Севастопольский государственный университет
Институт информационных технологий

Методы и системы искусственного интеллекта

Бондарев Владимир Николаевич

Севастопольский государственный университет
Институт информационных технологий и управления в технических системах

Лекция
Марковские процессы принятия решений.
Итерации по политикам (стратегиям)

Бондарев Владимир Николаевич

Марковский процесс принятия решений

Марковский процесс принятия решений (MDP) определяется множеством элементов:

1**. Множество состояний *s*** $\in$ ***S***. Состояния в MDP представляются также, как и состояния в традиционном поиске решений задач;

2. **Множество действий *a*** $\in$ ***A***. Действия в MDP также представляются аналогично действиям в традиционном поиске решений задач;

3. Переходная функция *T*(*s,a,s'*) (transition function), представляется вероятностями перехода *a* из состояния *s* в состояние *s'* посредством выполнения действия *a,* т.е. $P(s' \mid s, a)$;

4. Функция вознаграждения *R*(*s,a,s'*) (reward function) - определяет *награду*, получаемая агентом при переходе из состояния *s* в состояние *s'* посредством выполнения действия *a;*

5. Коэффициент дисконтирования *γ* (discount factor);

$$MDP=\{ S,A,T,R,\gamma\}$$

6. **Начальное состояние**;

7. Возможно **конечное состояние**.

Как правило, MDP использует небольшие награды на каждом шаге, наряду с большими вознаграждениями за достижение конечного состояния. Награда может быть положительной или отрицательной в зависимости от того, приносят ли действие пользу агенту.

Цель агента состоит в том, чтобы **получить максимально возможное вознаграждение** до того, как он достигнет некоторого конечного состояния.

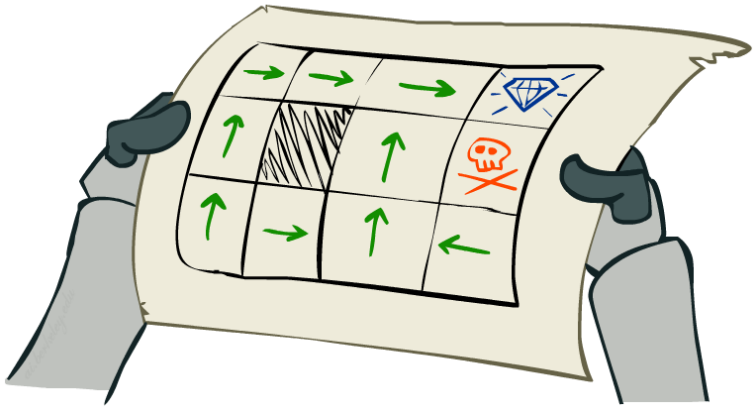
Политики (стратегии)

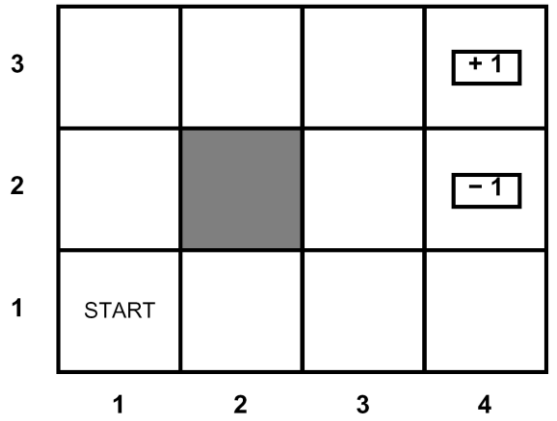
- Решение задачи в случае детерминированного поиска сводится к построению **оптимального плана** - последовательности действий от старта до цели.

- Решение задачи, представляемой в виде MDP, сводится к поиску **оптимальной политики:**

$$\pi^*: S \rightarrow A$$

  - **Политика** – это функция **π**, определяющая для каждого состояния *s* действие *a* ;
  - Оптимальная политика максимизирует ожидаемое накопленное вознаграждение, если ей следовать ;
  - Явная политика определяет поведение рефлексного агента: в заданном состоянии s агент, следующий политике **π,** выберет действие $a = \pi(s)$ без рассмотрения будущей последовательности своих действий.

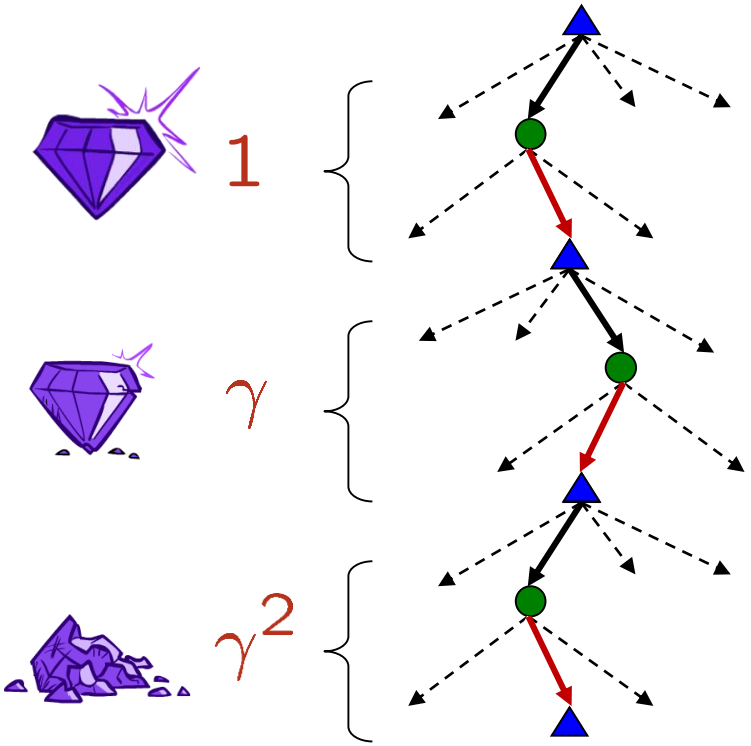




Оптимальная политика (обозначена стрелками) при R(s, a, s') = -0.03 для всех нетерминальных состояний s

Дисконтирование

- Как дисконтировать?
  - Каждый раз, когда мы спускаемся на 1 уровень , мы умножаем награду на ***γ*** .
- Зачем дисконтировать?
  - помогает алгоритмам сходиться
- Пример: Кумулятивная, накопленная награда (возврат)
- ***γ*** = 0.5
  - U([1,2,3]) = 1*1 + 0.5*2 + 0.25*3
  - U([1,2,3]) < U([3,2,1])



