Аппроксимация и обобщение в задачах обучения с подкреплением

Ограничения табличного представления

- До сих пор мы полагали, что функции ценности состояний (действий) могут быть сохранены таблице
  - Если состояний много, то
    - Для хранения нужно много места.
    - Для заполнения и обработки нужно много времени.
    - Для заполнения нужно много данных.
- Каким образом опыт, полученный на небольшом числе состояний, может быть перенесён на большее их число? – Обобщение.

Аппроксимация

- Обобщение, которое нам нужно, состоит в том, чтобы по примерам значений функции ценности в некоторых точках восстановить её значения в остальных.
- Это задача аппроксимации.
- Аппроксимация является примером задачи обучения с учителем.

Ценность состояний и аппроксимация

- Мы хотим определить V^{π}.
- Будем аппроксимировать эту функцию на каждом шаге t функцией V_t, полностью определяемой вектором параметров θ_t. $|\theta_t| << |S|$.
- Аппроксимация может вычисляться:
  - Нейронной сетью
  - Деревьями решений
  - …

Ценность состояний и аппроксимация

- Рассмотренные нами методы изменяли значения функции ценности для одного из состояний s в направлении некоторого значения v. Обозначим это $s \to v$.
  - DP: $s_t \to E\{r_{t+1} + \gamma V(s_{t+1}) | s_t = s\}$
  - Монте-Карло: $s_t \to R_t$.
  - TD(0): $s_t \to r_{t+1} + \gamma V(s_{t+1})$.
  - TD(λ): $s_t \to R^{\lambda}_t$.

- В случае аппроксимации пару $s \to v$ мы можем рассматривать как обучающий пример.

Ценность состояний и аппроксимация

- Для оценки качества аппроксимации обычно используется среднее значение квадрата ошибки:

$$MSE(\vec{\theta}_t) = \sum_{s \in S} P(s) \left[V^{\pi}(s) - V_t(s) \right]^2$$

где P(s) – распределение веса ошибки для разных состояний.

  - например, может совпадать с частотой, с которой агент попадает в состояния, действуя согласно своей стратегии.

- Целью аппроксимации является, как правило, поиск такого вектора θ^*, что:

$$MSE(\vec{\theta}^*) \leq MSE(\vec{\theta})$$

Градиентный спуск

- Имеется вектор параметров:

$$\vec{\theta}_t = (\theta_t(1), \theta_t(2), \ldots, \theta_t(n))^T$$

- $V_t(s)$ являются непрерывно-дифференцируемыми функциями от θ_t для всех s.

- На каждом шаге мы получаем новый обучающий пример вида $s_t \rightarrow V^{\pi}(s_t)$.

- Пусть распределение состояний, с которым мы получаем примеры, соответствует P.

Градиентный спуск

- Если мы хотим уменьшать MSE, то метод градиентного спуска говорит нам, что мы должны двигаться в направлении антиградиента этой функции:

$$\vec{\theta}_{t+1} = \vec{\theta}_t - \frac{1}{2}\alpha\nabla_{\vec{\theta}_t}\Big[V^{\pi}(s_t) - V_t(s_t)\Big]^2$$

$$= \vec{\theta}_t + \alpha\Big[V^{\pi}(s_t) - V_t(s_t)\Big]\nabla_{\vec{\theta}_t}V_t(s_t),$$

$$\nabla_{\vec{\theta}_t} f(\vec{\theta}_t) = \left(\frac{\partial f(\vec{\theta}_t)}{\partial \theta_t(1)}, \frac{\partial f(\vec{\theta}_t)}{\partial \theta_t(2)}, \ldots, \frac{\partial f(\vec{\theta}_t)}{\partial \theta_t(2)}\right)^T.$$

Градиентный спуск

- В случае, если мы имеем только приближения v_t к значениям функции $V^{\pi}(s_t)$ в виде примеров $s_t \rightarrow v_t$, то получим следующий метод градиентного спуска для аппроксимации функции ценности состояний:

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \Big[v_t - V_t(s_t) \Big] \nabla_{\vec{\theta}_t} V_t(s_t).$$

- Если v_t является несмещённой оценкой $V^{\pi}(s_t)$, то указанный алгоритм сходится к локальному минимуму V^{π}.

Градиентный спуск и TD(λ)

- Для TD(λ) $s_t \rightarrow R^{\lambda}_t$, соответственно, получаем:

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \left[R^{\lambda}_t - V_t(s_t) \right] \nabla_{\vec{\theta}_t} V_t(s_t).$$

- Или

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \delta_t \vec{e}_t,$$

$$\delta_t = r_{t+1} + \gamma V_t(s_{t+1}) - V_t(s_t),$$

$$\vec{e}_t = \gamma \lambda \vec{e}_{t-1} + \nabla_{\vec{\theta}_t} V_t(s_t), \qquad \vec{e}_0 = \vec{0}$$

Алгоритм TD(λ) с градиентным спуском

Инициализация: θ – произвольно

Повторять (для всех эпизодов)

e←0

s – начальное состояние

Повторять (для всех шагов эпизода)

a ← действие для s согласно π.

Выполнить a, получить s’ и r.

