+1} = j | i_k = i) = p_{ij}\big(\mu_k(i)\big)$。
- 对于有限视野的问题，即奖励在未来的有限步数内累积（N），对于一个策略π和初始状态i，其期望累计奖励为（其中$\alpha^N G(i_N)$是视野中最终状态的终止奖励）

$$J_N^{\pi}(i) = E[\alpha^N G(i_N) + \sum_{k=0}^{N-1} \alpha^k g(i_k, \mu_k(i_k), i_{k+1}) | i_0 = i]$$

- 同时状态i的最优的N阶段累计奖励为：$J_N^*(i) = max_{\pi} J_N^{\pi}(i)$

动态规划

- 一阶段决策（$N = 1$）

$$J_1^*(i) = \min_{\mu_0} \sum_{j=1}^{n} p_{ij}(\mu_0(i))(g(i, \mu_0, j) + \alpha G(j))$$

$$J_1^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + \alpha G(j))$$

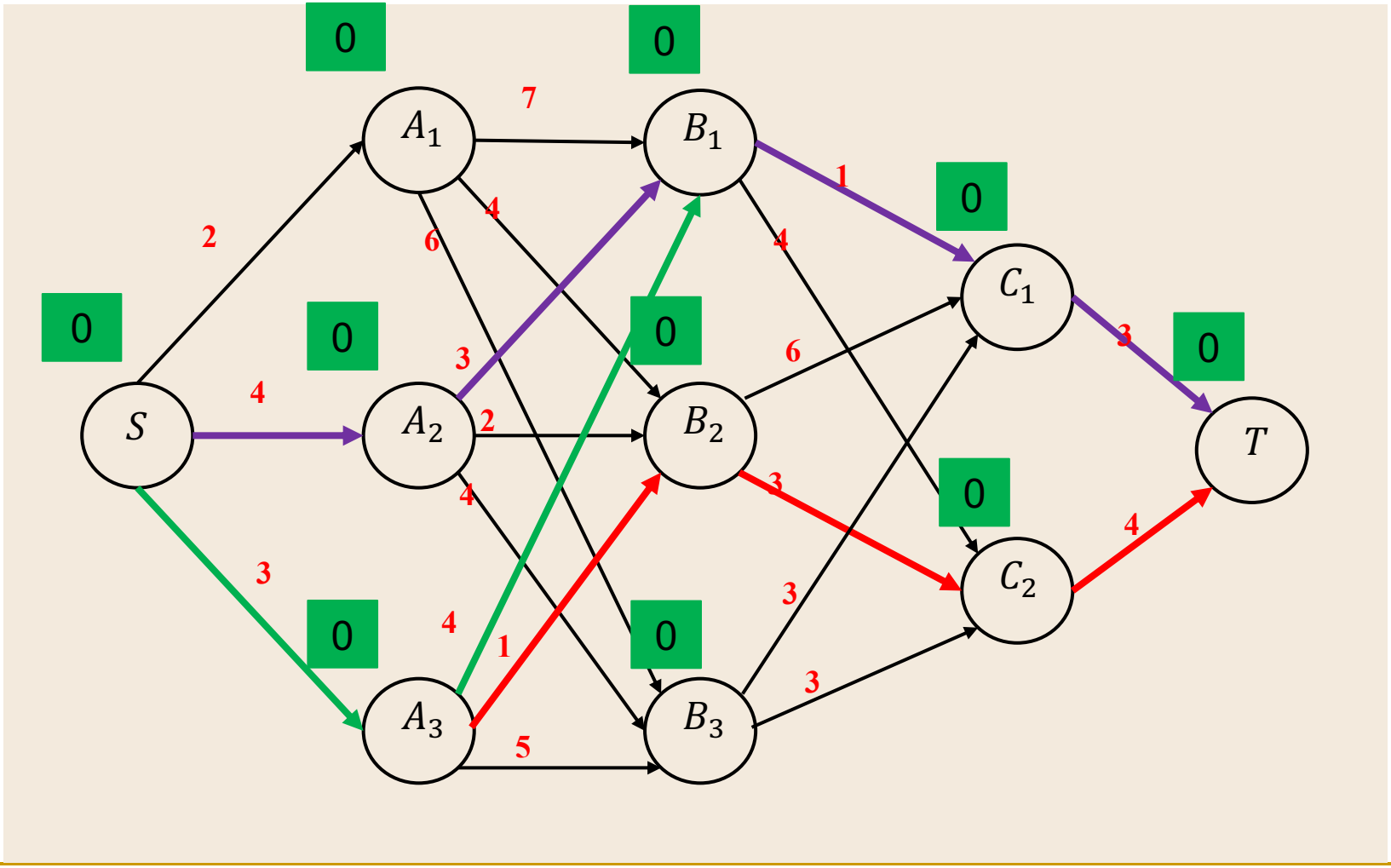
- …
- k阶段决策（$N = k$）

$$J_0^*(i) = G(i)\,, \qquad i = 1, \ldots n$$

$$J_k^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i, u, j) + \alpha J_{k-1}^*(j))$$

