

RL: Планирование

Обучение vs планирование

❑ Обучение

- Среда – black box
- Исследование методом проб и ошибок
- Минимизация сожаления (regret)

❑ Планирование

- Имеем модель среды
- Ищем оптимальное поведение
- Действуем оптимально

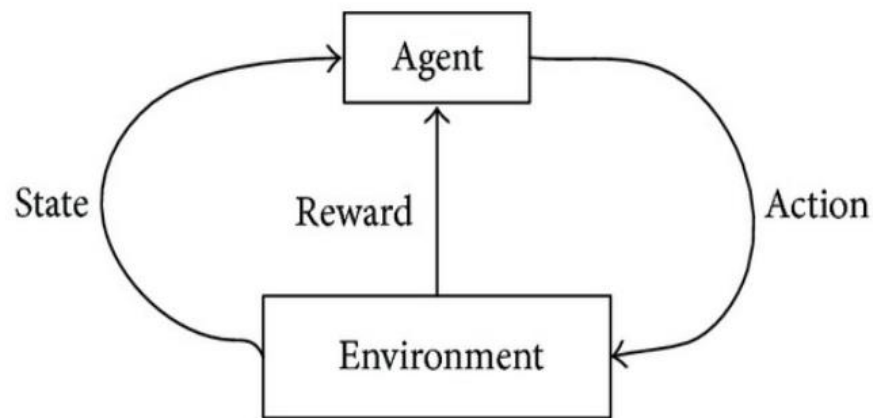
Настройка на основе модели

Что мы знаем

- Переходы между
состояниями

$$P(s_{next}|s, a) \text{ или } s_{next} = T(s, a)$$

- Награда $r(s_t, a_t)$



Настройка на основе модели

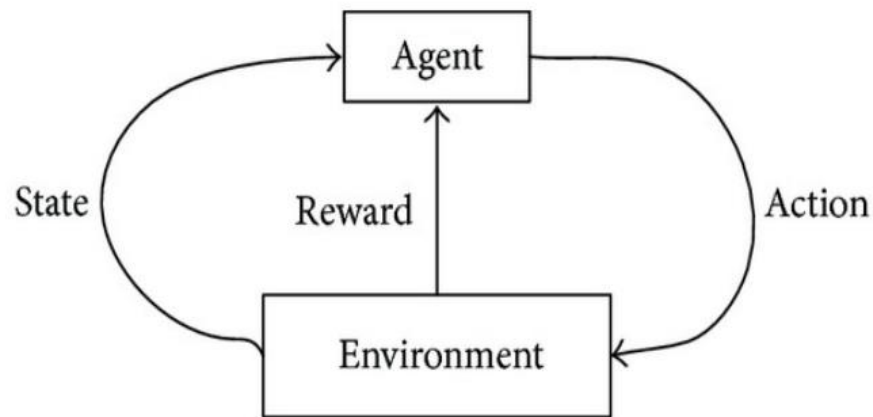
Что мы знаем

- Переходы между
состояниями

$$P(s_{next}|s, a) \text{ или } s_{next} = T(s, a)$$

Более слабая версия: мы можем
семплировать только из $P(s'|s, a)$

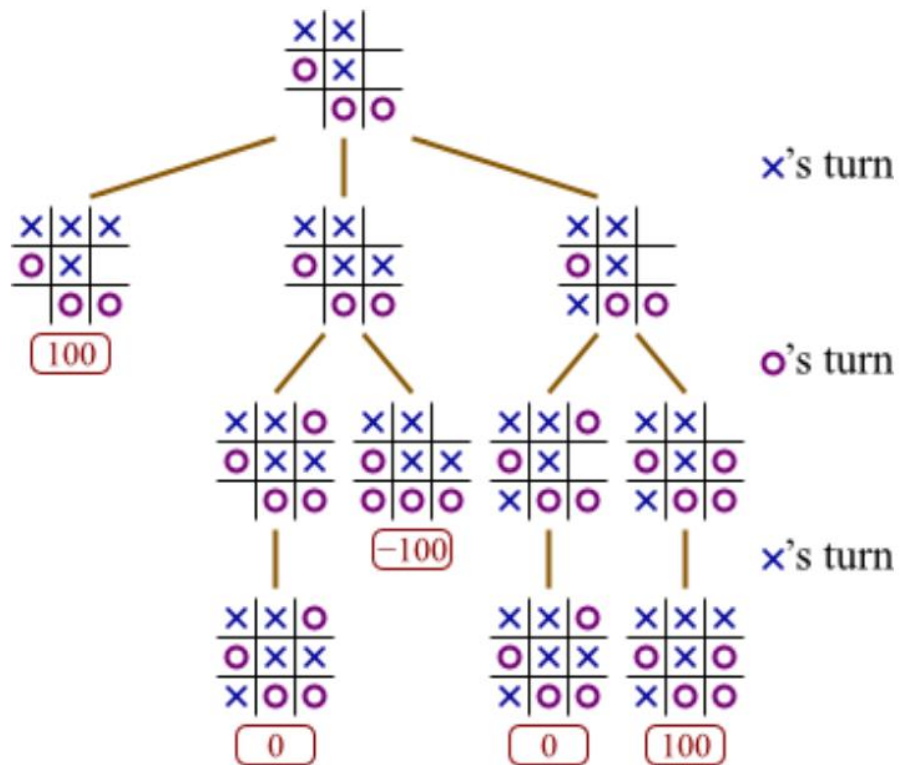
- Награда $r(s_t, a_t)$



Случай – состязательный агент

- Детерминированный случай, но есть второй агент...
- И он играет против нас!
- Мы хотим получить максимальное ожидаемое вознаграждение.
- Примеры:
 - Любая настольная игра: шахматы, шашки, го
 - Pong :)