$\delta \leftarrow r+\gamma V(s')-V(s)$

$e \leftarrow \gamma\lambda e+\nabla_{\theta t}V(s)$

$\theta \leftarrow \theta+\alpha\delta e$

s ← s’

Линейные методы

- Функции V_t являются линейными функциями параметров $\theta_t=(\theta_t(1),\ldots,\theta_t(n))$.
- Для каждого состояния s имеется вектор свойств:

$$\vec{\phi}_s = (\phi_s(1), \phi_s(2), \ldots, \phi_s(n))^T$$

- Аппроксимация строится как скалярное произведение векторов:

$$V_t(s) = \vec{\theta}_t^T \vec{\phi}_s = \sum_{i=1}^{n} \theta_t(i)\phi_s(i).$$

Линейные методы и градиентный спуск

- Градиент аппроксимирующей функции по θ есть

$$\nabla_{\vec{\theta}_t} V_t(s) = \vec{\phi}_s.$$

- В линейном случае имеется только один минимум функции ошибки

- TD(λ) с линейной аппроксимацией сходится к θ_∞, такому что

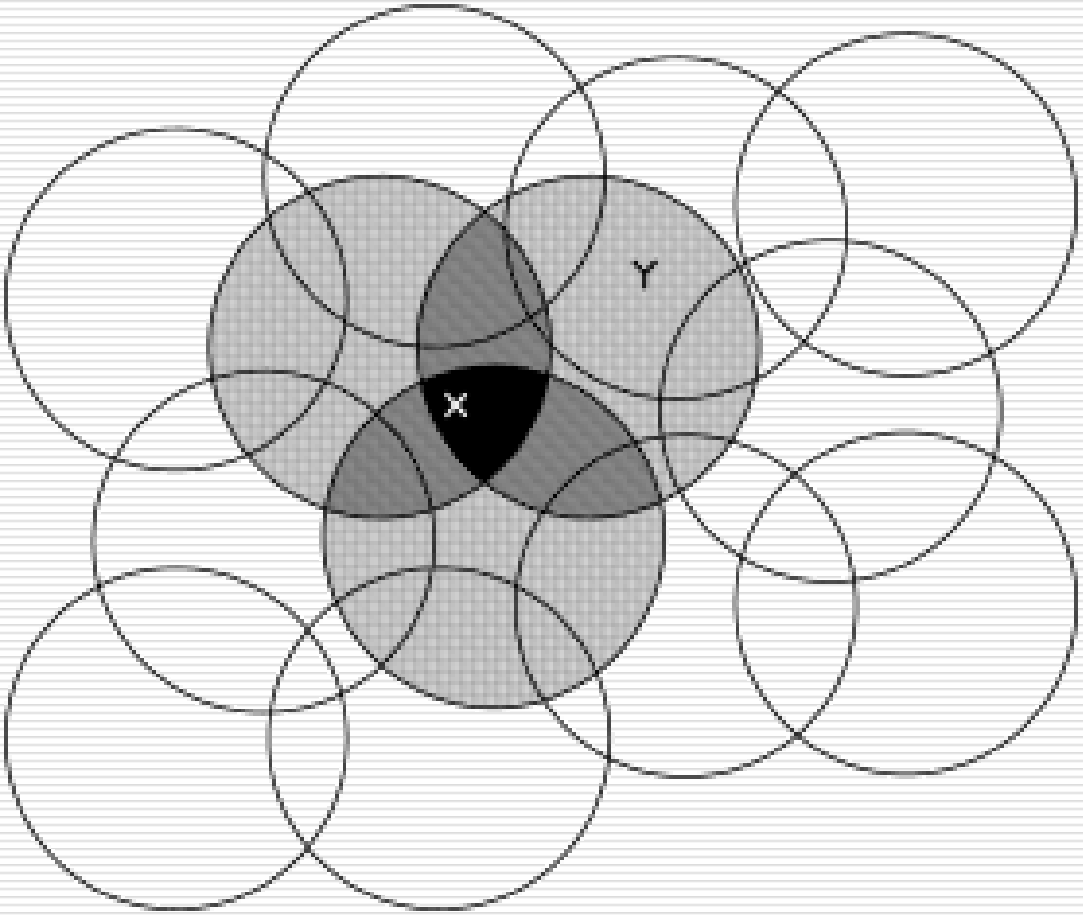
$$MSE(\vec{\theta}_\infty) \leq \frac{1-\gamma\lambda}{1-\gamma} MSE(\vec{\theta}^*).$$

Грубое кодирование

Если состояние попадает внутрь области свойства – свойство присутствует.

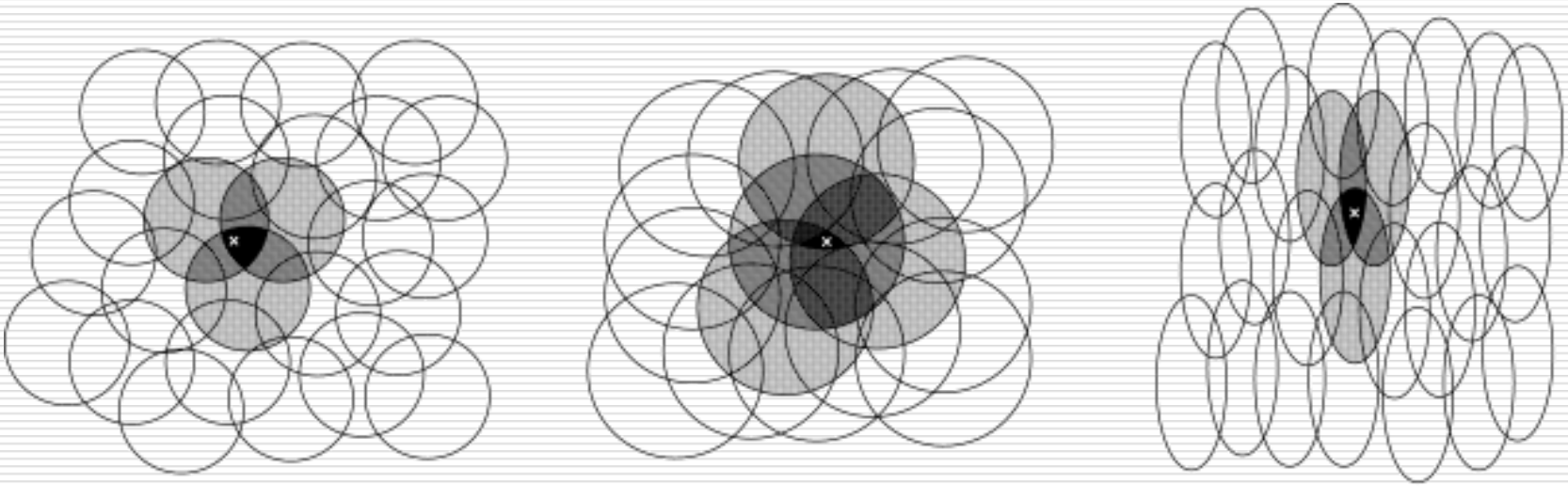
Если состояние не попадает внутрь – свойство отсутствует