最短路径案例

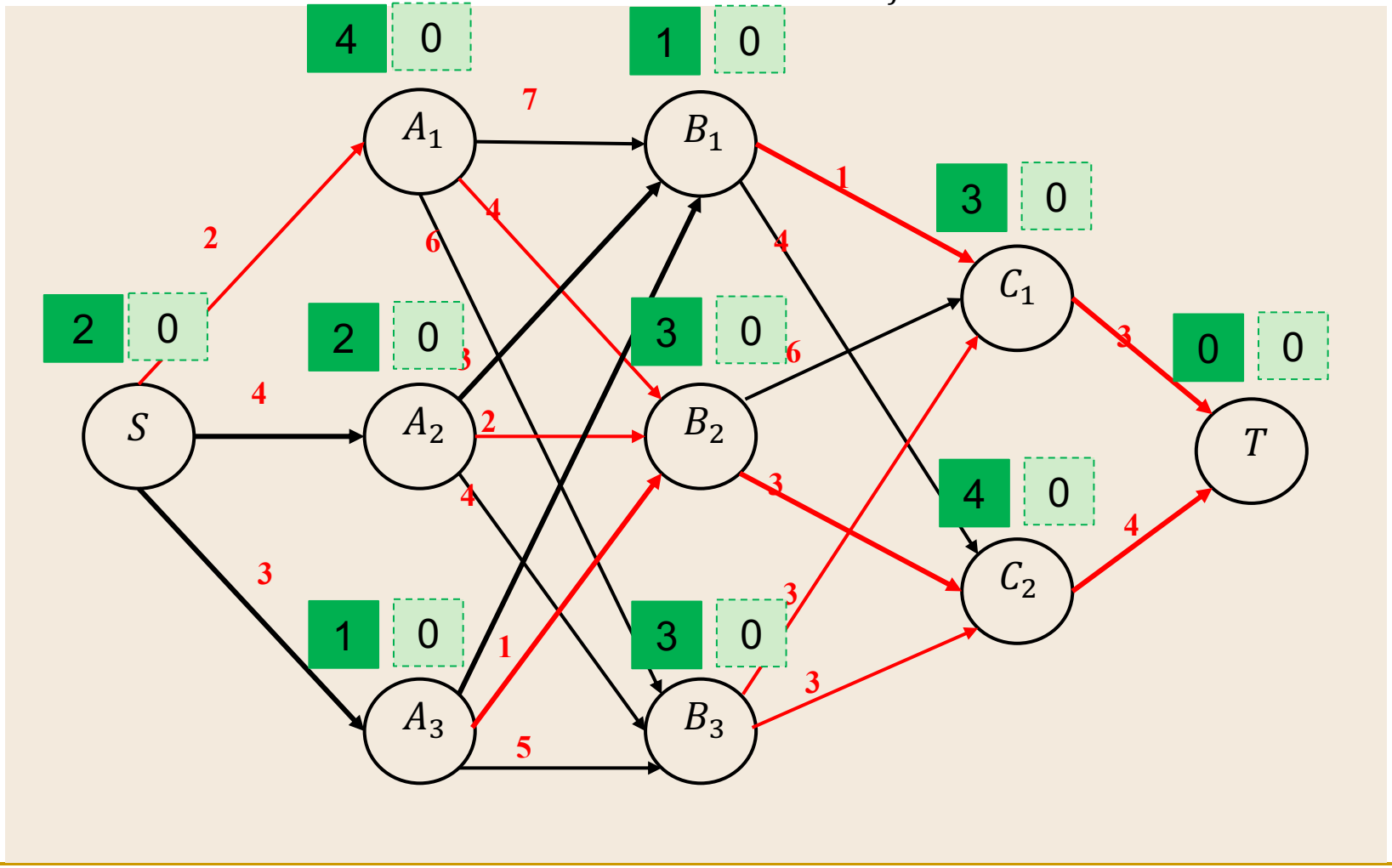
- 初始化



最短路径案例

- 第一轮迭代

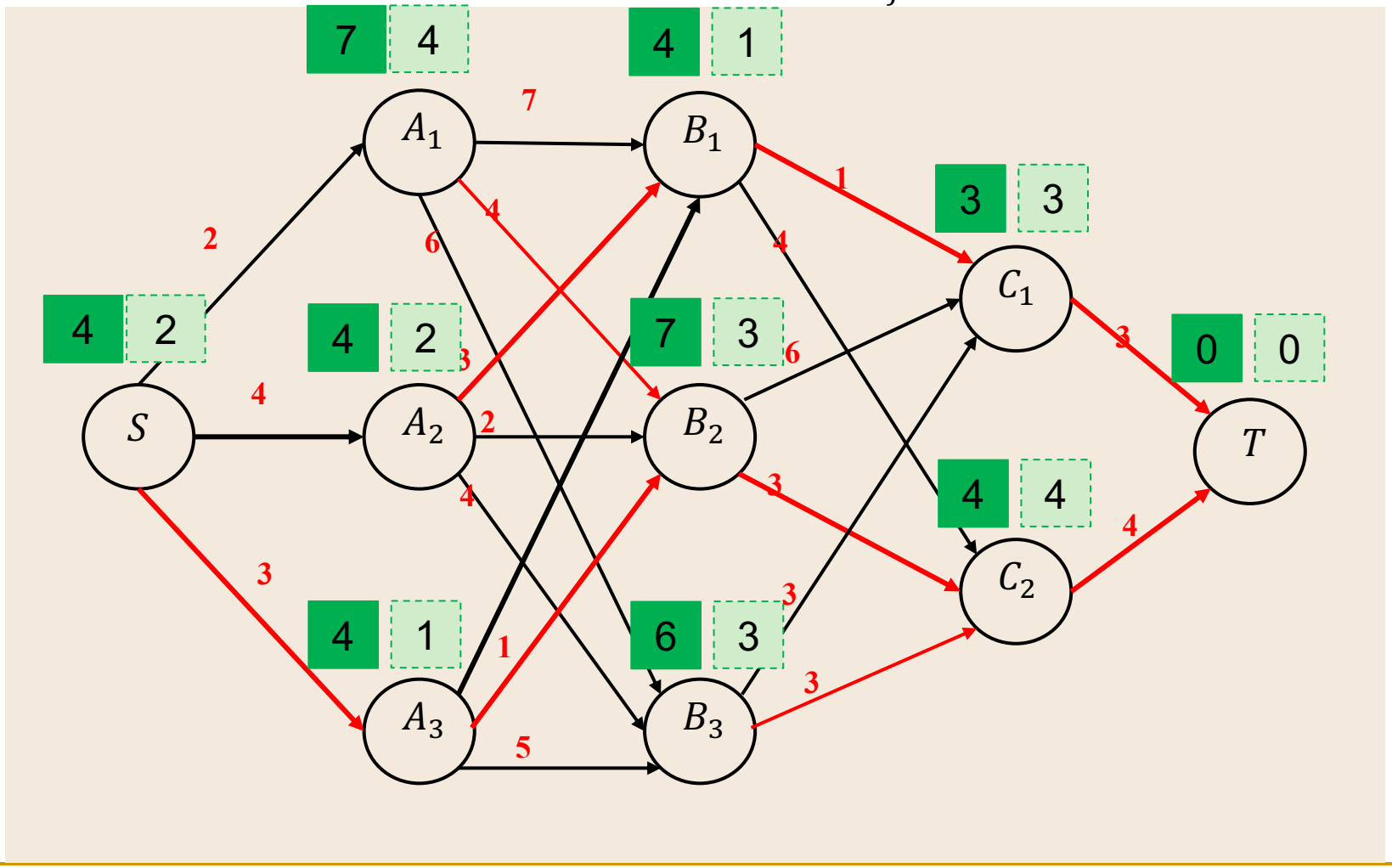
$$J_k^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j) + \alpha J_{k-1}^*(j))$$



最短路径案例

- 第二轮迭代

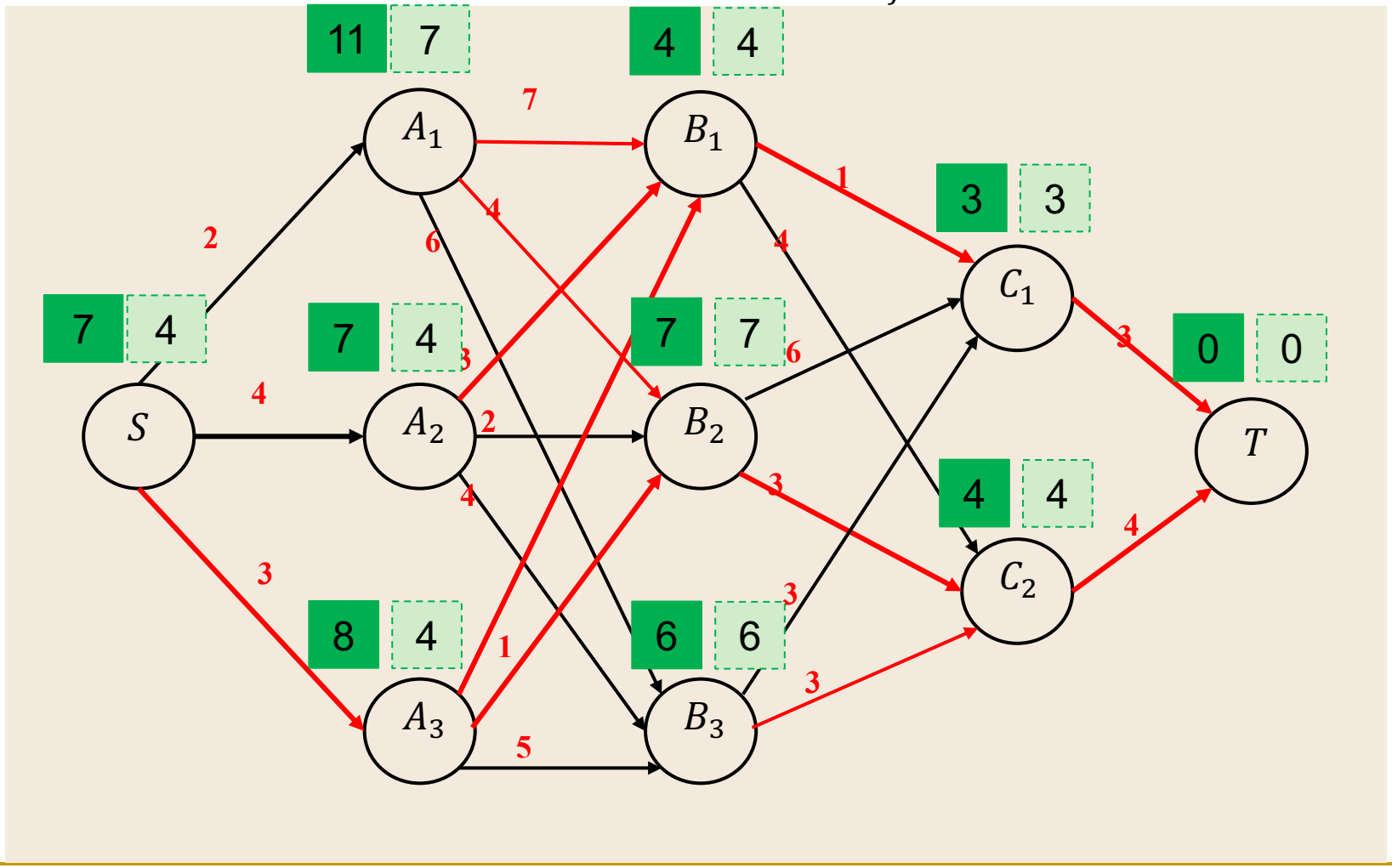
$$J_k^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j) + \alpha J_{k-1}^*(j))$$