Накопленное вознаграждение

$$U([r_0, r_1, r_2, \ldots]) = r_0 + r_1 + r_2 + \cdots$$

Дисконтированное накопленное вознаграждение

$$U([r_0, r_1, r_2, \ldots]) = r_0 + \gamma r_1 + \gamma^2 r_2 \cdots$$

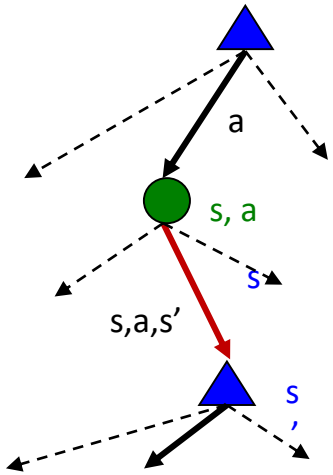
Коэффициент *γ* определяет относительную важность будущих и немедленных наград. Его значение лежит в диапазоне от 0 до 1; ***γ=*** 0 означает, что немедленные награды более важны, а ***γ=***1 означает, что будущие награды важнее немедленных.

Оптимальные функции ценности состояния, состояния-действия, политики

- Оптимальная ценность состояния s, $V^*(s)$ - ожидаемая кумулятивная награда, которую получит агент в s, если начнет из состояния s и будет следовать оптимальной политике π* (будет действовать оптимально)

- Оптимальная ценность q-состояния, $Q^*(s,a)$ - ожидаемая кумулятивная награда, которую получит агент при старте из состояния *s*, выбирая действия *a* и следуя оптимальной политике π*

- Функция оптимальной политики $\pi^*(s)$ - определяет оптимальное действие из состояния s, обеспечивающие максимум кумулятивной награды в s



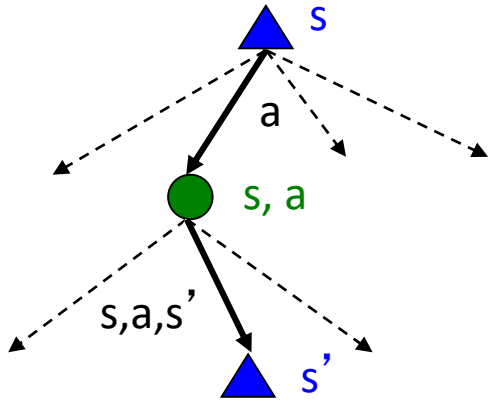
s состояние

(s, a) *q-состояние*

(s,a,s') переход

Уравнения Беллмана

- Рекурсивное определение ценности состояния



Максимальная (оптимальная) ценность состояния s:

$$V^*(s) = \max_a Q^*(s,a)$$

Оптимальная (ожидаемая) ценность q-состояния (s,a):

$$Q^*(s,a) = \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma\, V^*(s') \right]$$

Ценность, получаемая при переходе из (*s ,a)* в состояние *s'* и дальнейшем оптимальном поведении

Вероятность перехода из (*s,a*) в *s'*

Рекурсивное определение **оптимальной** ценности состояния s:

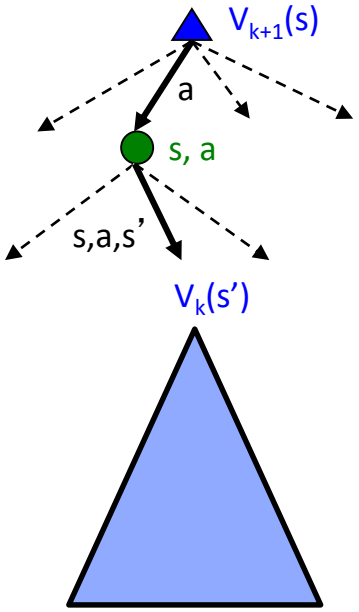
$$V^*(s) = \max_a \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma V^*(s')\right]$$

Итерация по значениям ценности состояний

Итерация по значениям V(s) - это алгоритм динамического программирования, который обеспечивает итеративное вычисления ценности состояний V(s), пока не начнет выполняться условие сходимости, т.е.

$$\forall s, V_{k+1}(s) = V_k(s))$$

Он работает следующим образом:

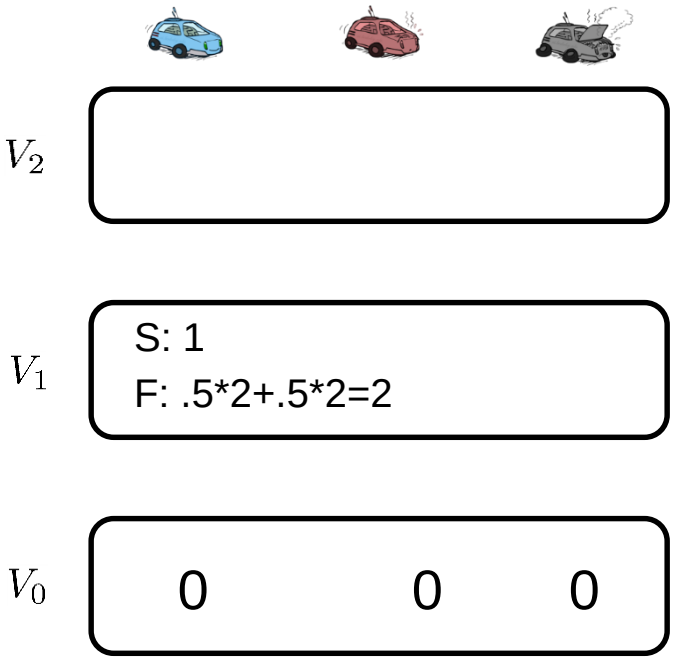


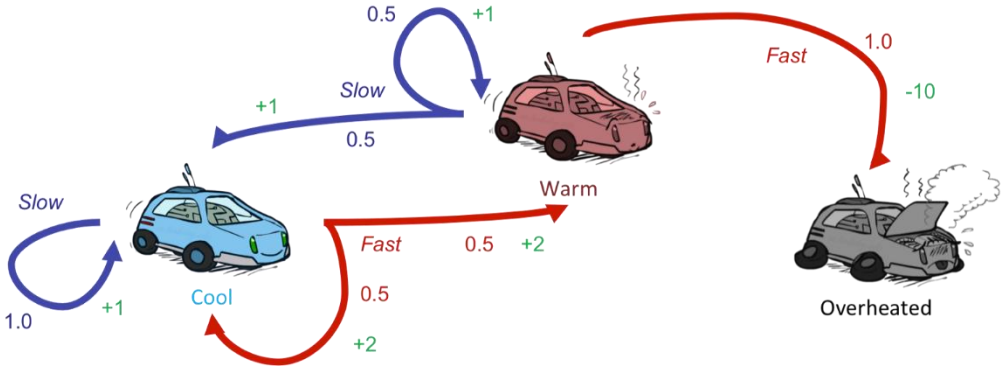
- Начало $V_0(s) = 0$: отсутствие действий на шаге 0 означает, что ожидаемая сумма наград равна нулю.
- Для каждого состояния выполнить один шаг Expectimax

$$\forall s \in S,\ V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V_k(s')]$$