- Двоичное кодирование

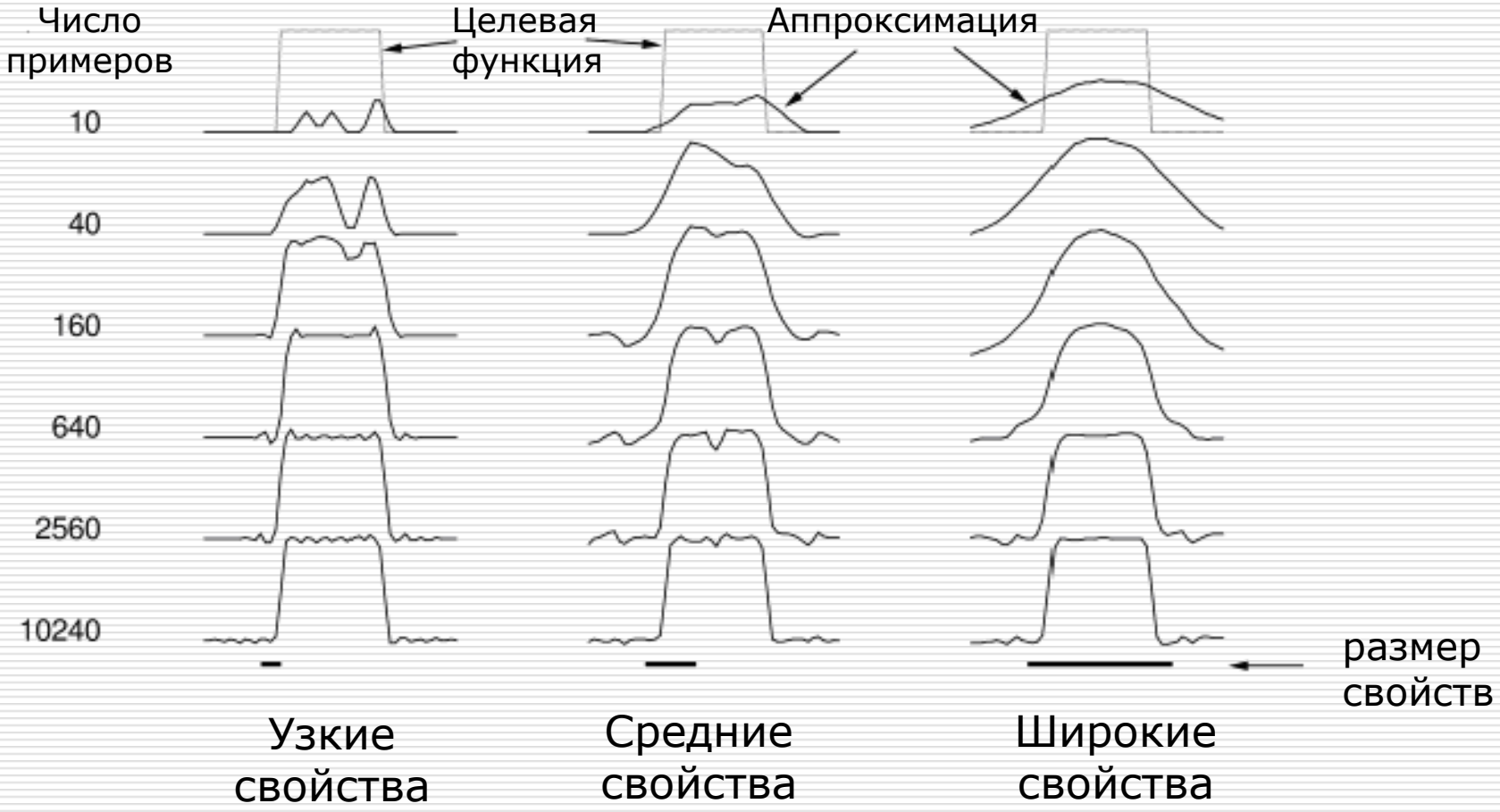


Грубое кодирование и обобщение

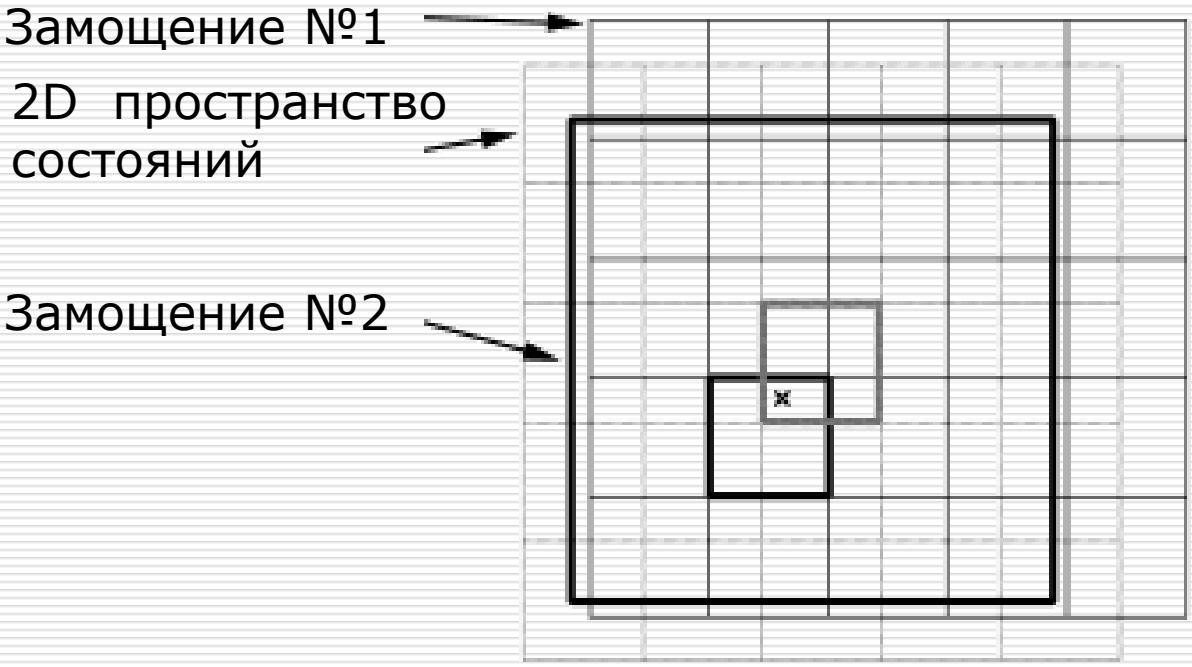
- Размер и форма свойств определяет способность к обобщению