最短路径案例

- 第三轮迭代

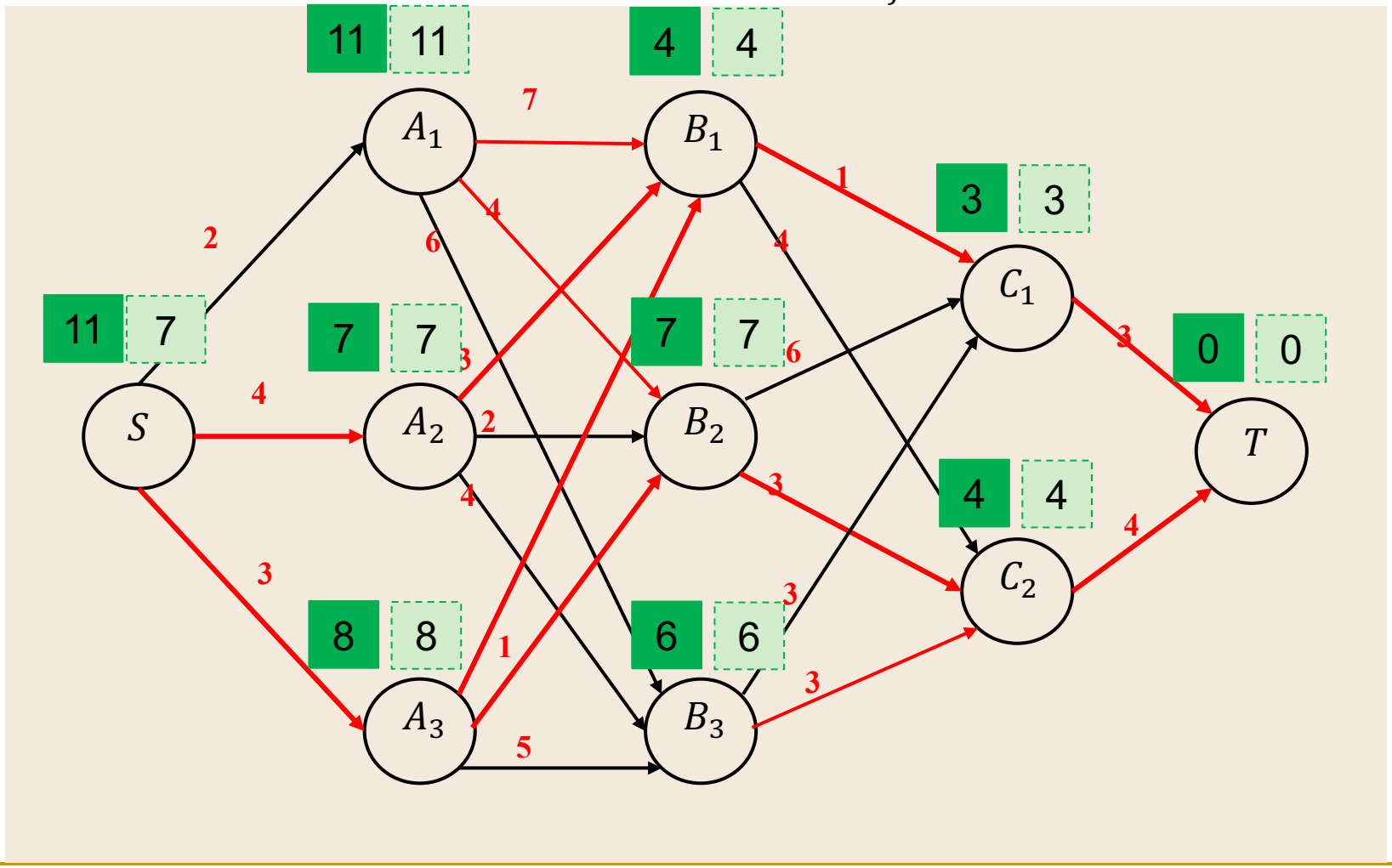
$$J_k^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j) + \alpha J_{k-1}^*(j))$$



最短路径案例

- 第四轮迭代

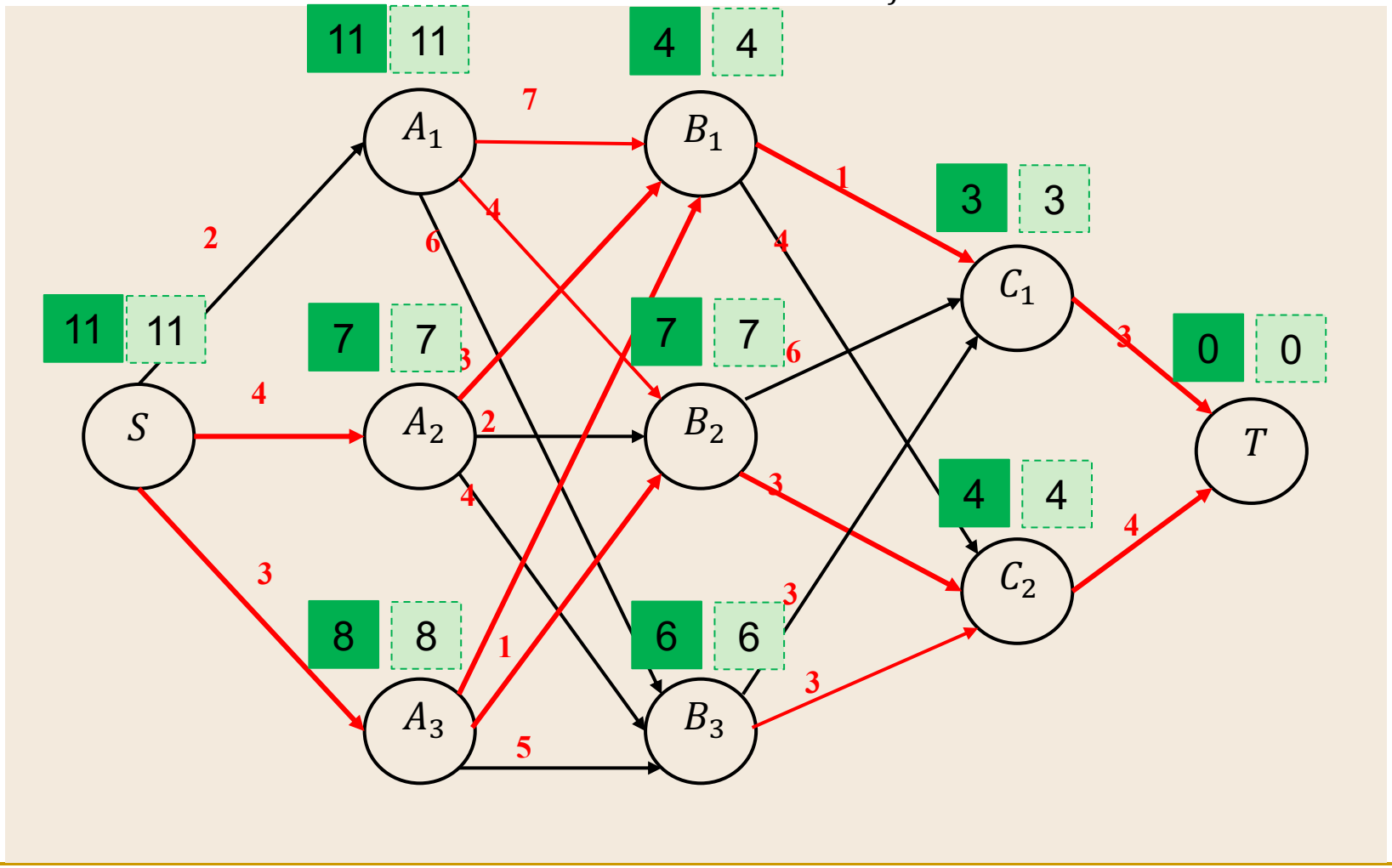
$$J_k^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j) + \alpha J_{k-1}^*(j))$$