Случай – состязательный агент



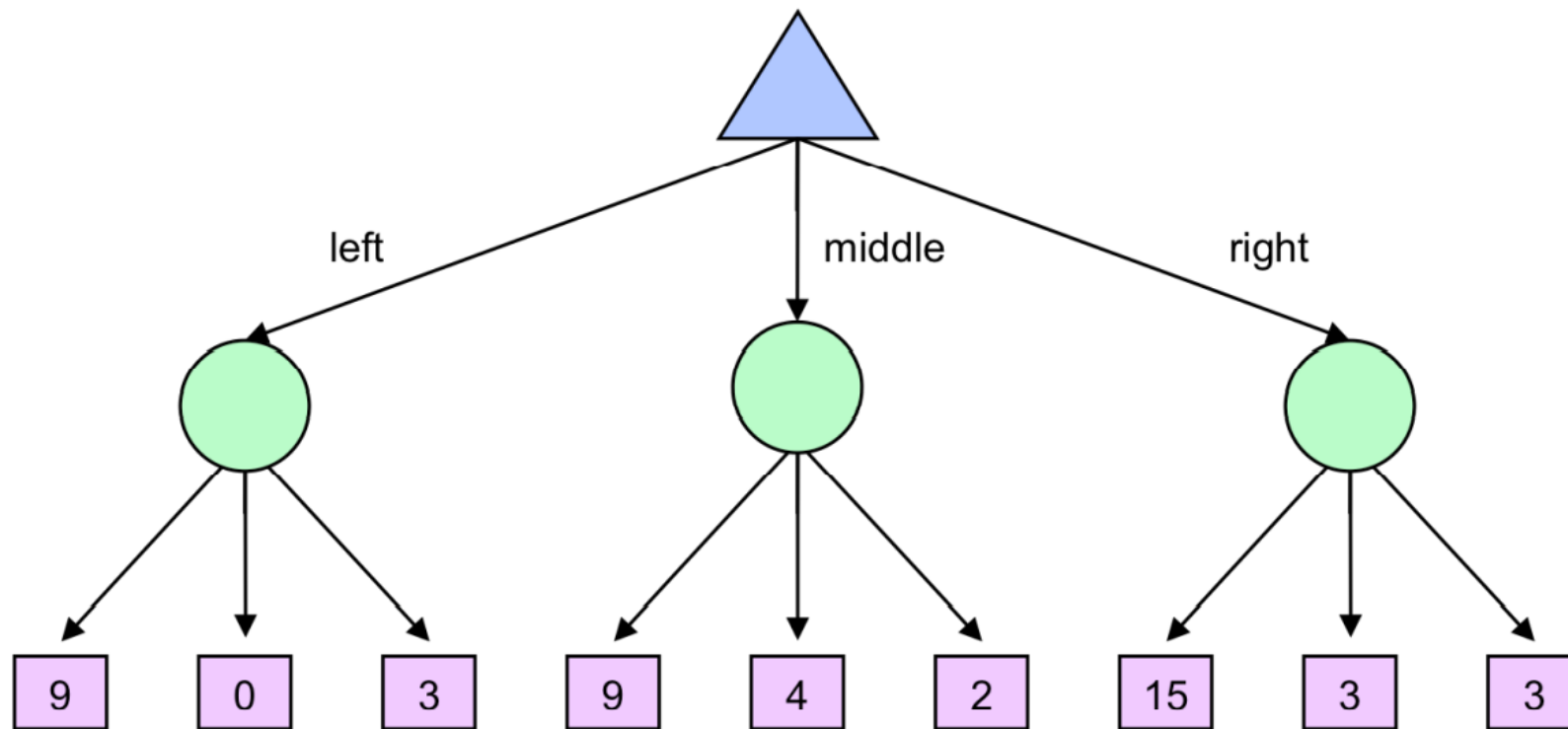
Общий случай: MDP

- Стохастичная среда

$$s \sim P(s_{next} | s, a)$$

- Мы хотим получить наибольшее ожидаемое вознаграждение или наименьшие ожидаемые затраты