Точность грубого кодирования

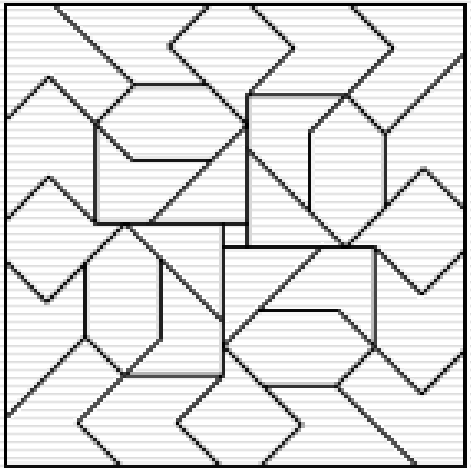


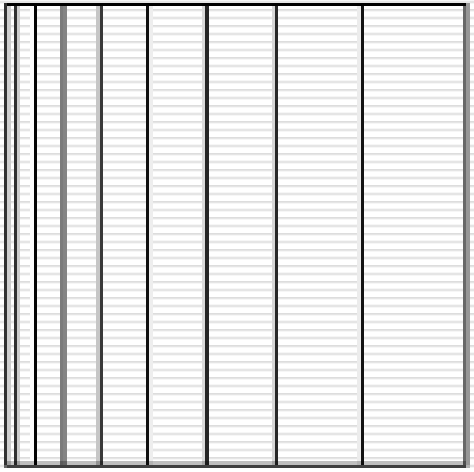
Замощения

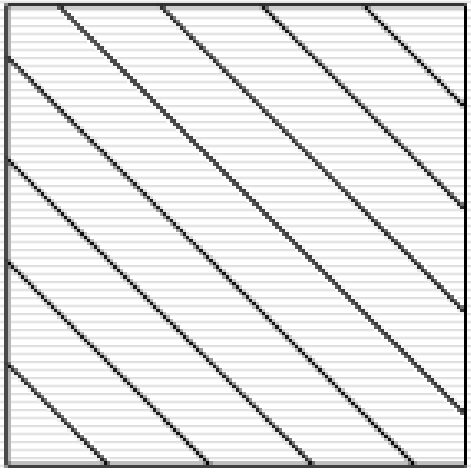


- Форма замощения – генерализация
- Число замощений – точность

Замощения



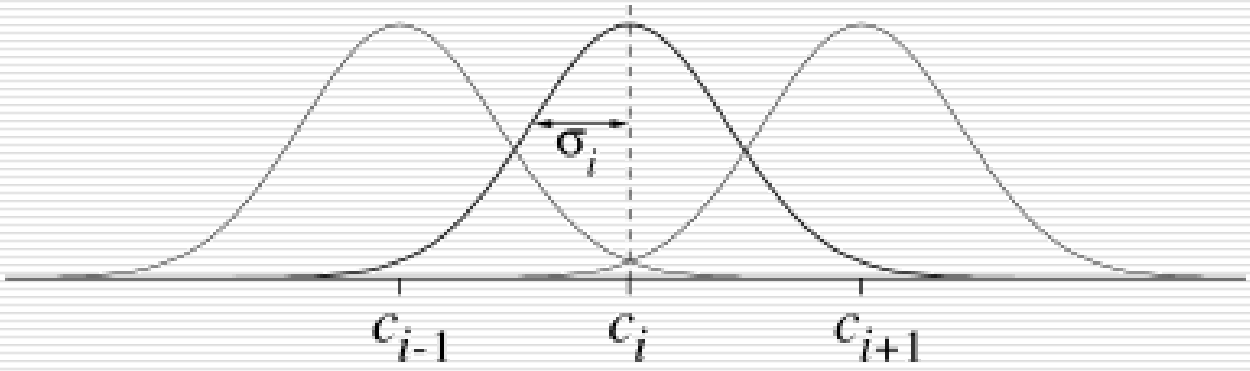




Радиально-базисные функции

- Свойства

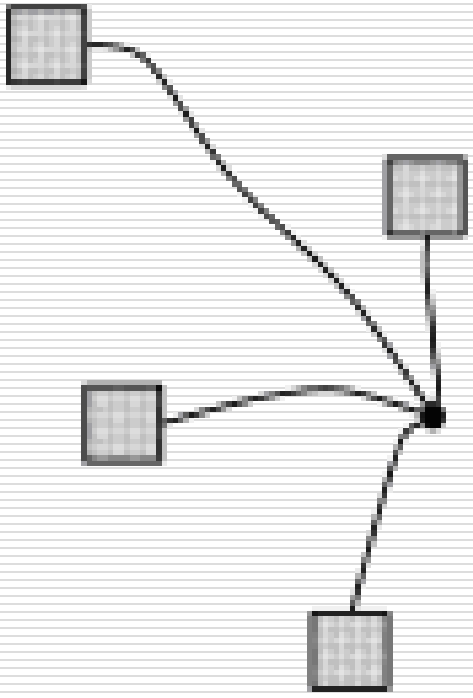
$$\phi_s(i) = \exp\left(-\frac{||s - c_i||^2}{2\sigma_i^2}\right).$$



Нечёткие функции

- Свойства μ_i - функция принадлежности.
- Аппроксимация: $\tilde{V}(s) = \sum_{i \in I} \mu_i(s)\theta_i$
- Автоматическое определение числа признаков (Jiaan Zeng, Yinghua Han, 2009) ASP-FCMAC:
  - Когда разделять признак?
    - Если стабилизировалась ошибка Беллмана (усреднённая за несколько шагов).
  - Какой признак разделять?
    - Наиболее посещаемый
    - С максимальной суммой обновлений – на практике работает лучше.

Хэширование



Кодирование Канерва

- Определяются состояния-прототипы
  - Например, случайно
- Состояния считаются близкими, если совпадают значения по достаточному числу измерений, даже если значения по остальным измерениям не похожи
  - Например, двоичное пространство состояний (состояние определяется двоичным вектором) и расстояние Хэмминга (число совпадающих бит).
- Сложность аппроксимации зависит от числа состояний-прототипов, а не от размерности входного пространства

Управление и аппроксимация

- Мы хотим построить $Q_t \approx Q^\pi$.

- Используем примеры вида $s_t, a_t \rightarrow v$:
  - Монте-Карло: $s_t, a_t \rightarrow R_t$.
  - SARSA: $s_t, a_t \rightarrow r_{t+1} + \gamma Q(s_{t+1}, a_{t+1})$.
  - …

- Алгоритм градиентного спуска даёт правило

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \left[v_t - Q_t(s_t, a_t) \right] \nabla_{\vec{\theta}_t} Q_t(s_t, a_t).$$

Управление и аппроксимация. TD(λ)

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \delta_t \vec{e}_t,$$

$$\delta_t = r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t),$$

$$\vec{e}_t = \gamma \lambda \vec{e}_{t-1} + \nabla_{\vec{\theta}_t} Q_t(s_t, a_t),$$