最短路径案例

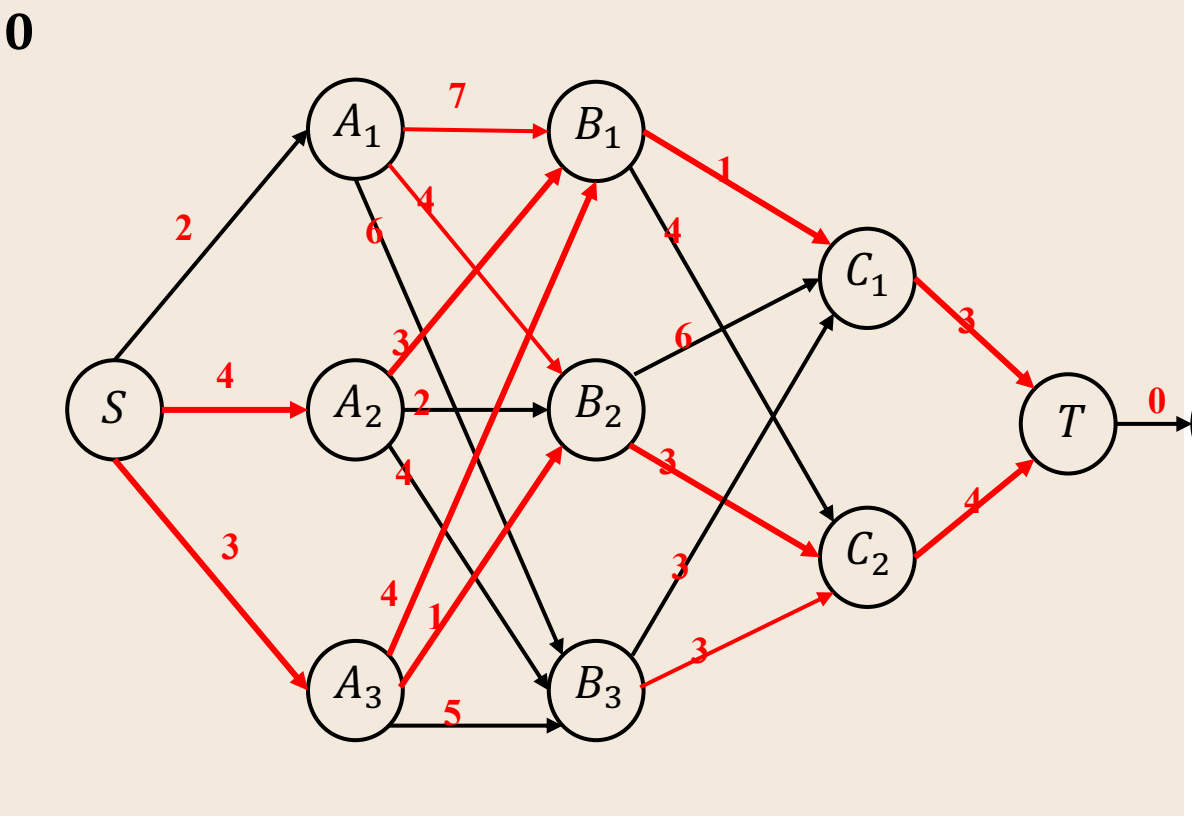
- 第五轮迭代…第N轮迭代

$$J_k^*(i) = \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)(g(i,u,j) + \alpha J_{k-1}^*(j))$$



动态规划–最短路径

- 为建模方便，我们增加结束节点，该节点无奖励，且自成环。
  - $p_{oo}(u) = 1$
  - $g(0, u, 0) = 0$
  - $\alpha = 1$



一般理论

给定一个策略μ, 如果采用该策略，不论初始状态，智能体至多进行n轮决策后进入终止状态的概率为正，则该策略为合理（proper）策略。即：

$$\rho_\mu = \max_{i=1,\dots n} P(i_n \neq 0 | i_0 = i, \mu) < 1$$

给定合理策略μ:

$$P(i_{2n} \neq 0 | i_0 = i, \mu) = P(i_{2n} \neq 0 | i_n \neq 0, i_0 = i, \mu) \times P(i_n \neq 0 | i_0 = i, \mu) \leq \rho_\mu^2$$

即：无限轮迭代后，进入终止状态是必然结果。

$$P(i_k \neq 0 | i_0 = i, \mu) \leq \rho_\mu^{k/n}$$

一般理论

动态规划速记符号：

- 给定一个向量$J = \big(J(1), J(2), \dots, J(n)\big)$，我们用$TJ$表示向$J$向量应用一次动态规划过程，即：

$$(TJ)(i) = \min_{u \in U(i)} \sum_{j=0}^{n} p_{ij}(u)(g(i,u,j) + J(j)) \ , \qquad i = 1, \dots n$$

- 如果给定策略μ：

$$\big(T_\mu J\big)(i) = \sum_{j=0}^{n} p_{ij}(\mu(i))(g(i,\mu(i),j) + J(j)), \qquad i = 1, \dots n$$

- 向向量J应用k次动态规划表示为$T^k J$，且：$T^k J(i) = (T(T^{k-1}J))(i)$，$T^0 J(i) = J(i)$，　T_μ同理。

一般理论

一般来说T和T_μ具备如下特性：

- 对任意n维的向量J和$\bar{J}$，假设$\mathrm{J}(\mathrm{i}) \leq \bar{J}(i)$，则对任意策略$\mu$:

$$T^k J(i) \leq (T^k \bar{J})(i)$$

$$T_\mu^k J(i) \leq (T_\mu^k \bar{J})(i)$$

- 对任意k, 向量J，策略μ，以及正值r:

$$T^k(J + re)(i) \leq \left(T^k J\right)(i) + r, \qquad i = 1, \ldots, n$$

$$T_\mu^k(J + re)(i) \leq \left(T_\mu^k J\right)(i) + r, \qquad i = 1, \ldots, n$$

如果r为负值，则不等号相反。

一般理论

在最短路径的问题中，假设1）至少有一个合理的策略，以及2）假设每一个不合理的策略都至少有1个状态的成本是无限大，则：

a） 最优的未来成本（奖励）J^*向量是有限元，并满足：

$$J^* = TJ^*$$

且J^*是迭代式$J = TJ$的唯一解；

b） 给定每一个向量J，$\lim_{k\to\infty} T^k J = J^*$；

c） 同理对于策略μ，和T_μ，上述规则也同样适用。

收缩映射

Definition 3.1 Let (X, d) be a metric space. A mapping $T : X \to X$ is a *contraction mapping*, or *contraction*, if there exists a constant c, with $0 \leq c < 1$, such that

$$d(T(x), T(y)) \leq c\, d(x, y) \tag{3.1}$$

for all $x, y \in X$.

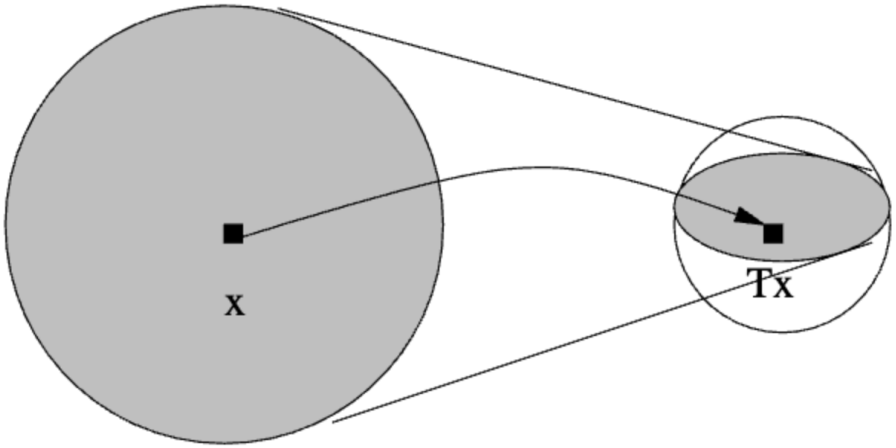


Fig. 3.1 T is a contraction.

https://www.scribd.com/document/8340854/The-Contraction-Mapping-Theorem

一般理论

在动态规划问题中，映射T通常是加权最大范数的收缩映射，即存在一个正向量$\xi = (\xi(1), \xi(2), \ldots \xi(n))$和一个标量$\beta < 1$:

$$\|TJ - T\bar{J}\|_\xi \leq \beta \|J - \bar{J}\|_\xi$$

且$\|.\|_\xi$被定义为：

$$\|J\|_\xi = \max_{i=1,\ldots n} \frac{|J(i)|}{\xi(i)}$$

理论证明

Proposition 2.2: Suppose that all stationary policies are proper. Then, there exists a vector ξ with positive components such that the mapping T and the mappings T_μ, for all stationary policies μ, are contraction mappings with respect to the weighted maximum norm $\|\cdot\|_\xi$. In particular, there exists some $\beta < 1$ such that

$$\sum_{j=1}^{n} p_{ij}(u)\xi(j) \leq \beta\xi(i), \qquad \forall\, i,\, u \in U(i).$$