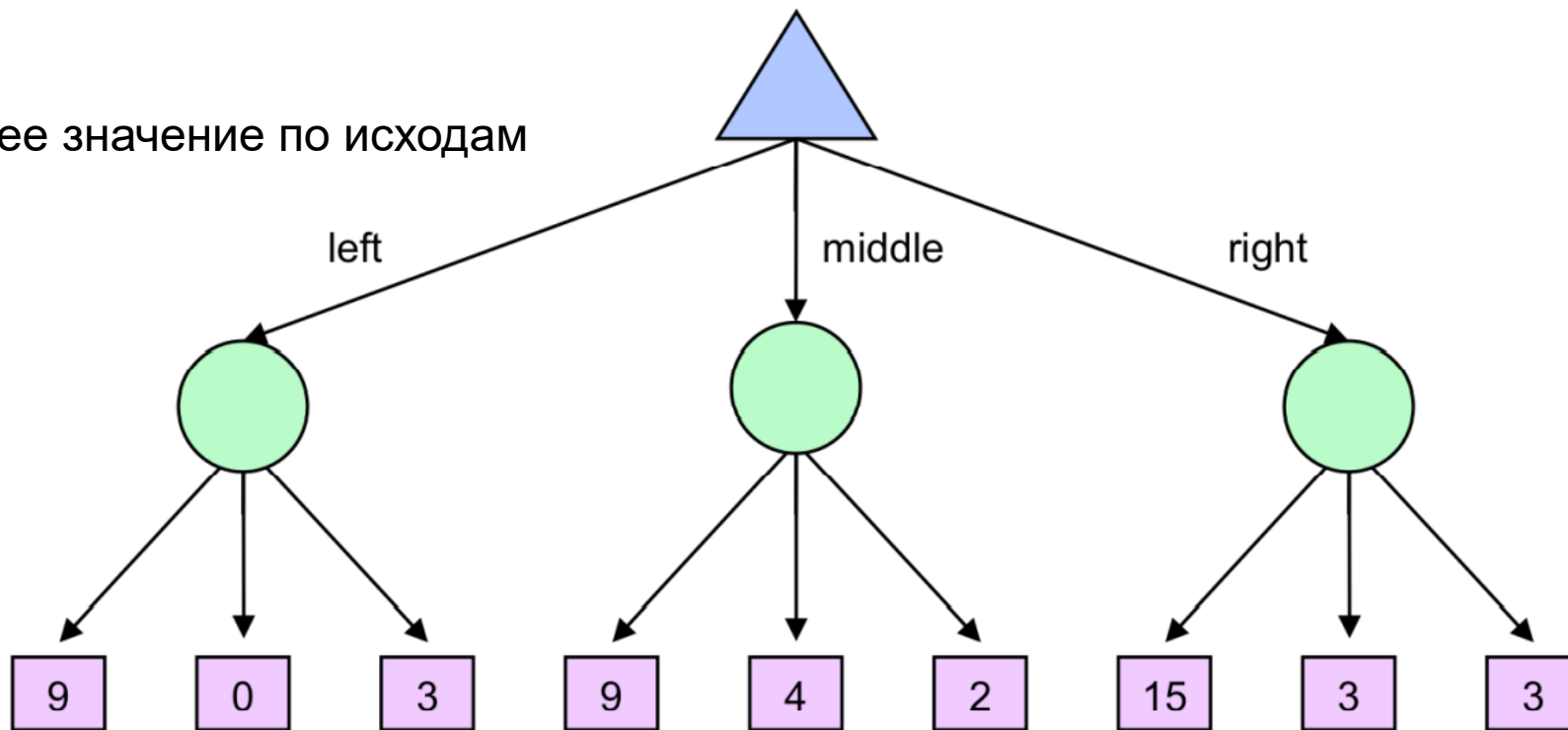
Общий случай: MDP



Как оценить ценность действия?

Общий случай: MDP

Среднее значение по исходам



Как оценить ценность действия?

Большое/непрерывное пространство состояний

- Мы не можем исследовать все узлы.
- Нужно выбрать наиболее интересные!
- Примеры:
 - ~ любой практический случай использования :)

Atari

Исследование на основе подсчета

□ UCB-1 for bandits

Идея


- Отдавайте приоритет действиям с неопределенными результатами!
- Меньше посещений = больше неопределенности.
- Математика: добавление верхней доверительной связи к вознаграждению.

Исследование на основе подсчета

UCB-1 для бандитов

Выбираем действие пропорционально \tilde{v}_a

Верхняя доверительная
граница для $r \in [0, 1]$

$$\tilde{v}_a = v_a + \sqrt{\frac{2 \log N}{n_a}}$$


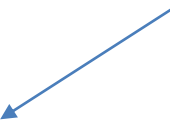
- N общее количество шагов
- n_a количество случаев выбора действия a