$$\vec{e}_0 = \vec{0}$$

Управление и аппроксимация. TD(λ)

- Улучшение стратегии и выбор действий
  - Для дискретного и небольшого пространства действий можем непосредственно вычислять жадное действие и ε-жадную стратегию.

- Можем использовать одну и ту же или разные стратегии для управления и оценки
  - Одна стратегия - Sarsa(λ)
  - Разные стратегии – Watkins's Q(λ)

Sarsa(λ)

Инициализация: θ – произвольно

Повторять для всех эпизодов
- $e \leftarrow 0$
- $s,a \leftarrow$ начальные состояние и действие
- $F_a \leftarrow$ множество свойств, присутствующих в (s,a)
- Повторять для всех шагов эпизода
  - Для всех $i \in F_a$
    - $e(i) \leftarrow e(i)+1$ // $e(i) \leftarrow 1$
  - Выполнить a, получить s’ и r.
  - $\delta \leftarrow r-\sum_{i \in Fa}\theta(i)$
  - С вероятностью $1-\varepsilon$
    - Для всех $a \in A(s)$
      - $F_a \leftarrow$ множество свойств в (s,a)
      - $Q_a \leftarrow \sum_{i \in Fa}\theta(i)$
    - $a=\arg\max_a Q_a$
  - иначе
    - $a \leftarrow$ случайное из $A(s)$
    - $F_a \leftarrow$ множество свойств в (s,a)
    - $Q_a \leftarrow \sum_{i \in Fa}\theta(i)$
  - $\delta \leftarrow \delta+\gamma Q$
  - $\theta \leftarrow \theta+\alpha\delta e$
  - $e \leftarrow \lambda\gamma e$

Watkins's Q(λ)

Инициализация: θ – произвольно

Повторять для всех эпизодов
- $e \leftarrow 0$
- s,a ← начальные состояние и действие
- F_a ← множество свойств, присутствующих в (s,a)
- Повторять для всех шагов эпизода
  - Для всех $i \in F_a$
    - $e(i) \leftarrow e(i)+1$ // $e(i) \leftarrow 1$
  - Выполнить a, получить s' и r.
  - $\delta \leftarrow r-\sum_{i \in Fa}\theta(i)$
  - Для всех $a \in A(s)$
    - F_a ← множество свойств в (s,a)
    - $Q_a \leftarrow \sum_{i \in Fa}\theta(i)$
  - $\delta \leftarrow \delta+\gamma max_a Q_a$
  - $\theta \leftarrow \theta+\alpha\delta e$
  - С вероятностью $1-\varepsilon$
    - Для всех $a \in A(s)$
      - $Q_a \leftarrow \sum_{i \in Fa}\theta(i)$
    - $a=arg\ max_a Q_a$
    - $e \leftarrow \lambda\gamma e$
  - иначе
    - a ← случайное из A(s)
    - $e \leftarrow 0$