- Повторять пока не сойдется
- Сложность каждой итерации: $O(S^2A)$
- Теорема: Решение сходится к уникальному оптимальному значению
  - приближения уточняются в направлении оптимальных значений ;
  – политика может сходиться задолго до того, как сойдутся ценности

Пример: итерация по значениям V(s)

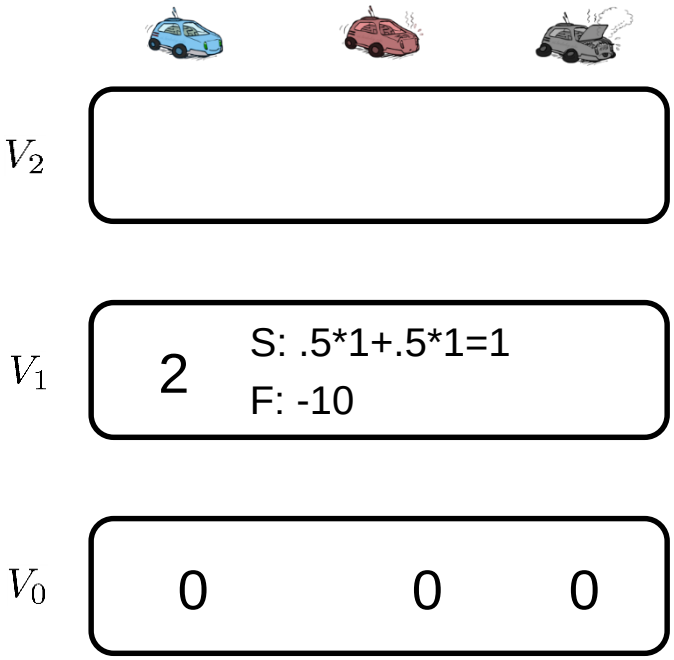


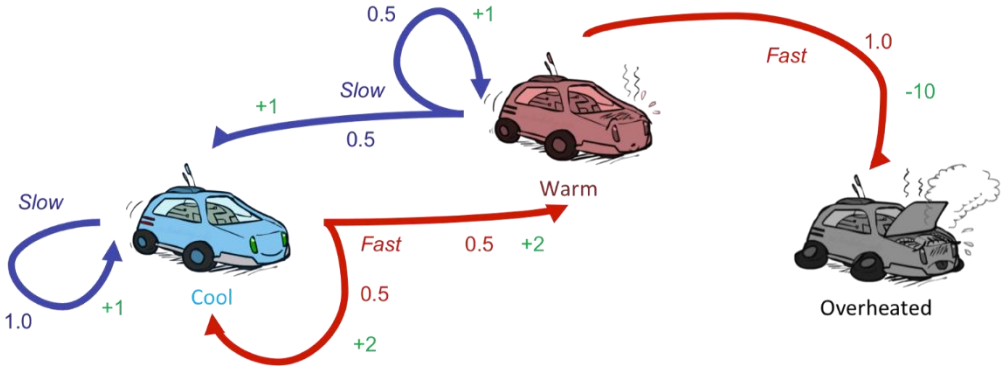


Предположим, что нет дисконтирования!

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

Пример: итерация по значениям V(s)

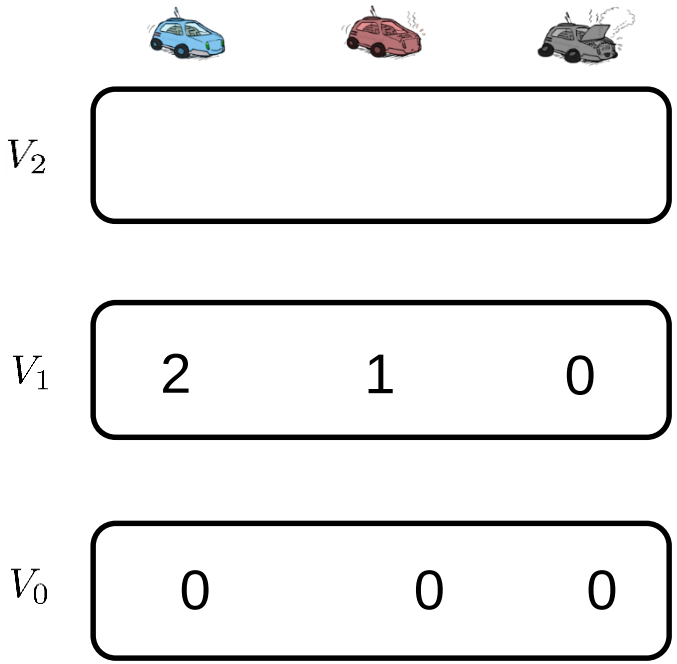


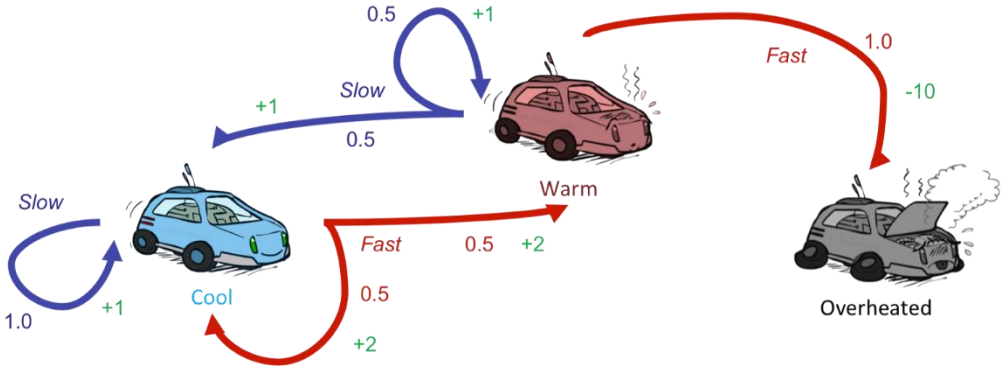


Предположим, что нет дисконтирования!

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

Пример: итерация по значениям V(s)





Предположим, что нет дисконтирования!