Исследование на основе подсчета

Обобщенный UCS для множества состояний

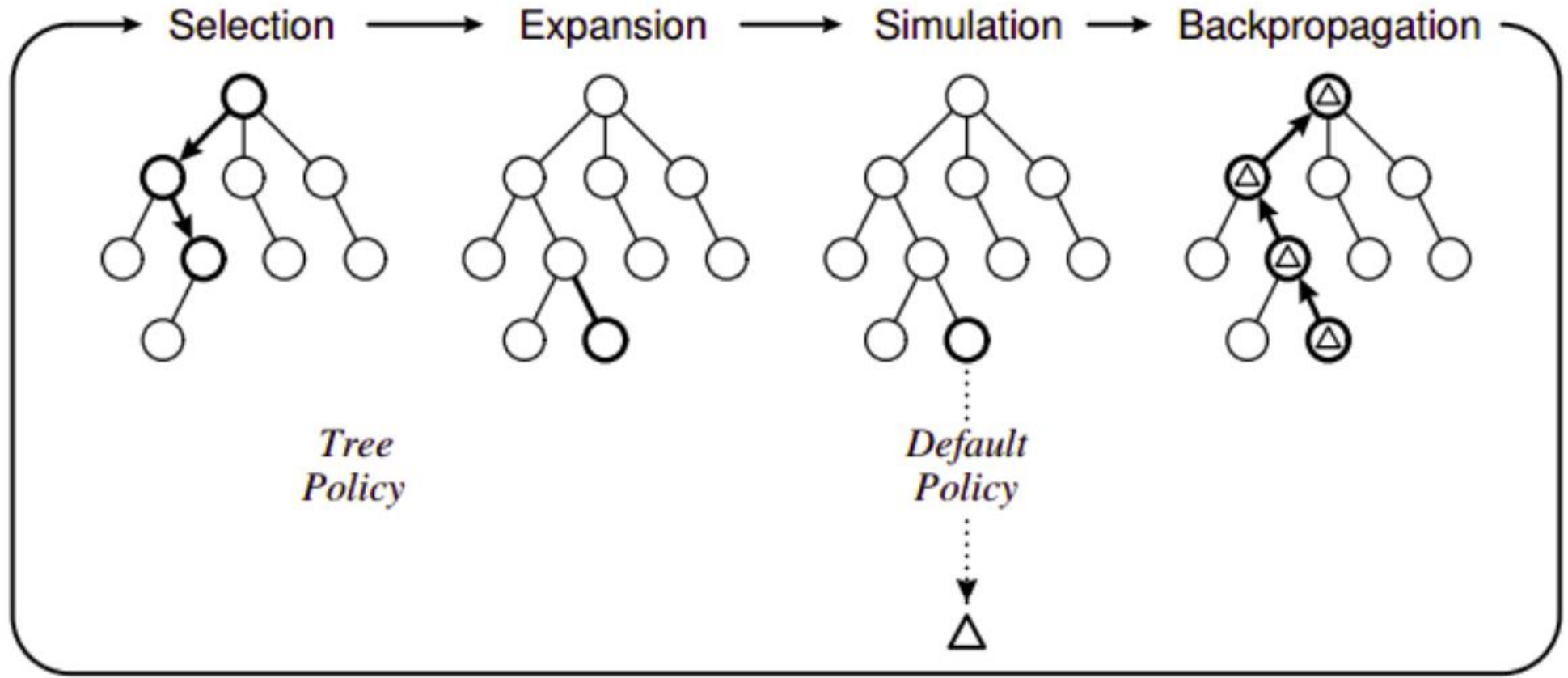
Выбираем действие пропорционально \tilde{v}_a

Верхняя доверительная граница для $r \in [0, 1]$

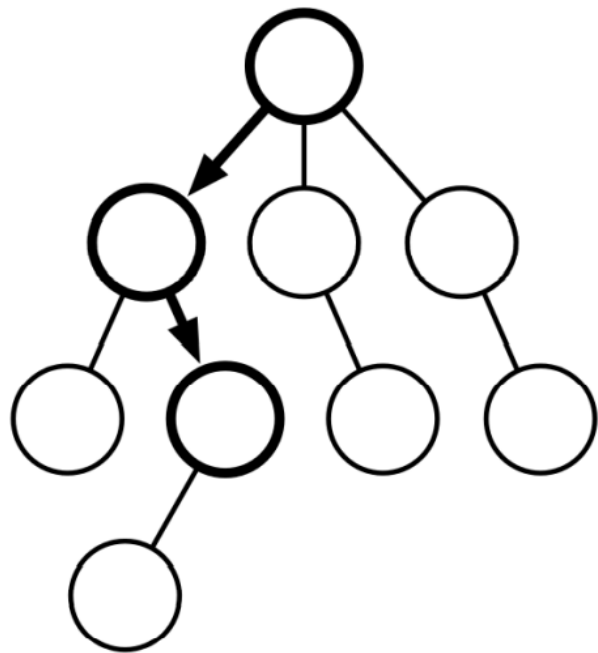

$$\tilde{Q}(s, a) = Q(s, a) + \sqrt{\frac{2 \log N_s}{n_{s,a}}}$$

- N_s общее количество посещений состояния s
- $n_{s,a}$ количество случаев выбора действия a из состояния s