Замещающие следы и аппроксимация

- При использовании замещающих следов мы заменяли значение следа на 1 для состояния, в которое мы попадали. При использовании аппроксимации след соответствует группе состояний, а не одному из них.
- Если у нас двоичные свойства, то мы можем использовать свойства так, как использовали состояния – устанавливать след для свойства, когда оно посещается.
- При использовании следов для действий в некоторых случаях было полезно сбрасывать след для всех альтернативных действий из текущего состояния.
- Аналогичным образом можно поступить и при аппроксимации:
  - Сбросим след для присутствующих для текущего состояния и не присутствующих при текущем действии свойствах.
  - Установим след равный 1 для свойств, пристуствующих при текущем действии и состоянии.

Рекурсивные алгоритмы и аппроксимация

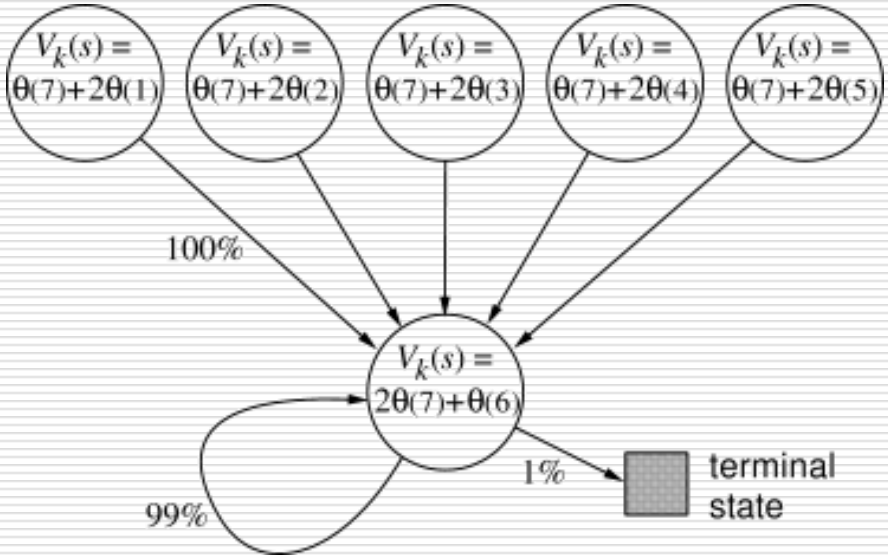
- У нас есть два вида алгоритмов вычисления функций ценности:
  - Используют (не точные) значения функции от одних аргументов для вычисления приближения для других аргументов:
    - TD(0)
    - DP
  - Не используют другие значения
    - Монте-Карло
    - TD(1)
  - TD(λ) для промежуточных значений совмещает эти два варианта

Рекурсивные алгоритмы и аппроксимация

- Рассмотрим случай линейной аппроксимации и градиентного спуска
- Нерекурсивные алгоритмы находят локальный минимум при произвольном распределении обучающих примеров
- Рекурсивные алгоритмы находят близкое к минимуму значение и только для распределения, соответствующего стратегии.
  - Например, для TD(λ)

$$MSE(\vec{\theta}_\infty) \leq \frac{1-\gamma\lambda}{1-\gamma} MSE(\vec{\theta}^*).$$

Контрпример Baird’a



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

$$\vec{\theta}_t = \vec{0}$$

- Используем DP:

$$\vec{\theta}_{k+1} = \vec{\theta}_k + \alpha \sum_s \Big[E\left\{ r_{t+1} + \gamma V_t(s_{t+1}) \mid s_t = s \right\} - V_k(s) \Big] \nabla_{\vec{\theta}_k} V_k(s).$$

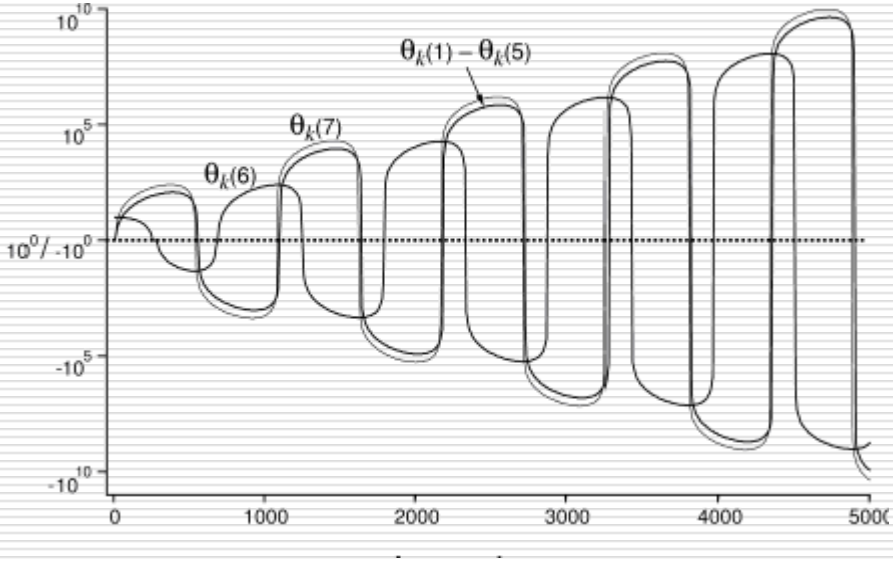
- Равномерное распределение примеров

Контрпример Baird’a

- Параметры и начальное значение:

$$\gamma = 0.99 \quad \vec{\theta}_0 = (1, 1, 1, 1, 1, 10, 1)^T$$

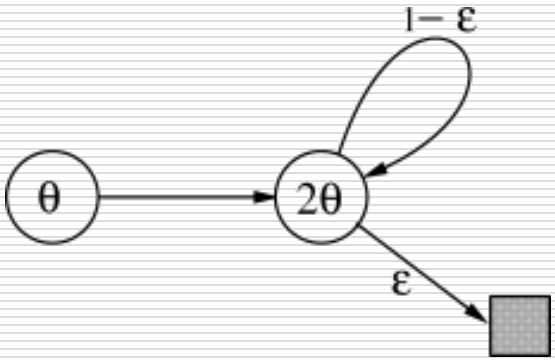
$$\alpha = 0.01$$



Контрпример Baird’a

- Как мы можем обеспечить сходимость?
- На каждом шаге будем доходить до минимума
- Так как свойства в примере Baird’a линейно-независимы, то мы найдём точное приближение и метод будет эквивалентен обычному DP.
- Однако это не помогает в случаях, когда точное решение не может быть достигнуто.

Контрпример Tsitsiklis и Van Roy‘я



$$\theta_k = 0$$

- DP:

$$\theta_{k+1} = \arg\min_{\theta\in\Re} \sum_{s\in\mathcal{S}} \Big[V_\theta(s) - E_\pi\{ r_{t+1} + \gamma V_{\theta_k}(s_{t+1}) | s_t = s \} \Big]^2$$

$$= \arg\min_{\theta\in\Re} \Big[\theta - \gamma 2\theta_k \Big]^2 + \Big[2\theta - (1-\varepsilon)\gamma 2\theta_k \Big]^2$$

$$= \frac{6-4\varepsilon}{5}\gamma\theta_k,$$

- Расходится если

$$\begin{cases} \gamma > \frac{5}{6-4\varepsilon} \\ \theta_0 \neq 0 \end{cases}$$

Как обеспечить сходимость?

- Использовать методы аппроксимации, которые не интерполируют:
  - метод ближайшего соседа;
  - взвешенная локальная регрессия;
  - не подходят: замощения и обратное распространение.
- Использовать пакетный режим.
- Использовать другие критерии для аппроксимации.

Ошибка Беллмана

- Средняя ошибка ожидаемого одношагового возврата

$$\sum_s P(s)\Big[E_\pi\{r_{t+1}+\gamma V_t(s_{t+1})|s_t=s\}-V_t(s)\Big]^2.$$

- С такой ошибкой и градиентным спуском получаем

$$\begin{aligned}\vec{\theta}_{t+1} &= \vec{\theta}_t-\frac{1}{2}\alpha\nabla_{\vec{\theta}_t}\Big[E_\pi\{r_{t+1}+\gamma V_t(s_{t+1})\}-V_t(s_t)\Big]^2\\ &= \vec{\theta}_t+\alpha\Big[E_\pi\{r_{t+1}+\gamma V_t(s_{t+1})\}-V_t(s_t)\Big]\Big[\nabla_{\vec{\theta}_t}V_t(s_t)-E_\pi\big\{\nabla_{\vec{\theta}_t}V_t(s_{t+1})\big\}\Big],\end{aligned}$$

- Сходится, однако работает только для детерминированных задач или когда есть модель, так как нужно два независимых примера для следующего состояния s_{t+1}.

Спроецированная ошибка Беллмана

- Методы, использующие аппроксимацию, не могут найти минимум ошибки Беллмана, так как значение $E_\pi\{r_{t+1} + \gamma V_t(s_{t+1})|s_t = s\} \stackrel{\text{def}}{=} TV$ не может быть точно представлено в параметрической форме.
- Пусть Π – оператор проекции, который отображает произвольную функцию v в ближайшую функцию, представимую с помощью аппроксиматора:
$$\Pi v = V_\theta, \theta = \arg\min_\theta \|V_\theta - v\|^2 .$$
- Спроецированная ошибка Беллмана:
$$MSPBE(\theta) = \|V_\theta - \Pi TV\|^2.$$
- Градиентный спуск по $MSPBE$ приводит к алгоритму Greedy-GQ.

Greedy-GQ

Инициализация:

θ – произвольно

$w \leftarrow 0$

Повторять для всех эпизодов

$s \leftarrow$ начальные состояние

Повторять для всех шагов эпизода

$a \leftarrow \epsilon$- жадное действие для s.

Выполнить a, получить s′ и r.

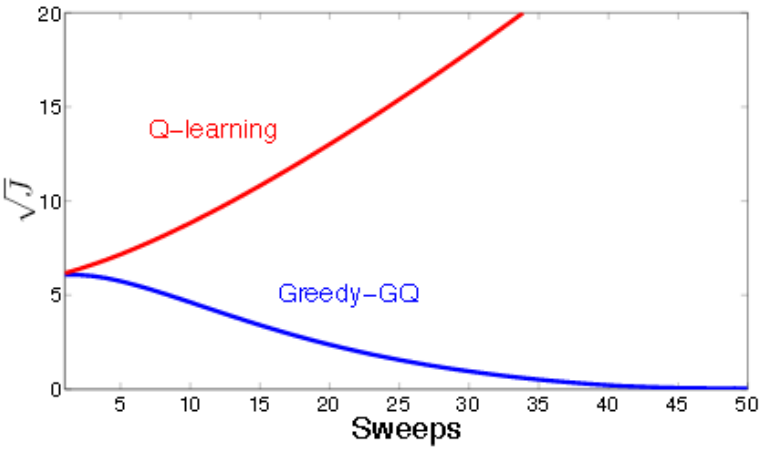
$a' \leftarrow \arg\max_a Q_\theta(s,a)$