理论证明

Proof: We first define the vector ξ as the solution of a certain DP problem, and then show that it has the required property. Consider a new stochastic shortest path problem where the transition probabilities are the same as in the original, but the transition costs are all equal to -1 (except at the termination state 0, where the self-transition cost is 0). Let $\hat{J}(i)$ be the optimal cost-to-go from state i in this new problem. By Prop. 2.1(a), we have for all $i = 1, \ldots, n$, and stationary policies μ,

$$\begin{aligned} \hat{J}(i) &= -1 + \min_{u \in U(i)} \sum_{j=1}^{n} p_{ij}(u)\hat{J}(j) \\ &\leq -1 + \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)\hat{J}(j) \end{aligned} \tag{2.11}$$

Define

$$\xi(i) = -\hat{J}(i), \qquad i = 1, \ldots, n.$$

Then for all i, we have $\xi(i) \geq 1$, and for all stationary policies μ, we have from Eq. (2.11),

$$\sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)\xi(j) \leq \xi(i) - 1 \leq \beta\xi(i), \qquad i = 1, \ldots, n, \tag{2.12}$$

where β is defined by

$$\beta = \max_{i=1,\ldots,n} \frac{\xi(i) - 1}{\xi(i)} < 1$$

理论证明

For any stationary policy μ, any state i, and any vectors J and $\overline{J}$, we have using Eq. (2.12),

$$\begin{aligned}\left|(T_\mu J)(i) - (T_\mu \overline{J})(i)\right| &= \left|\sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)\big(J(j) - \overline{J}(j)\big)\right| \\ &\leq \sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)\left|J(j) - \overline{J}(j)\right| \\ &\leq \left(\sum_{j=1}^{n} p_{ij}\big(\mu(i)\big)\xi(j)\right)\left(\max_{j=1,\ldots,n} \frac{\left|J(j) - \overline{J}(j)\right|}{\xi(j)}\right) \\ &\leq \beta\xi(i) \max_{j=1,\ldots,n} \frac{\left|J(j) - \overline{J}(j)\right|}{\xi(j)}.\end{aligned}$$

Dividing both sides by $\xi(i)$ and taking the maximum over i of the left-hand side, we obtain

$$\max_{i=1,\ldots,n} \frac{\left|(T_\mu J)(i) - (T_\mu \overline{J})(i)\right|}{\xi(i)} \leq \beta \max_{j=1,\ldots,n} \frac{\left|J(j) - \overline{J}(j)\right|}{\xi(j)},$$

理论证明

so that T_μ is a contraction with respect to the weighted maximum norm $\|\cdot\|_\xi$.

The preceding calculation also yields

$$(T_\mu J)(i) \leq (T_\mu \overline{J})(i) + \beta\xi(i) \max_{j=1,\ldots,n} \frac{|J(j) - \overline{J}(j)|}{\xi(j)},$$

and by taking the minimum of both sides over μ, we obtain

$$(TJ)(i) \leq (T\overline{J})(i) + \beta\xi(i) \max_{j=1,\ldots,n} \frac{|J(j) - \overline{J}(j)|}{\xi(j)}.$$

By interchanging the roles of J and $\overline{J}$, we also have

$$(T\overline{J})(i) \leq (TJ)(i) + \beta\xi(i) \max_{j=1,\ldots,n} \frac{|J(j) - \overline{J}(j)|}{\xi(j)},$$

and by combining the last two relations, we obtain

$$\left|(TJ)(i) - (T\overline{J})(i)\right| \leq \beta\xi(i) \max_{j=1,\ldots,n} \frac{|J(j) - \overline{J}(j)|}{\xi(j)}.$$

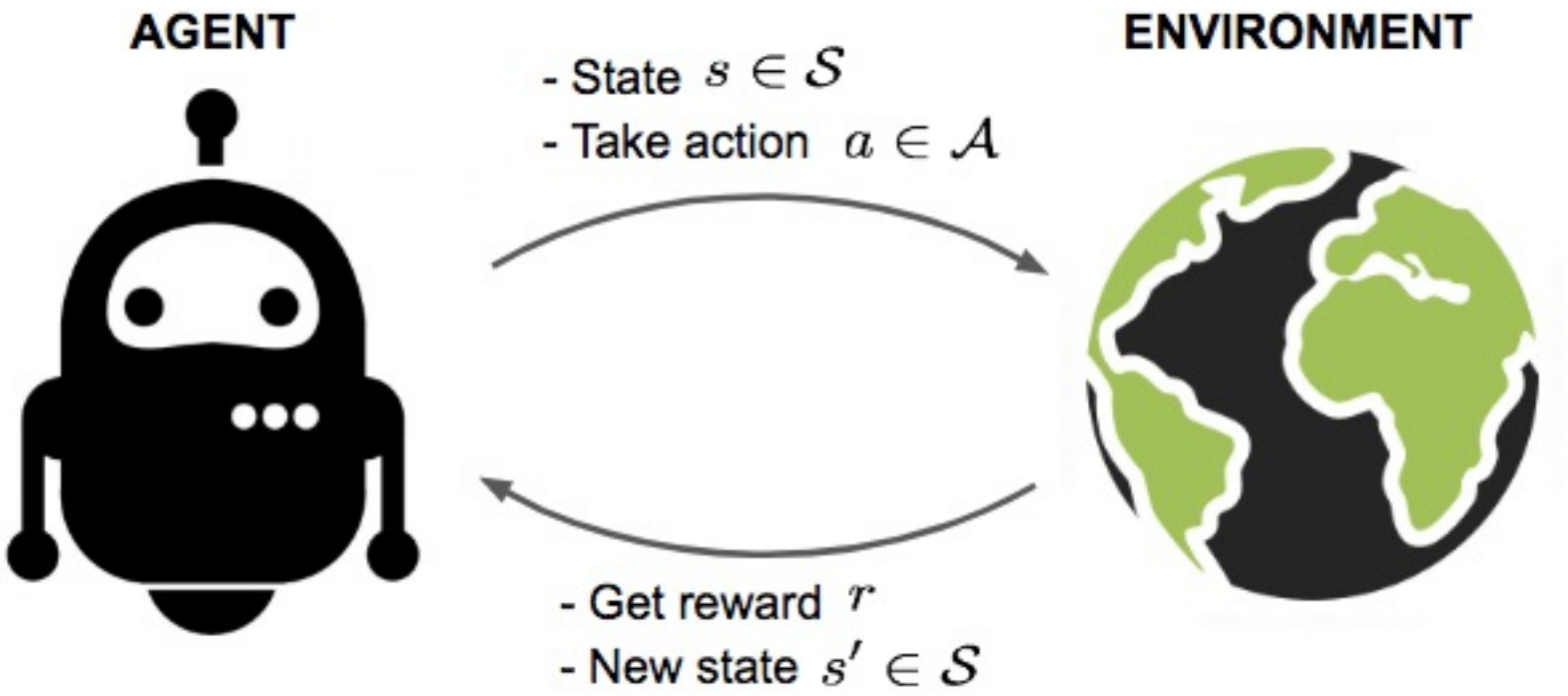
Dividing both sides by $\xi(i)$ and taking the maximum over i of the left-hand side, we obtain that T is a contraction with respect to $\|\cdot\|_\xi$. **Q.E.D.**

课程大纲

腾讯开悟

- 课程概述
- 动态规划
- **MDP建模**
- 策略评估与优化

MDP建模


AGENT
ENVIRONMENT
- State $s \in \mathcal{S}$
- Take action $a \in \mathcal{A}$
- Get reward r
- New state $s' \in \mathcal{S}$

MDP建模

智能体**(agent)**在一个环境**(environment)**中执行动作/行为**(action)**。环境如何对智能体的动作做出响应由一个**已知或未知**的模型**(model)**来定义。执行智能体可以停留在环境中的某个状态**(state)** $\boldsymbol{s \in S}$，可以通过执行某个行为/动作**(action)** $\boldsymbol{a \in A}$来从一个状态s进入到另一个状态s'。智能体会到达什么状态由**状态转移概率**$(\boldsymbol{P})$决定。智能体执行了一个动作之后，环境会给出一定的奖励**(reward)** $\boldsymbol{r \in R}$作为反馈。

MDP建模

- 几乎所有的强化学习问题可以用马尔科夫决策过程(MDPs)来描述。MDP中所有的状态都具有“马尔科夫”性，也就是未来仅依赖于当前状态，而与历史状态无关：

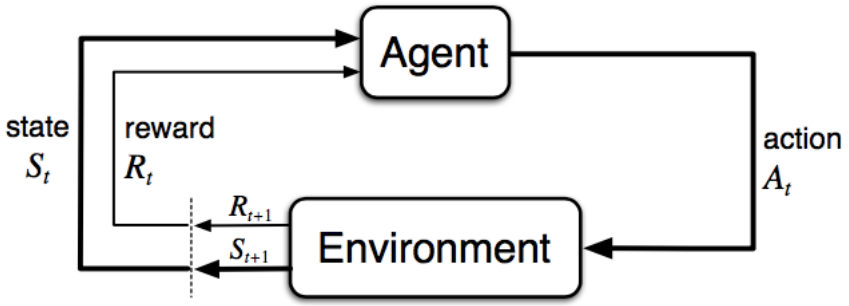
$$P(S_{t+1}|S_t) = P(S_{t+1}|S_1, \dots, S_t)$$