MCTS



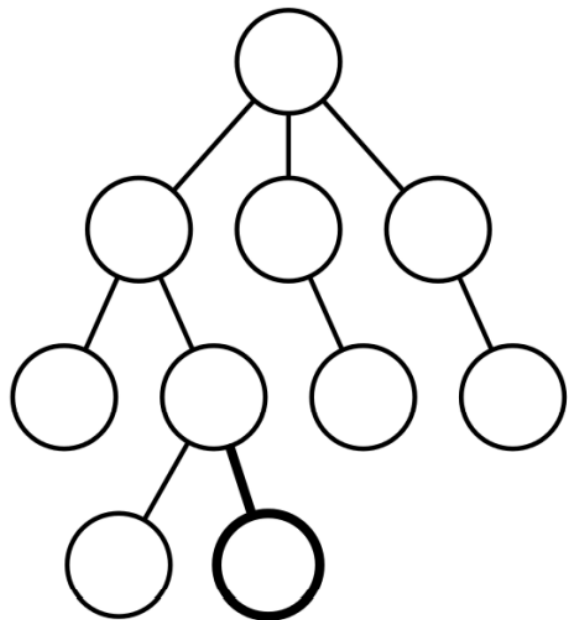
MCTS: selection



Начиная с корня,
рекурсивно выберите узел с
наивысшей оценкой ucb-1

$$\tilde{Q}(s, a) = Q(s, a) + \alpha \sqrt{\frac{2 \log N_s}{n_{s,a}}}$$

MCTS: expansion

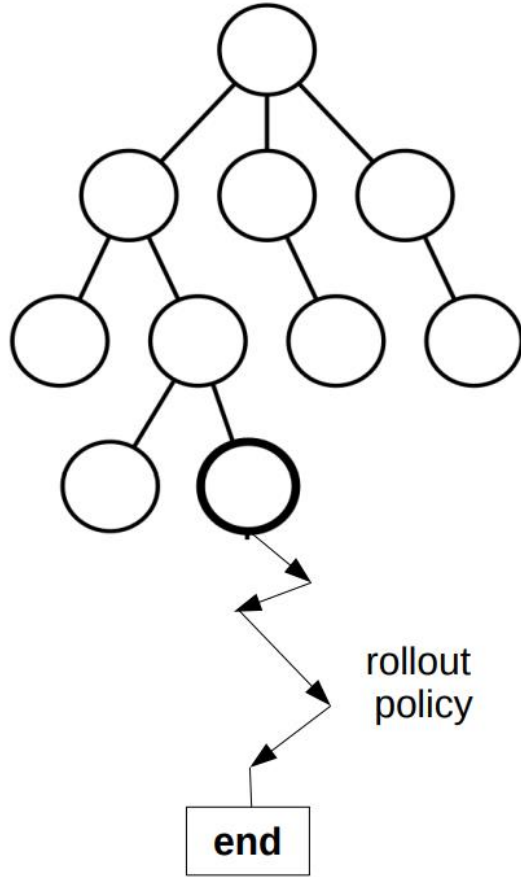


Добавление одного или нескольких детей от выбранного узла.

Каждый ребенок является результатом одношаговой симуляции $s \rightarrow s', a, r$

Простой случай: добавление одного узла на одно действие.

MCTS: Rollout (sampling)

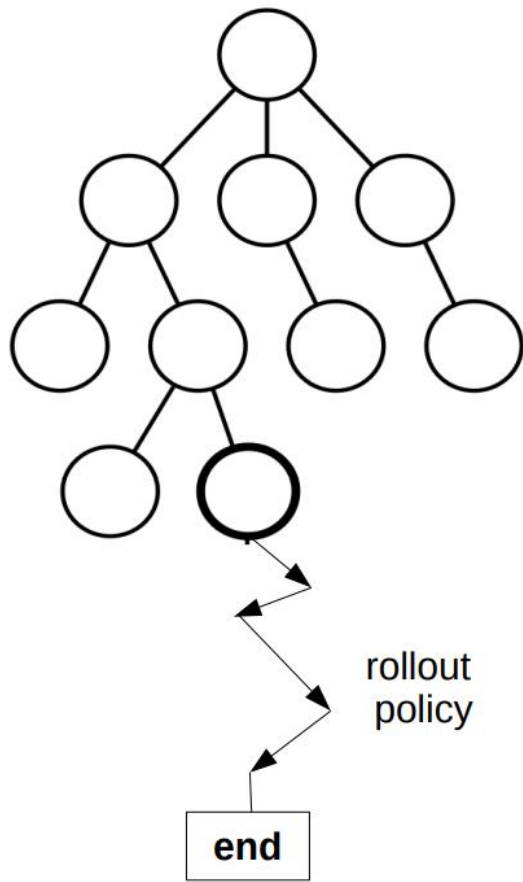


Оценить значение узла играя в игру от этого состояния до конца с простой политикой.

Например, применяя случайные действия (политику)

Запомним общую награду.

MCTS: Rollout (sampling)



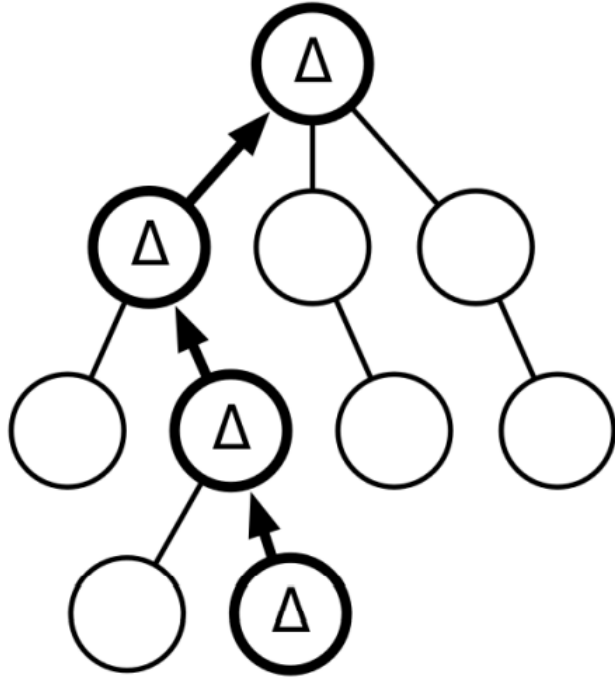
Оценить значение узла играя в игру от этого состояния до конца с простой политикой.