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

Пример: итерация по значениям V(s)

V_2

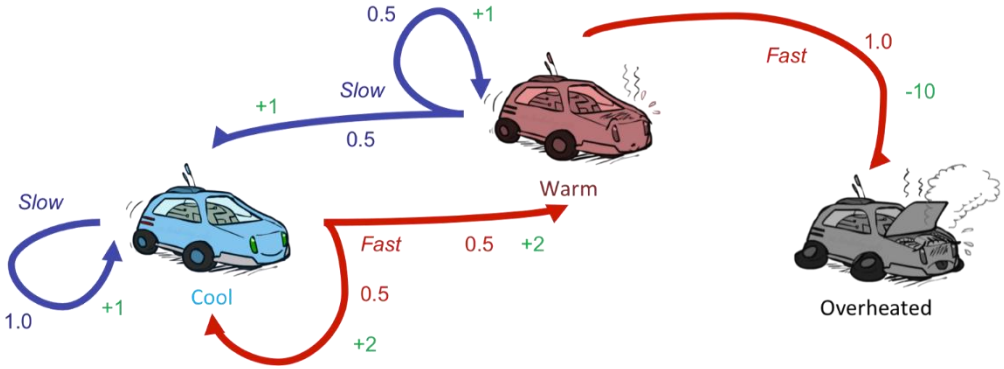
S: 1+2=3
F:
.5*(2+2)+.5*(2+1)=3.5

V_1

2	1	0

V_0

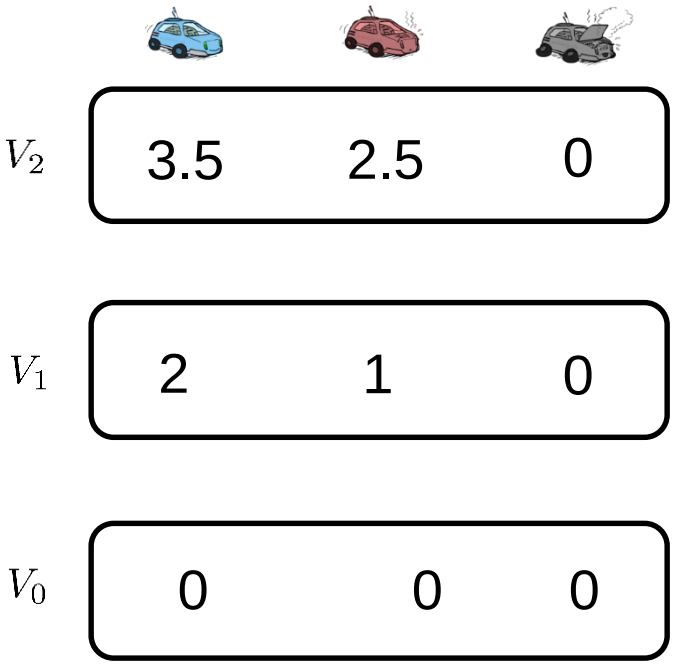
0	0	0

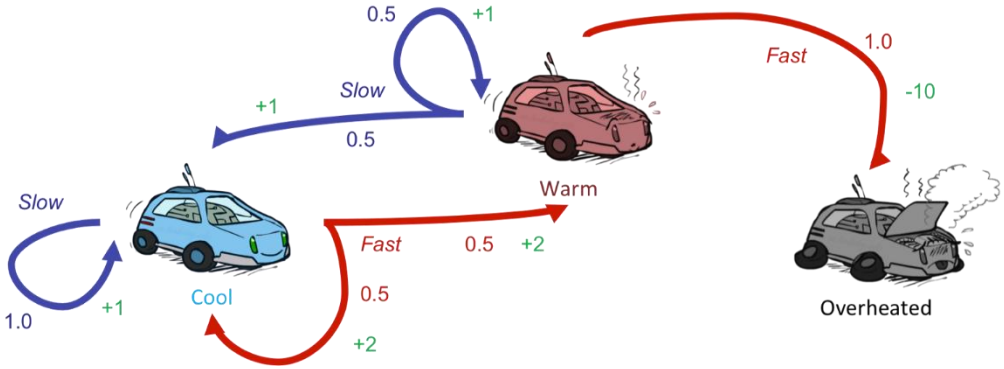


Предположим, что нет дисконтирования!

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

Пример: итерация по значениям V(s)





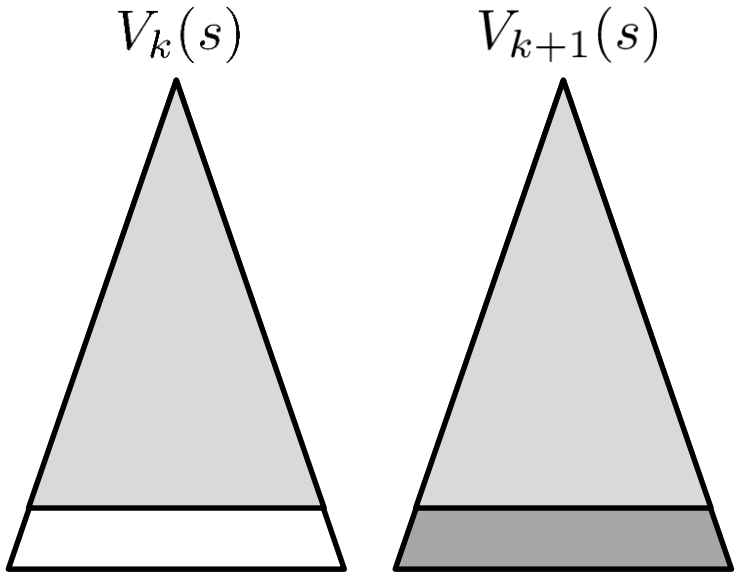
Предположим, что нет дисконтирования!