$\delta \leftarrow r + \gamma Q_\theta(s',a') - Q_\theta(s,a)$

$\theta \leftarrow \theta + \alpha\left[\delta\phi(s,a) - \gamma\left(w^T\phi(s,a)\right)\phi(s',a')\right]$

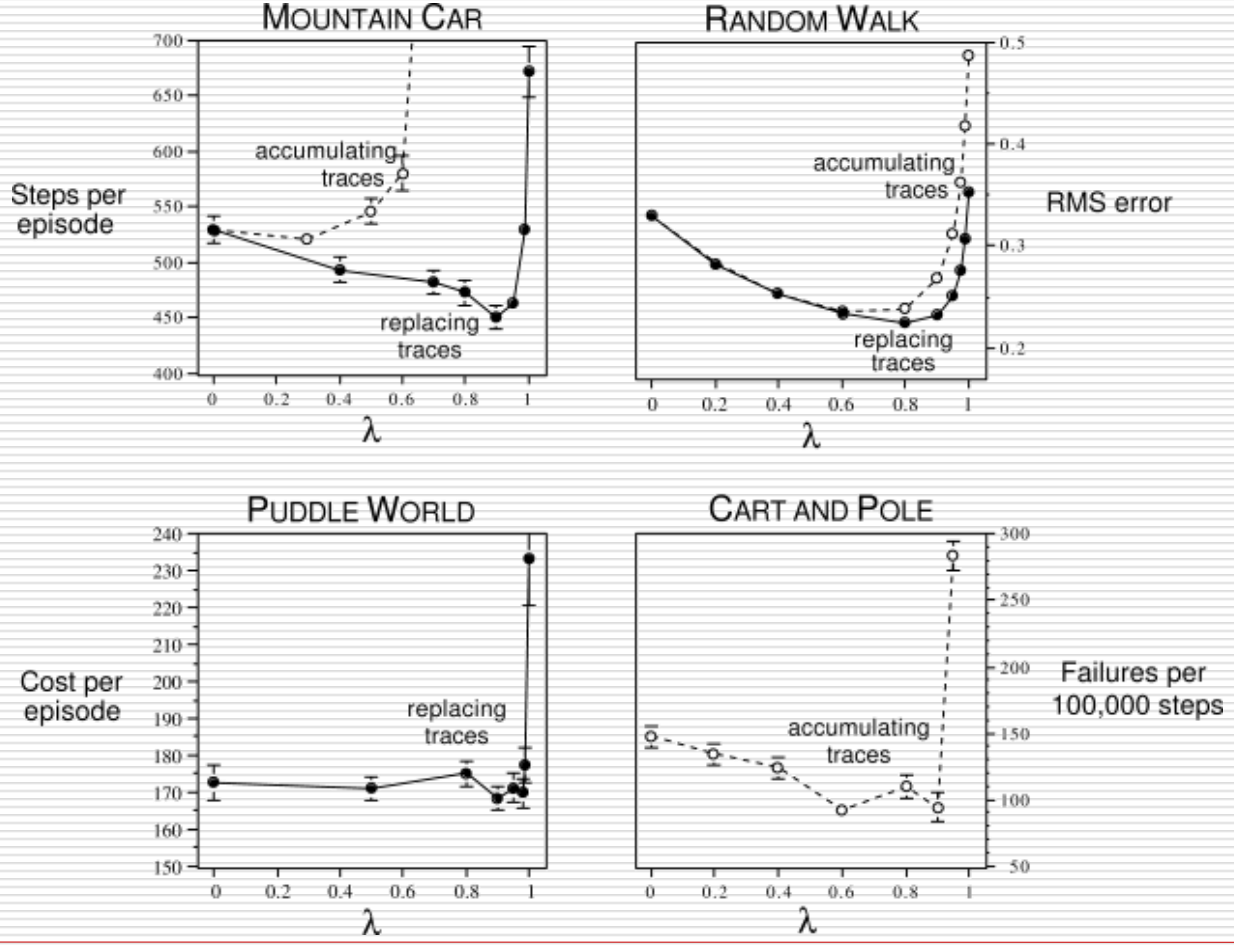
$w \leftarrow w + \beta\left[\delta - \left(w^T\phi(s,a)\right)\right]\phi(s,a)$

$s \leftarrow s'$



- Greedy-GQ сходится при использовании линейной аппроксимации.

Следует ли использовать рекурсивные алгоритмы?



Заключение

- Для применения обучения с подкреплением во многих практических задачах необходимо использовать обобщение.
- Такой результат может быть достигнут использованием методов обучения с учителем.
- Важную роль играет выбор свойств.
- При совмещении аппроксимации, рекурсивных алгоритмов и разных распределений примеров для обучения возникает возможность расхождения алгоритма обучения.
- Тем не менее, на практике такой вариант часто работает лучше варианта не рекурсивных методов.

Задача 3

- Действия: a=1, a=-1, a=0
- Перемещения машины:
  - Координаты $x_{t+1} = bound\left[x_t + \dot{x}_{t+1}\right]$
  - Скорость $\dot{x}_{t+1} = bound\left[\dot{x}_t + 0.001a_t + -0.0025\cos(3x_t)\right],$
  - Ограничения *bound:* $-1.2 \leq x_{t+1} \leq 0.5 \quad -0.07 \leq \dot{x}_{t+1} \leq 0.07$

    Если x достигает левой границы, то скорость сбрасывается в 0
- Подкрепление: r=-1 для всех шагов пока не достигнута цель, тогда r=0
- Алгоритм: Sarsa(λ)

  с линейной интерполяцией

замощение: 10 сеток 9x9

$\lambda = 0.9 \quad \varepsilon = 0 \quad \alpha = 0.05$