Например, применяя случайные действия (политику)

Запомним общую награду.

Можем ли мы добиться большего, чем случайность?

MCTS: Backprop

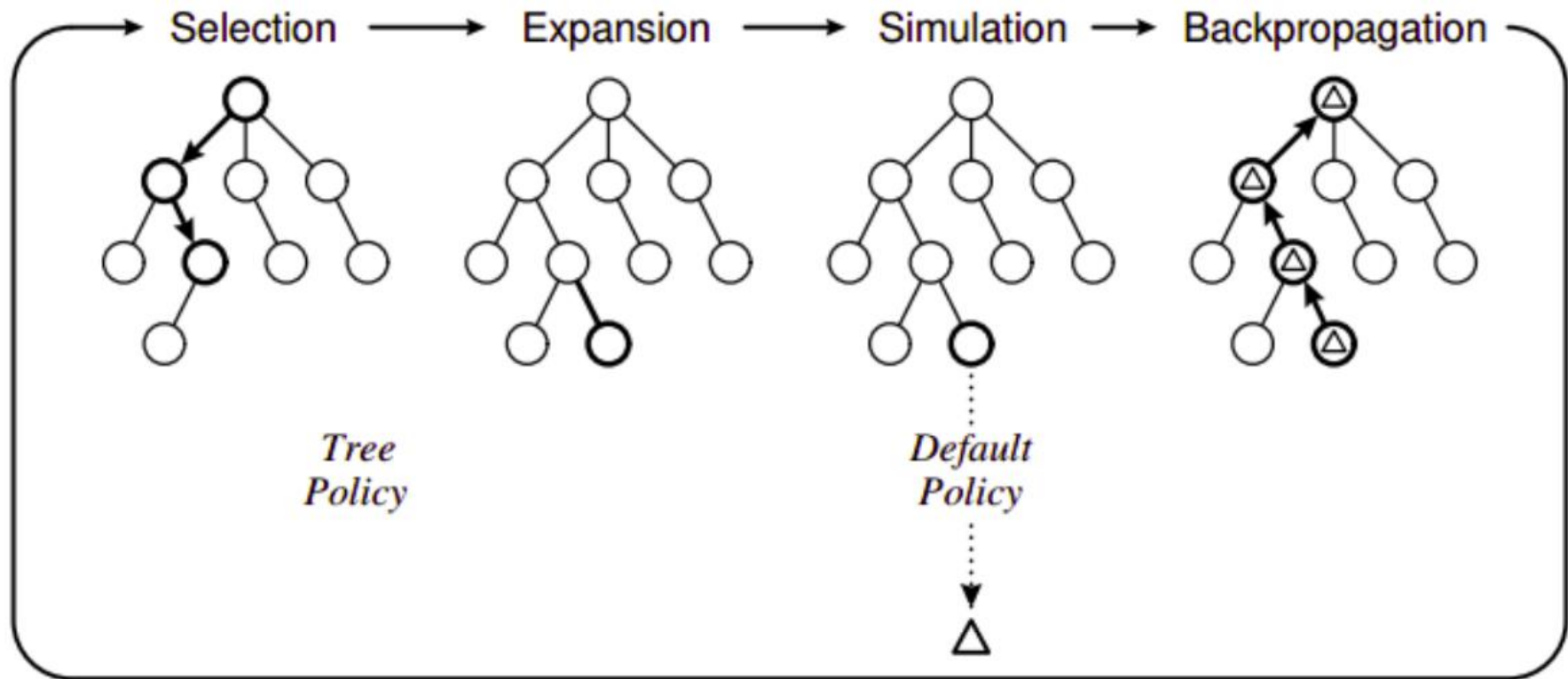


Учитывая rollout reward, обновить значение листа и всех его родителей.

$$V(\text{parent}) = r + \gamma V(\text{child})$$

Также увеличиваем счетчик посещений (N и n_a для ucb-1)

MCTS



Как выбрать действие из корня?

Model-based vs model-free

Model - free (DQN + hacks)

Agent	<i>B.Rider</i>	<i>Breakout</i>	<i>Enduro</i>	<i>Pong</i>	<i>Q*bert</i>	<i>Seaquest</i>	<i>S.Invaders</i>
DQN	4092	168	470	20	1952	1705	581
<i>-best</i>	5184	225	661	21	4500	1740	1075

Model-based (MCTS + UCT)