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

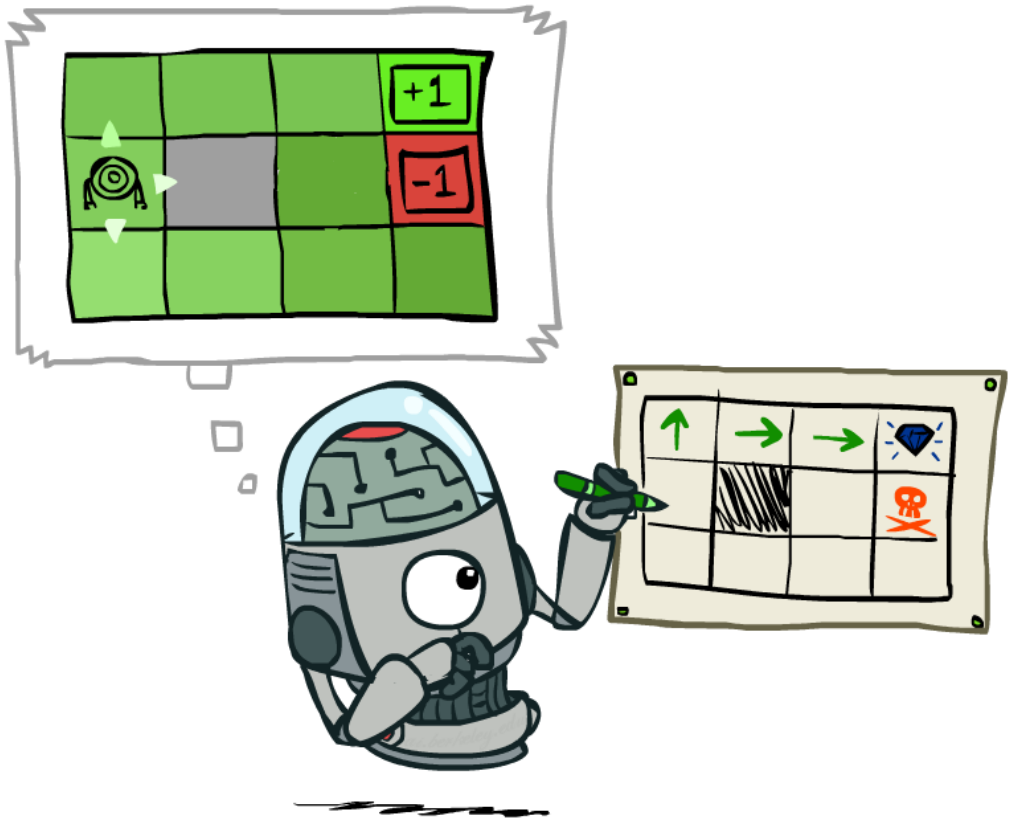
Сходимость*

- Откуда мы знаем, что значения V_k сходятся?
- **Случай 1**: Если дерево имеет максимальную глубину M, то V_M хранит фактические неусеченные значения.
- **Случай 2**: Если коэф. дисконт. меньше 1
  - То для любого состояния V_k и V_{k+1} можно рассматривать как результаты expectimax на глубине k и k + 1 в почти идентичных деревьях поиска.
  - Разница в том, что на нижнем уровне V_{k+1} имеет реальные награды, а V_k - нули.
  - Награды для последнего слоя в лучшем случае R_{MAX}
  - В худшем случае R_{MIN}
  - Награды дисконтируются пропорционально γ^k
  - Таким образом, V_k и V_{k+1} различаются не более чем

    на γ^k max | R |
  - Таким образом, когда k увеличивается, значения сходятся

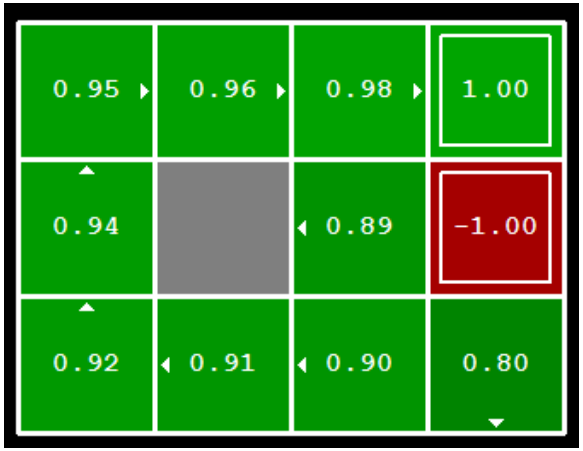


Извлечение политики (policy extraction)



Определение действий через ценность состояний (извлечение политики)

- Цель решения MDP – определение оптимальной политики. Предположим, что вычислены оптимальные значения ценности состояний V*(s)



- Каким образом нам следует действовать в каждом состоянии?
  - Это не очевидно!

- Чтобы определить политику, необходимо выполнить шаг Expectimax

$$\pi^*(s) = \arg\max_a \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma V^*(s')]$$

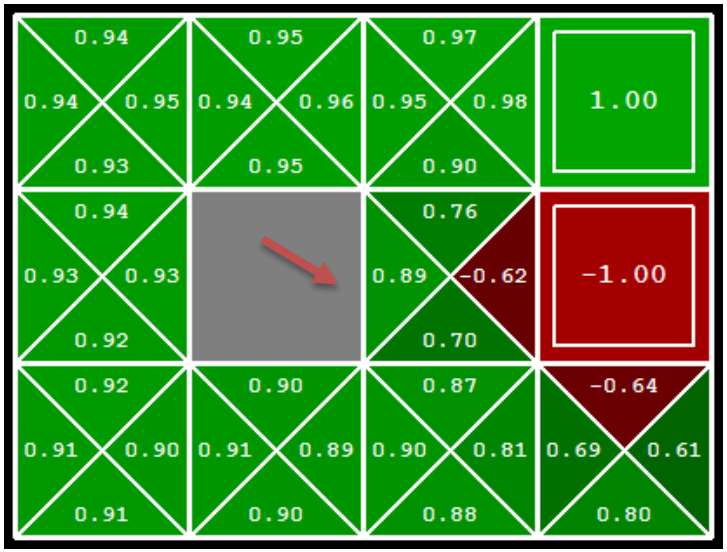
- Этот шаг называется извлечением политики, т.к. он позволяет определить действие ***a***, обеспечивающее получение максимальной ожидаемой суммарной награды в состоянии ***s*** по значениям V*(s).