- 即：在给定当前状态的条件下，未来与过去条件独立，也就是当前状态包含了决定未来所需的所有信息。

MDP建模

更正式地说，马尔科夫过程由5个元素组成$M =< S, A, P, R, \gamma >$其中：

- S:状态集合
- A:行为集合
- P:状态转移函数
- R:奖励函数
- γ:未来奖励衰减系数。



在一个未知的环境中，转移概率P和奖励R由环境给出，无法获取其全部信息。

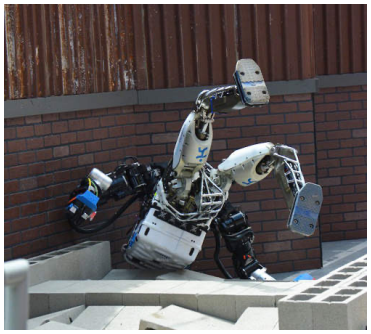
POMDP建模

部分可观测马尔科夫决策过程（Partially Observable Markov Decision Process）是带有隐藏状态的MDP，是一个带有行为决策的隐马尔科夫模型。

一个POMDP问题可以由$M =< S, A, O, P, R, Z, \gamma >$构成，其中：

- S是有限状态集合；
- A是有限行为集合；
- O为有限观测集合；
- P为状态转移函数：$P_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' | S_t = s, A_t = a]$；
- R为奖励函数：$R_s^a = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$；
- Z为基于状态S的观测函数：$Z_{s',o}^{a} = \mathbb{P}[O_{t+1} = o | S_{t+1} = s', A_t = a]$
- γ为折扣系数，对未来奖励衰减。

MDP建模



贝尔曼方程

贝尔曼方程指的是一系列的等式，它将**价值函数和动作价值函数**分解为当前直接奖励加上后续衰减后的未来奖励。

- 价值函数：

$$
\begin{aligned}
V(s) &= \mathbb{E}[G_t | S_t = s] \\
&= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ \ldots | S_t = s] \\
&= \mathbb{E}[R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \ \ldots) | \ S_t = s] \\
&= \mathbb{E}[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\
&= \mathbb{E}[R_{t+1} + \gamma V(S_{t+1}) | S_t = s]
\end{aligned}
$$

- 动作价值函数：

$$
\begin{aligned}
Q(s,a) &= \mathbb{E}[R_{t+1} \ + \gamma V(S_{t+1}) | S_t = s, A_t = a] \\
&= \mathbb{E}[R_{t+1} + \gamma \mathbb{E}_{a \sim \pi} Q(s_{t+1}, a) | S_t = s, A_t = a]
\end{aligned}
$$

贝尔曼期望方程

在强化学习中，我们更关注对于策略π的提升。迭代更新过程可以被进一步分解为基于状态和行为价值的等式。在当前的策略π的前提下：

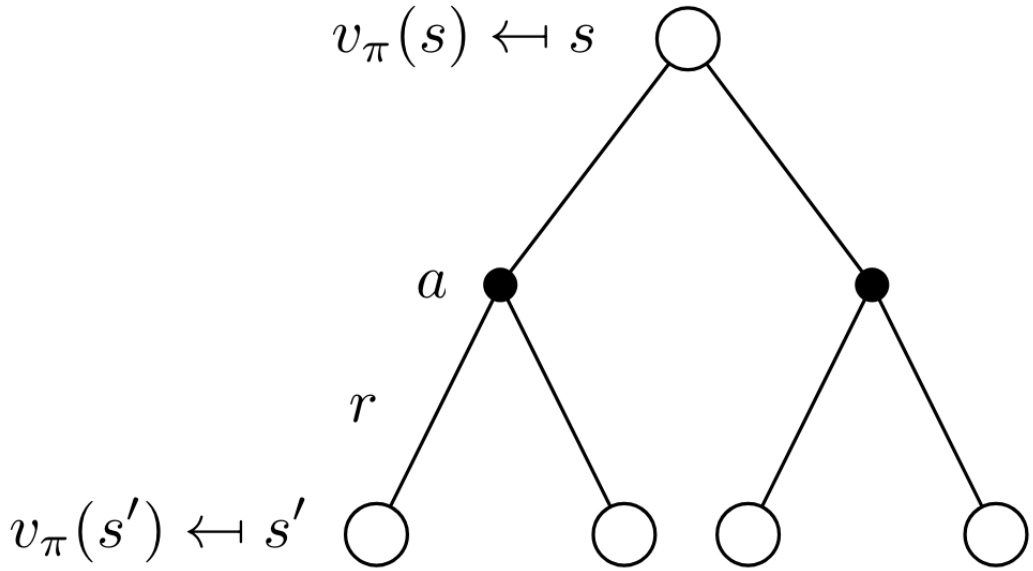
$$V_\pi(s) = \sum_{a \in A} \pi(a|s) Q_\pi(s,a)$$

$$Q_\pi(s,a) = R(s,a) + \gamma \sum_{s' \in S} P_{ss'}^a V_\pi(s')$$

$$V_\pi(s) = \sum_{a \in A} \pi(a|s)(R(s,a) + \gamma \sum_{s' \in S} P_{ss'}^a V_\pi(s'))$$

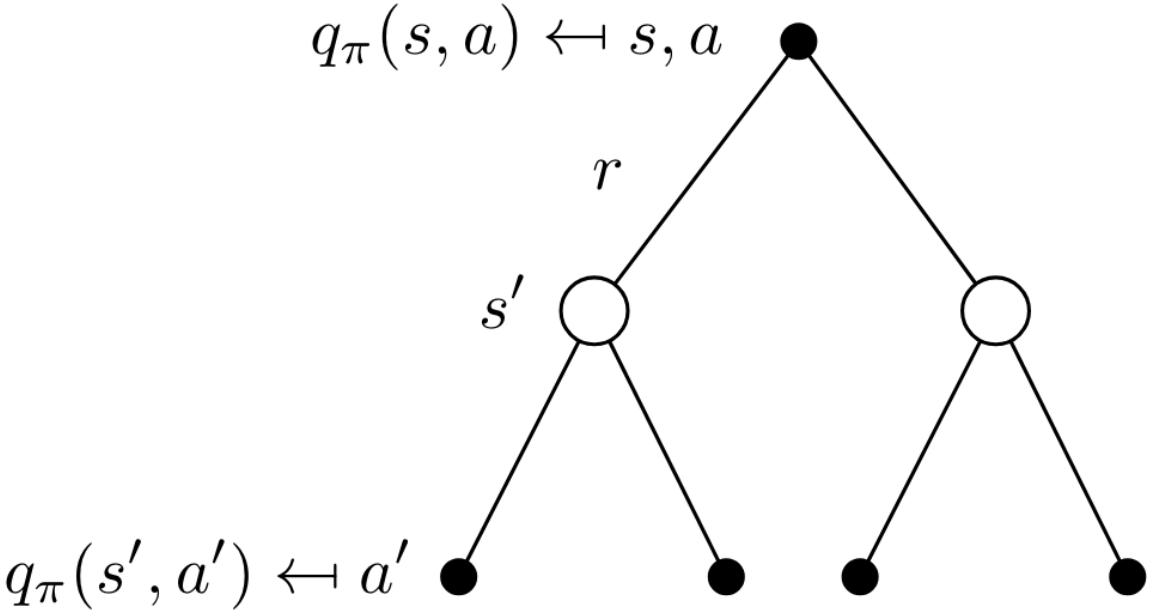
$$Q_\pi(s,a) = R(s,a) + \gamma \sum_{s' \in S} P_{ss'}^a \sum_{a' \in A} \pi(a'|s') Q_\pi(s',a')$$