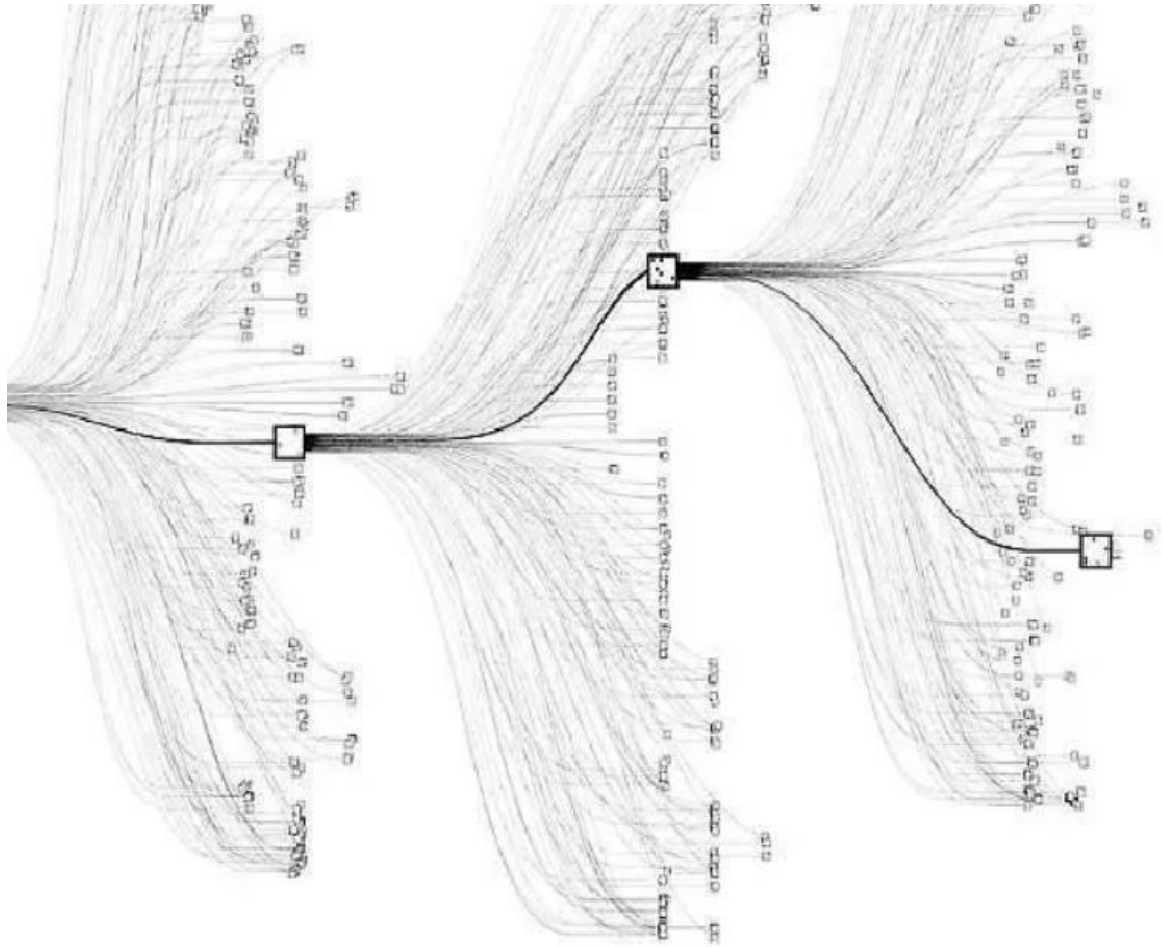
Agent	<i>B.Rider</i>	<i>Breakout</i>	<i>Enduro</i>	<i>Pong</i>	<i>Q*bert</i>	<i>Seaquest</i>	<i>S.Invaders</i>
UCT	7233	406	788	21	18850	3257	2354

ИСТОЧНИК: tinyurl.com/atari-mcts-guo

Дерево MCTS

img source:

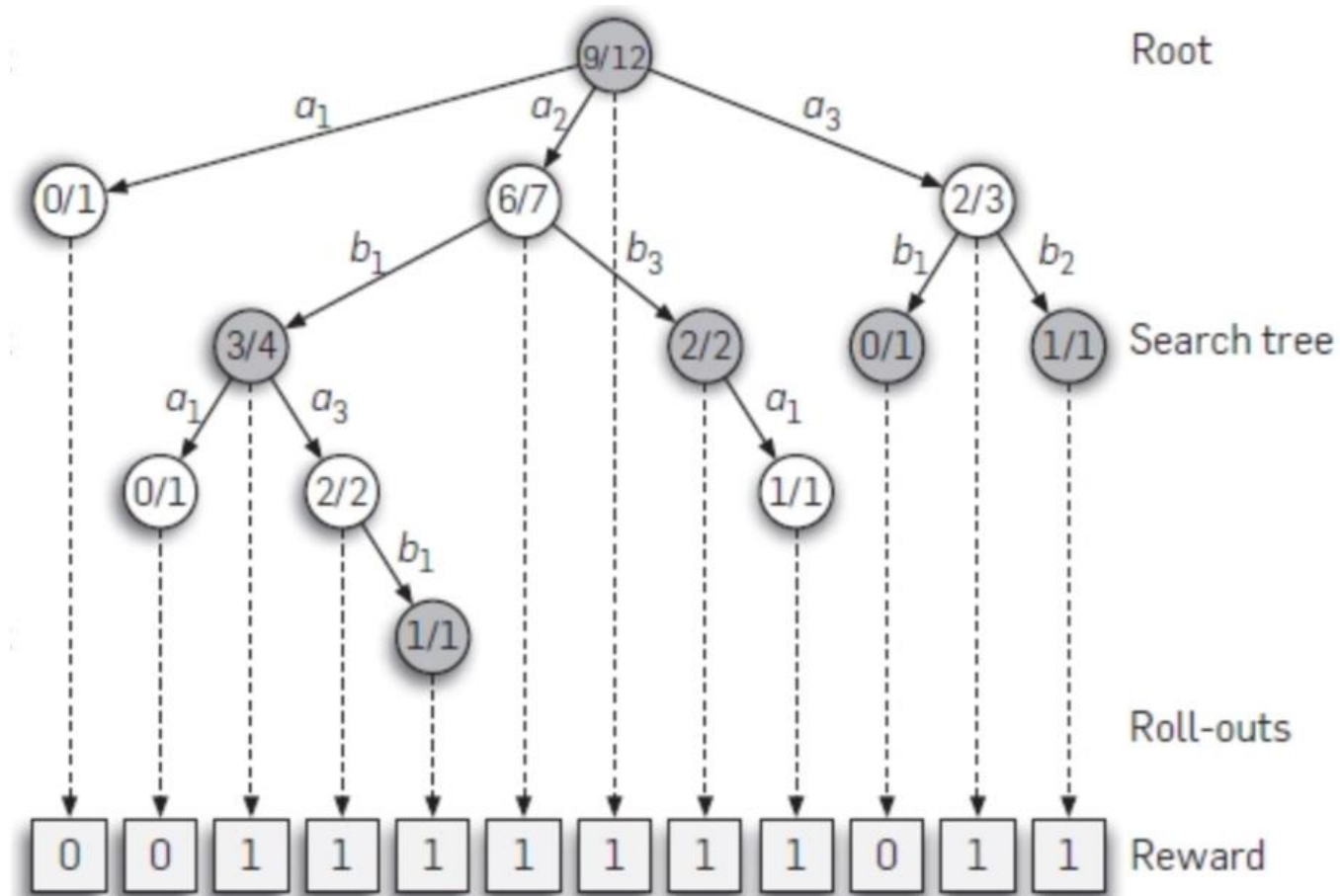
nikcheerla.github.io



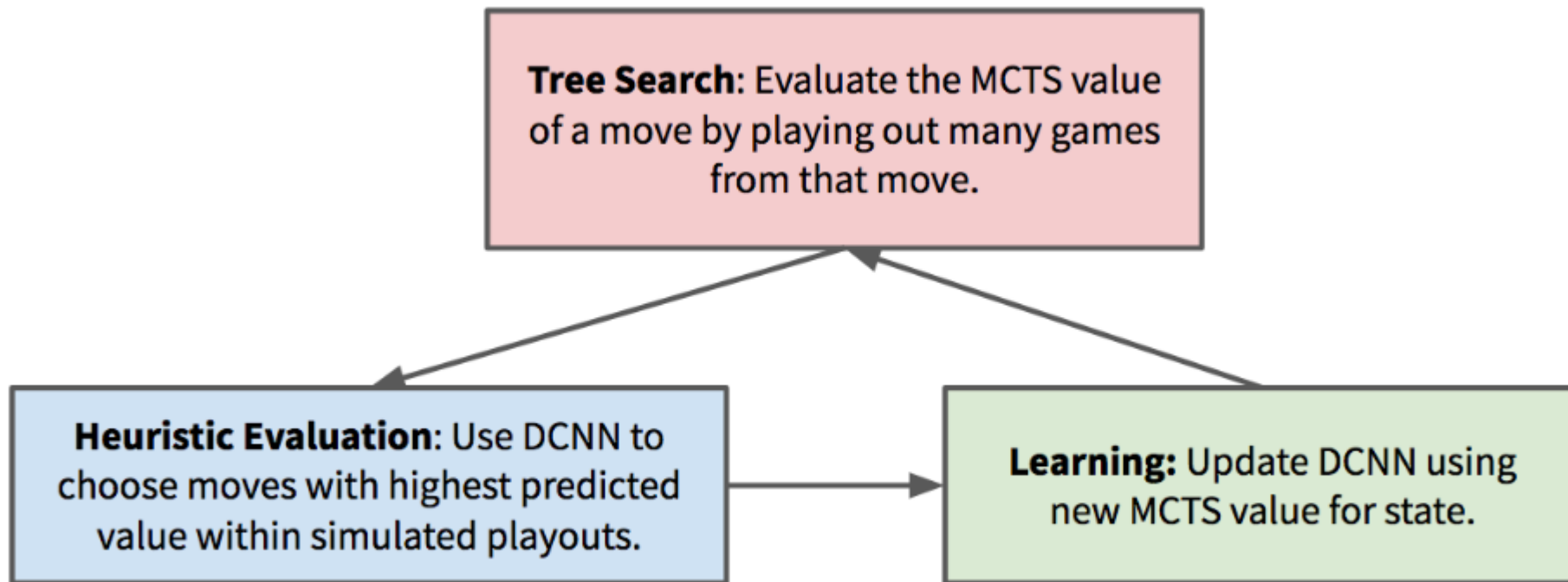
Дерево MCTS

img source:

[nikcheerla.github.io](https://github.com/nikcheerla)



Интеграция обучения и планирования



img source: [nikcheerla.github.io](https://github.com/nikcheerla)

AlphaGo, AlphaZero

Использование нейронной сети
для улучшения поиска

Использование MCTS для обучения
нейронной сети

Простая ConvNet; слегка настроенный MCTS; множество хаков