Определение действий через q-ценности

- Предположим, что имеются оптимальные значения q-ценностей
- Каким образом нам следует действовать?
  - полностью тривиальное решение!

$$\pi^*(s) = \arg\max_a Q^*(s, a)$$

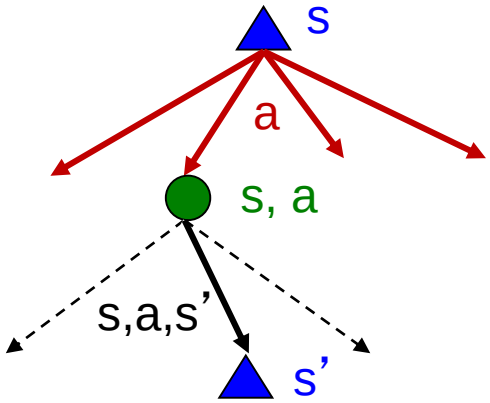
- Важный урок: ***проще выбирать действия по q-ценностям, чем по ценностям состояний!***



Проблемы с итерациями по значениям ценности состояний

- Итерация по ценностям V(s) основана на повторении обновлений значений в соответствии с выражением:

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s') \right]$$

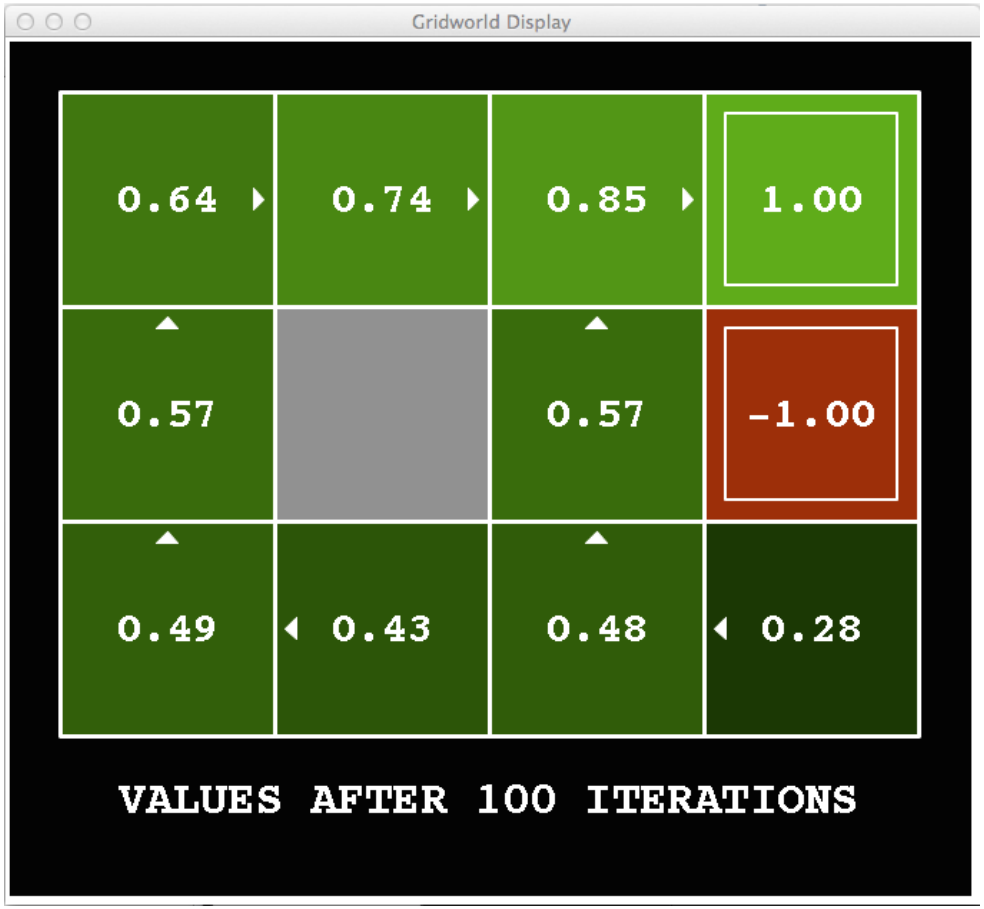


- Проблема 1: сложность – $O(S^2A)$ на 1 итерацию. Поэтому итерация по значениям может быть очень медленной
- Проблема 2: “max” для каждого состояния изменяется редко
- Проблема 3: Если мы пытаемся определить оптимальную политику , используя итерации по значениям для V(s), то мы сталкиваемся с большим объемом вычислений. В то время как политика, определяемая на основе метода извлечения политики, часто сходится задолго до того, как начинают сходиться ценности состояний
- Указанные проблемы решаются в **методе итерации по политикам**.

k=12

Шум = 0.2
Дисконт = 0.9
Текущая награда = 0

k=100



Шум = 0.2
Дисконт = 0.9
Текущая награда = 0

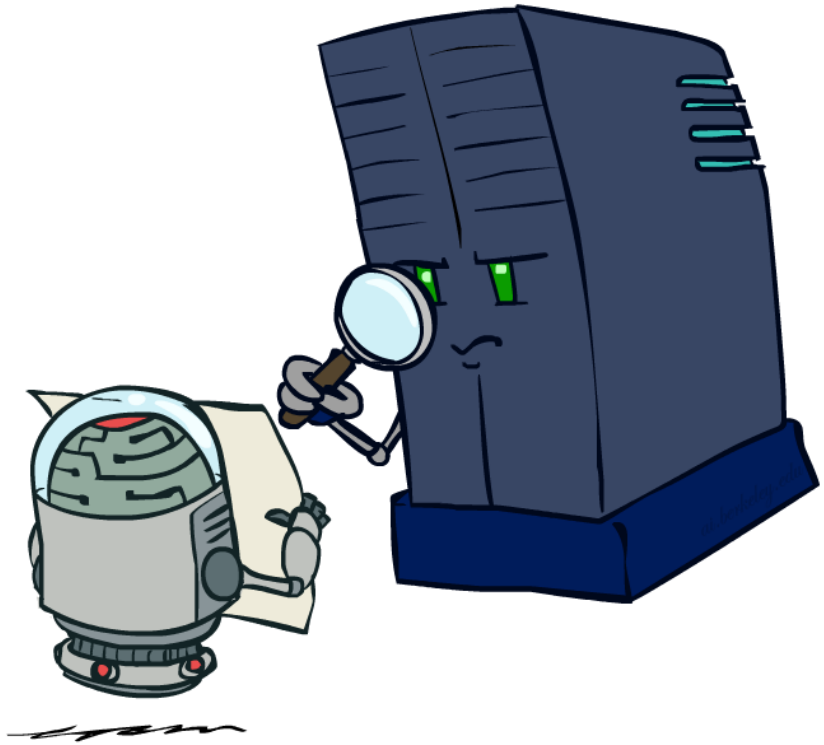
Итерации по политикам (стратегиям)

- Это альтернативный подход по отношению к итерациям по значениям V(s):
  - Шаг 1: Оценка политики (policy evaluation): вычисляем итеративно ценности состояний при некоторой выбранной (фиксированной) политике (это не оптимальные ценности!) пока значения не сойдутся
  - Шаг 2: Улучшение политики (policy improvement): обновляем политику, используя одношаговое предсказание с полученными (но не оптимальными!) значениями ценностей
  - Повторять шаги, пока политика не сойдется

Это алгоритм итерации по политикам.