贝尔曼期望函数



$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_\pi(s') \right)$$

贝尔曼期望函数

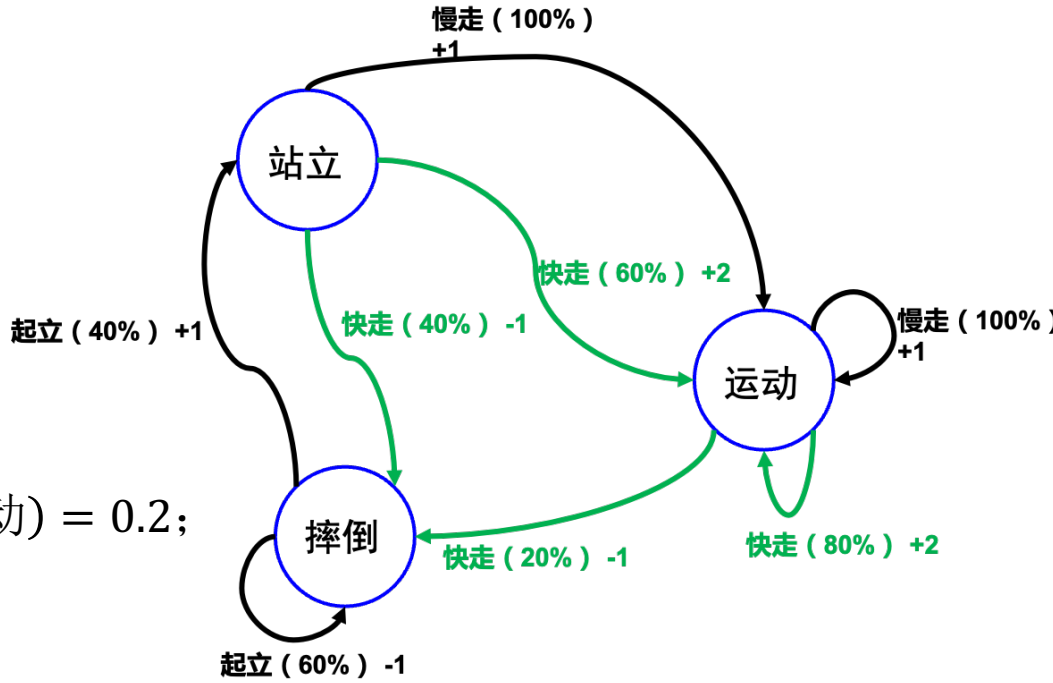


$$q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a' \in \mathcal{A}} \pi(a'|s') q_\pi(s',a')$$

贝尔曼期望方程

假设：

- $V(\text{站立}) = 1$
- $V(\text{摔倒}) = 0$
- $V(\text{运动}) = 1.4$
- $\gamma = 0.95$
- $\pi(\text{慢走}|\text{运动}) = 0.8$；$\pi(\text{快走}|\text{运动}) = 0.2$；



则：

- $Q_\pi(\text{运动，慢走}) = 100\% * (1 + 0.95 * 1.4) = 2.33$
- $Q_\pi(\text{运行，快走}) = 80\% * (2 + 0.95 * 1.4) + 20\% * (-1 + 0.95 * 0) = 2.464$

- $V(\text{运动}) = 0.8 * 2.33 + 0.2 * 2.464 = 2.3568$

贝尔曼最优方程

在策略迭代中，我们仅对最优值感兴趣，而不是针对某个策略下的期望值感兴趣，因此在计算中，我们仅选择价值函数的最大值，即：

$$V_*(s) = \max_{a \in A} Q_*(s,a)$$

$$Q_*(s,a) = R(s,a) + \gamma \sum_{s' \in S} P_{ss'}^a V_*(s')$$

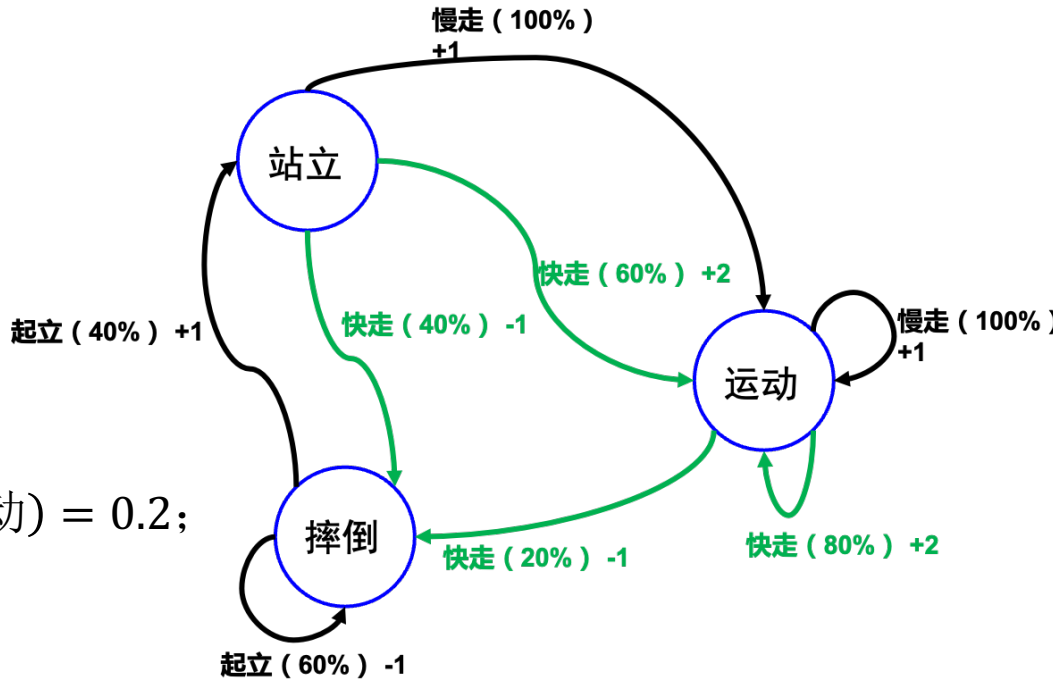
$$V_*(s) = \max_{a \in A} (R(s,a) + \gamma \sum_{s' \in S} P_{ss'}^a V_*(s')$$

$$Q_*(s,a) = R(s,a) + \gamma \sum_{s' \in S} P_{ss'}^a \max_{a' \in A} Q_*(s',a')$$

贝尔曼最优方程

假设：

- $V(\text{站立}) = 1$
- $V(\text{摔倒}) = 0$
- $V(\text{运动}) = 1.4$
- $\gamma = 0.95$
- $\pi(\text{慢走}|\text{运动}) = 0.8$；$\pi(\text{快走}|\text{运动}) = 0.2$；



则：

- $Q_\pi(\text{运动，慢走}) = 100\% * (1 + 0.95 * 1.4) = 2.33$
- $Q_\pi(\text{运行，快走}) = 80\% * (2 + 0.95 * 1.4) + 20\% * (-1 + 0.95 * 0) = 2.464$

- $V(\text{运动}) = \max(2.33, 2.464) = 2.464$

课程大纲

腾讯开悟

- 课程概述
- 动态规划
- MDP建模
- **策略评估与优化**

策略评估与优化

- 如果我们有环境的完整信息，那么问题就会转变成可以使用DP解决的动态规划问题。但不幸的是，大多数情况下，我们很难直接获取状态转移和奖励函数。

- 虽然我们无法直接使用贝尔曼方程来解决MDP问题，但是它是强化学习的理论基础，策略评估与策略改进也是算法迭代的基础。

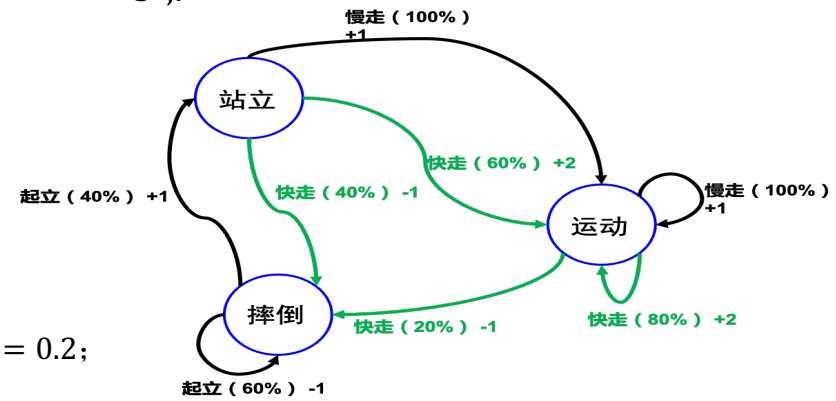
策略评估

- 对于给定的策略π，策略评估用于计算状态价值函数：

$$V_{t+1}(s) = \mathbb{E}_\pi[r + \gamma V_t(s')|S_t = s]$$
$$= \sum_a \pi(a|s) \sum_{s',r} P(s', r|s, a)(r + \gamma V_t(s'))$$