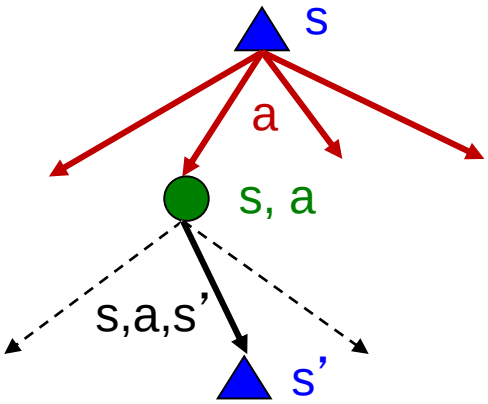
Найденная политика оптимальна! Может сходиться (намного) быстрее

Шаг 1. Оценка политики (policy evaluation)

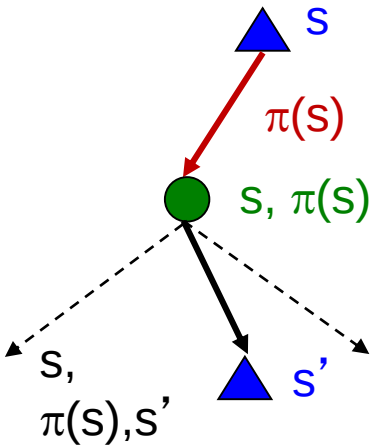


Фиксация политики

Итерации по V(s): выполнить оптимальное действие



Действовать в соответствии с π



- В Expectimax деревьях ищут max по всем действиям, чтобы вычислить оптимальную ценность состояния

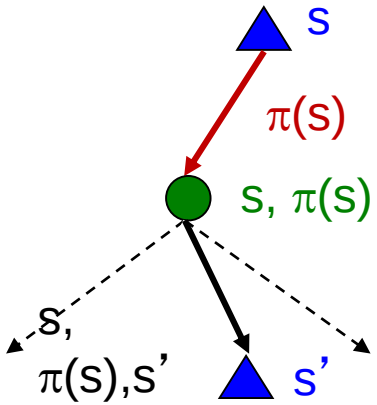
- Если зафиксировать некоторую политику $\pi(s)$, то дерево становится проще – только одно действие в состоянии s
  - … хотя ценность состояния будет зависеть от зафиксированной политики

Вычисление ценности состояний при фиксированной политике

- Определение ценности состояния s при фиксированной политике π:

  $V^{\pi}(s)$ = ожидаемая полная дисконтированная награда, начиная с состояния **s** при следовании политике π

- Рекурсивное соотношение (одно-шаговое предсказание на основе уравнения Беллмана):



$$V^{\pi}(s) = \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V^{\pi}(s')]$$

Ценность состояния **s** при выборе (фиксации) **π(s)**

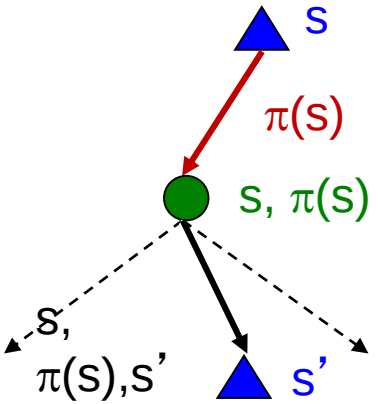
Оценка политики

- Как вычислять $\mathbf{V^{\pi}(s)}$ при фиксированной политике π?

- **Способ 1:** Превратить рекурсивное уравнение Беллмана в обновление (подобно итерации по значениям)

$$V_0^{\pi}(s) = 0$$

$$V_{k+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V_k^{\pi}(s')]$$



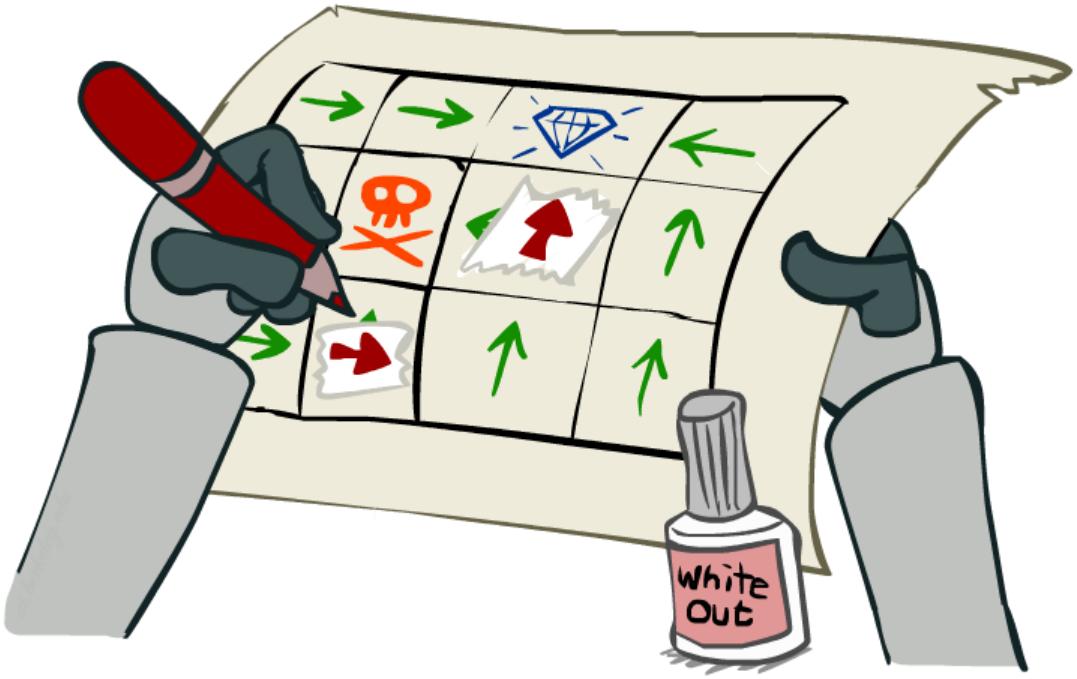
- Сложность: $O(S^2)$ на 1 итерацию. Нахождение $\mathbf{V^{\pi}(s)}$ таким способом на практике оказывается медленнее, чем способ 2 (ниже).