假设：

- $V(\text{站立}) = 1$
- $V(\text{摔倒}) = 0$
- $V(\text{运动}) = 1.4$
- $\gamma = 0.95$
- $\pi(\text{慢走}|\text{运动}) = 0.8$；$\pi(\text{快走}|\text{运动}) = 0.2$；



则：

- $Q_\pi(\text{运动，慢走}) = 100\% * (1 + 0.95 * 1.4) = 2.33$
- $Q_\pi(\text{运行，快走}) = 80\% * (2 + 0.95 * 1.4) + 20\% * (-1 + 0.95 * 0) = 2.464$

- $V(\text{运动}) = 0.8 * 2.33 + 0.2 * 2.464 = 2.3568$

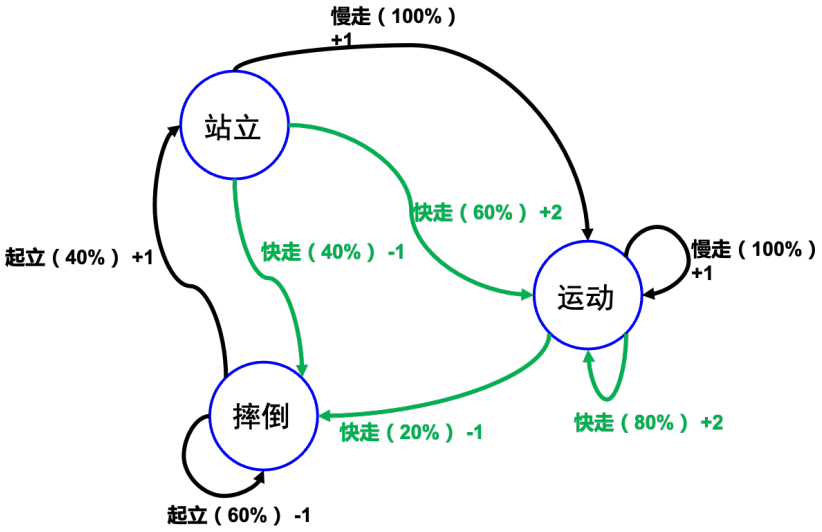
策略优化

- 基于价值函数，策略优化通过更贪婪的行为生成更好的策略 $\pi' \geq \pi$：

$$Q_\pi(s,a) = \mathbb{E}[R_{t+1} + \gamma V_\pi(S_{t+1}) | S_t = s, A_t = a]$$
$$= \sum_{s',r} P(s', r|s, a)(r + \gamma V_\pi(s'))$$
$$a = \underset{a \in A}{\operatorname{argmax}}\, Q_\pi(s,a)$$

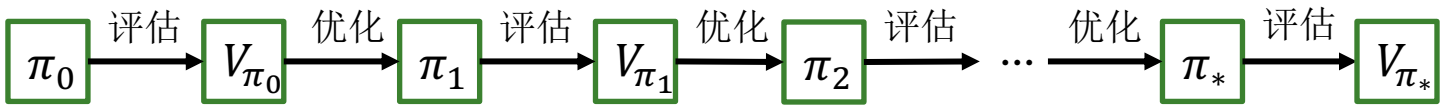
- Q_π(运动，慢走) $= 100\% * (1 + 0.95 * 1.4) = 2.33$
- Q_π(运行，快走) $= 80\% * (2 + 0.95 * 1.4) +$
  $20\% * (-1 + 0.95 * 0) = 2.464$

- V(运动) $= \max(2.33, 2.464) = 2.464$
- $a^* = argmax(2.33, 2.464) =$ 快走



策略迭代

- 一般策略迭代（Generalized Policy Iteration）算法指的是一个改进策略的迭代过程，它将策略评估和策略优化相结合：



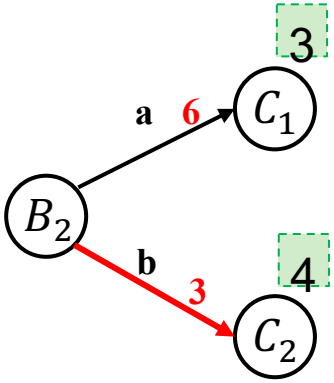
- 在GPI中，价值函数可以通过重复迭代来不断接近当前策略的实际值；策略也在此过程中不断接近最优策略；且总是可以收敛到最优策略。

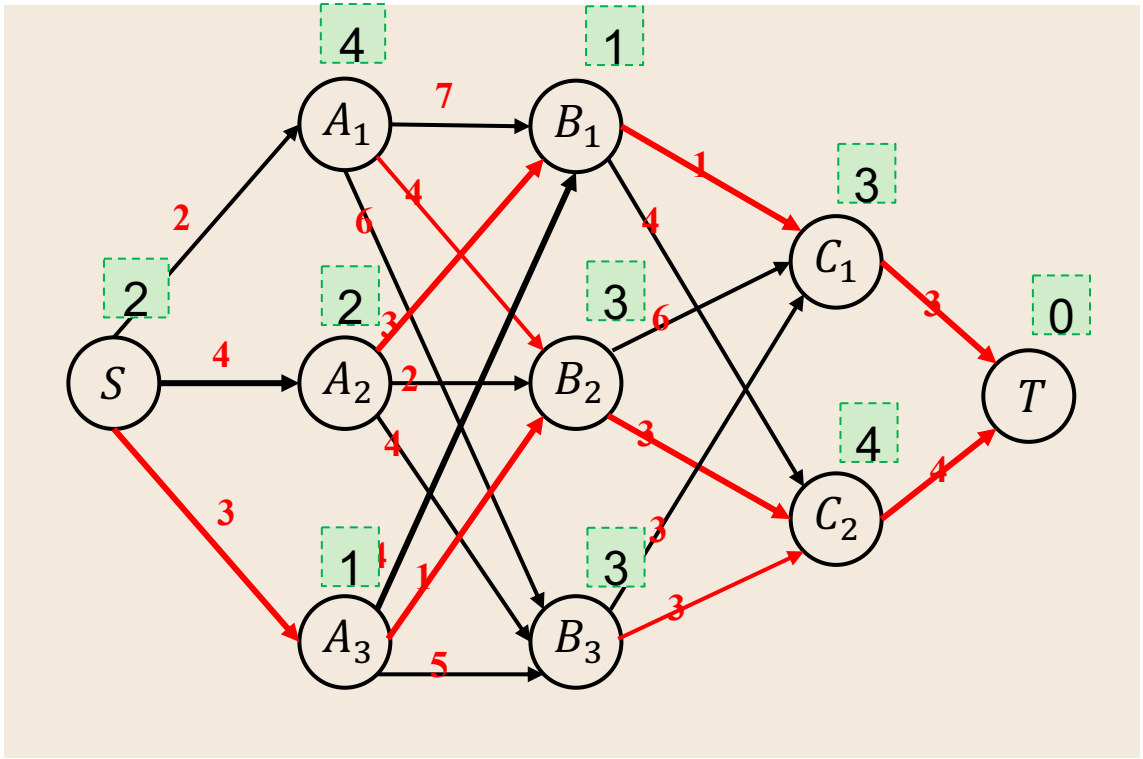
策略迭代

- 给定一个策略π，如果采用贪婪的方式优化策略，即$\pi^* = argmax_{a \in A} Q_\pi(s,a)$生成一个新的更优策略，这个改进的策略可以确保比原来的策略好，因为：

$$
\begin{aligned}
Q_\pi\big(s,\pi^*(s)\big) &= Q_\pi\left(s, \operatorname*{argmax}_{a \in A} Q_\pi(s,a)\right) \\
&= \max_{a \in A} Q_\pi(s,a) \\
&\geq Q_\pi\big(s,\pi(s)\big) \\
&= V_\pi(s)
\end{aligned}
$$

回看最短路径案例





- $V(C_1) = 3$
- $V(C_2) = 4$
- $\gamma = 1$

则：

- $Q(B_2, a) = 6 + 1 * 3 = 9$
- $Q(B_2, b) = 3 + 1 * 4 = 7$
- $V(B_2) = \min\big(Q(B_2, a), Q(B_2, b)\big) = \min(9, 7) = 7$
- 决策：$\arg\min\big(Q(B_2, a), Q(B_2, b)\big) = b$

下午实验课

需要完成：

- 配置强化学习环境；
- 安装gym；
- 完成案例CartPole环境的测试；
- 完成王者荣耀环境部署（一周内完成即可）；

提问环节

谢 谢