- **Способ 2**: Без операции max уравнения Беллмана представляют собой просто линейную систему

$$V^{\pi}(s) = \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V^{\pi}(s')]$$

Тогда оценка политики выполняется путем решения СЛАУ (например, в Matlab) относительно значений $\mathbf{V^{\pi}(s)}$ для всех состояний**.**

Шаг 2. Улучшение политики



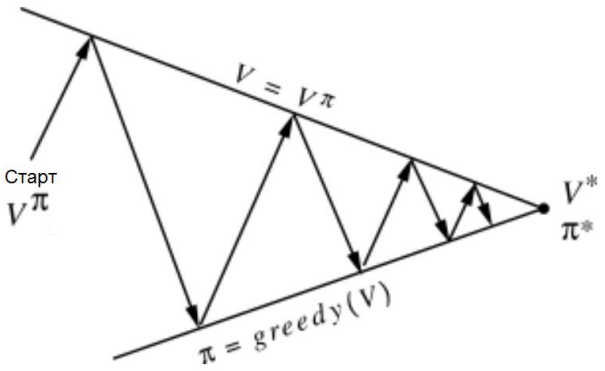
Итерации по политикам

- **Оценка политики**: при выбранной (фиксированной) текущей политике π_i находят ценности всех состояний :
  - Выполняют итерации, пока значения не сойдутся:

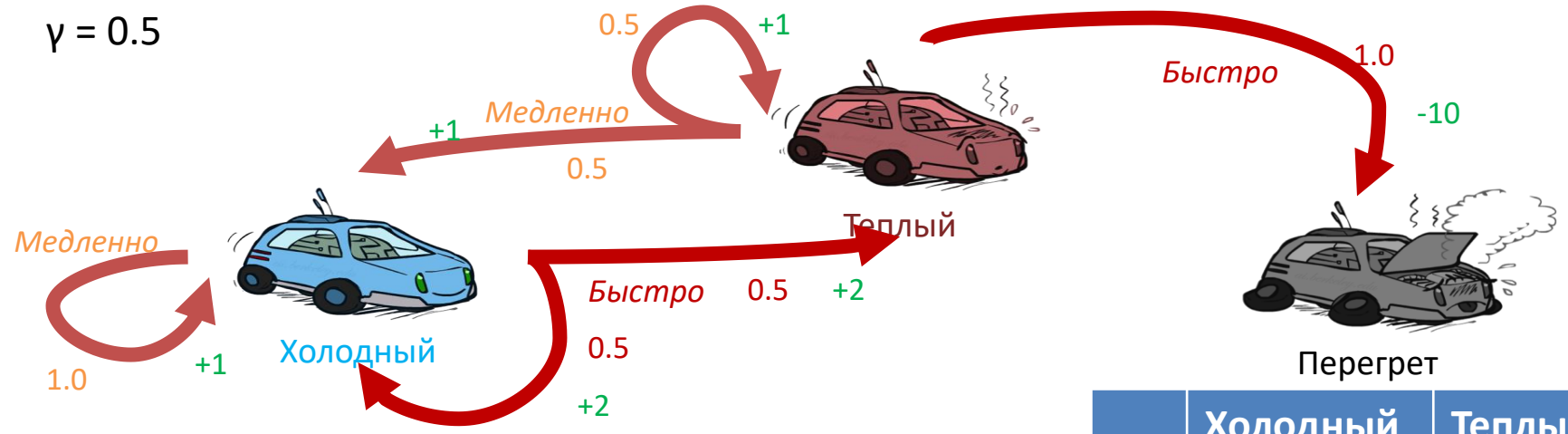
$$V_{k+1}^{\pi_i}(s) \leftarrow \sum_{s'} T(s, \pi_i(s), s') \left[R(s, \pi_i(s), s') + \gamma V_k^{\pi_i}(s') \right]$$

- **Улучшение политики**: После оценки текущей политики при фиксированных значениях ценностей состояний улучшают политику, применяя **метод извлечения политики:**
  - одношаговое улучшение:

$$\pi_{i+1}(s) = \arg\max_a \sum_{s'} T(s, a, s') \left[R(s, a, s') + \gamma V^{\pi_i}(s') \right]$$



Пример: итерации по политикам



	Холодный	Теплый	Перегрет
π_0	*Медленно*	*Медленно*	-

	Холодный	Теплый
π_0	*Медленно*	*Медленно*
π_1	*Быстро*	*Медленно*
π_2	*Быстро*	*Медленно*

$V^{\pi 0}$ (*холодный*)= [1+0,5 $V^{\pi 0}$ (*холодный*)]

$V^{\pi 0}$ (*теплый*)= 0,5 [1+0,5 $V^{\pi 0}$ (*холодный*)]+0,5 [1+0,5$V^{\pi 0}$ (*теплый*)]

	Холодный	Теплый	Перегрет
$V^{\pi 0}$	2	2	-

$\pi_1 = \pi_2 = \pi^*$

π_1(*холодный*) = argmax {*медленно*: 1 *[1+0,5*2]; *быстро*: 0,5*[2+0,5*2]+0,5*[2+0,5*2]}
= argmax{*медленно*: 2; *быстро*: 3}=***быстро***

π_1(*теплый*) = argmax{*медленно*: 0,5*[1+0,5*2]+0,5*[1+0,5*2]; *быстро*: 1*[-10+0,5*0]}
= argmax{*медленно*: 2; *быстро*: -10} = ***медленно***

Сравнение

Итерации по ценностям состояний и итерации по политикам вычисляют одни и те же значения

В итерации по ценностям состояний:

- Каждая итерация обновляет как ценности, так и (неявно) политику.
- Мы не отслеживаем политику, но выбираем максимум ценности по действиям, что неявно определяет ее.

В итерации по политикам:

- Мы выполняем несколько проходов, которые обновляют ценности состояний при фиксированной политике (каждый проход выполняется быстро, потому что мы рассматриваем только одно действие, а не все из них). Это называется оценкой политики.
- После оценки политики выбирается новая политика (подобно итерации по ценностям), которая таким образом улучшается.

Оба подхода соответствуют алгоритмам динамического программирования для решения MDP.

Заключение : MDP алгоритмы

Когда использовать каждый из алгоритмов?

Итерация по ценностям: используется для вычисления оптимальных ценностей состояний путем их итеративного обновления до сходимости.

Оценка политики: используется для вычисления ценности состояний в рамках определенной (фиксированной) политики.

Извлечение политики: используется для определения политики при известных значениях ценности состояний. Если состояния оптимальны, то и политика оптимальна. Этот метод используется после выполнения итерации по ценностям для определения оптимальной политики на основе оценки оптимальных значений состояний; или как подпрограмма в ходе итераций по политикам, чтобы определить лучшую политику для текущих оценок ценностей состояний.

Итерация по политикам: метод, который инкапсулирует как оценку политики, так и извлечение политики, и используется для итеративного схождения к оптимальной политике. Он имеет тенденцию превосходить по производительности итерации по ценностям в силу того, что политики обычно сходятся намного быстрее, чем ценности состояний.