Решеточное обучение с подкреплением

метод Монте-Карло и сходство

Д.Виноградов

ВЦ им. Дородницына РАН

25 июня 2025 г.

Парадигма обучения с подкреплением

В обучении с подкреплением игрок (называемый **агентом**) взаимодействует с оппонентом (**окружением**).
Окружение может находиться в одном из своих состояний $s \in \mathbf{S}$, наблюдаемым игроком. В каждом из состояний агент может совершить одно из нескольких действий $a \in \mathbf{A}$, зависящем от состояния.
Так как оппонент может использовать рандомизованную стратегию, то переход к новому наблюдаемому состоянию имеет условное вероятностное распределение

$$S_{t+1} \sim \mathcal{P}(\cdot|S_t, A_t) = \mathbb{P}\left[S_{t+1}|S_t, A_t\right]$$

оказаться в состоянии S_{t+1} при выборе (допустимого) действия A_t в состоянии S_t.
Игрок получает (непосредственное) **вознаграждение** $R_t \sim \mathcal{R}(\cdot|S_t, A_t)$ за выбор действия A_t в состоянии S_t.
Важно: Предполагается, что распределения $\mathcal{P}(\cdot|S_t, A_t)$ и $\mathcal{R}(\cdot|S_t, A_t)$ **стационарны**, т.е. не зависят от шага t. В играх вознаграждение зависит только от заключительного состояния, поэтому задается как $R_l = r(s_l)$ для детерминированной функции $r : \mathbf{S} \to \mathbb{R}$, причем, если состояние $s \in \mathbf{S}$ не является терминальным, то $r(s) = 0$.

Оптимальные политики

Целью игрока является выбор оптимальной рандомизованной политики $A_t \sim \pi(\cdot|S_t)$ — стратегии выбора действия $A_t \in \mathbf{A}$ для каждого состояния $S_t \in \mathbf{S}$, в котором он может находиться.
Оптимальность вычисляется относительно дисконтированной со скоростью дисконта $1 \geq \gamma \geq 0$ накопленной награды (для всех $S_0 = s_0$)

$$V^{\pi}(s_0) = \sum \gamma^k \cdot \mathbb{E}_{\mathcal{R}}\left[R_k(s_k, a_k)\right] \cdot \prod_{j=0}^{k} \mathcal{P}\left(s_{j+1}|s_j, a_j\right) \cdot \pi(a_j|s_j).$$

Суммирование идет по всем частичным траекториям $s_1, \ldots, s_n, s_{n+1}, \ldots$ до s_{k+1} и всем последовательностям действий $a_0, \ldots, a_k$.
Если вознаграждение выдается только после завершения партии, то суммируются только полные траектории $S_1, \ldots, s_{l-1}, s_l$

$$V^{\pi}(s_0) = \sum \gamma^{l-1} \cdot r(s_l) \cdot \prod_{j=0}^{l-1} \mathcal{P}\left(s_{j+1}|s_j, a_j\right) \cdot \pi(a_j|s_j).$$

Рекуррентные соотношения

Обозначив сумму по всем тректориям длины k

$$\sum \mathbb{E}_{\mathcal{R}} \left[R_{k-1}(s_{k-1}, a_{k-1}) \right] \cdot \prod_{j=0}^{k-1} \mathcal{P}\left(s_{j+1} | s_j, a_j\right) \cdot \pi(a_j | s_j)$$

через $\mathbb{E}_{\pi,\mathcal{P},\mathcal{R}} G_k$, можно выписать рекуррентное соотношение:

$$V^{\pi}(s_0) = \mathbb{E}_{\pi,\mathcal{P},\mathcal{R}} \left[\sum_{k=0}^{\infty} \gamma^k \cdot G_k | S_0 = s_0 \right] = \mathbb{E} \left[G_1 + \gamma \cdot \sum_{k=1}^{\infty} \gamma^{k-1} \cdot G_k | S_0 = s_0 \right] =$$

$$= \sum_{a_0, s_1} \mathbb{E}_{\pi,\mathcal{P},\mathcal{R}} \left[R_0(s_0, a_0) + \gamma \cdot \sum_{k=0}^{\infty} \gamma^k \cdot G_k | S_1 = s_1 \right] \cdot \mathcal{P}\left(s_1 | s_0, a_0\right) \cdot \pi(a_0 | s_0) =$$

$$= \sum_{a_0} \left(\mathbb{E}_{\mathcal{R}} \left[R_0(s_0, a_0) \right] + \sum_{s_1} \gamma \cdot V^{\pi}(s_1) \cdot \mathcal{P}\left(s_1 | s_0, a_0\right) \right) \cdot \pi(a_0 | s_0).$$

Для случая игр имеем

$$V^{\pi}(s_0) = \sum_{a_0, s_1} \left[r(s_1) + \gamma \cdot V^{\pi}(s_1) \right] \cdot \mathcal{P}\left(s_1 | s_0, a_0\right) \cdot \pi(a_0 | s_0).$$

Динамическое программирование

Если значения функции вознаграждения $r : \mathbf{S} \to \mathbb{R}$ ограничены, то при $\gamma < 1$ сжимающий оператор перехода

$$T^\pi V^\pi(s_0) = \sum_{a_0} \left\{ \sum_{s_1} [r(s_1) + \gamma \cdot V^\pi(s_1)] \cdot \mathcal{P}(s_1|s_0, a_0) \right\} \cdot \pi(a_0|s_0).$$

имеет неподвижную точку $V^\pi = T^\pi V^\pi$.
Обозначив оптимальную политику через π^*, а ее значение - через V^*, получим уравнение Беллмана:

$$V^*(s_0) = \max_{a_0} \left\{ \sum_{s_1} [r(s_1) + \gamma \cdot V^*(s_1)] \cdot \mathcal{P}(s_1|s_0, a_0) \right\}.$$

Аналогично, имеем $V^* = T^{\pi^*} V^*$. Тут используется непрерывность функции max и компактность ограниченных функций на конечном множестве $\mathbf{S} \times \mathbf{A}$.
Случай $\gamma = 1$ получается предельным переходом $\varepsilon \to 0$ для $\gamma = 1 - \varepsilon$.
Проблема: Распределение перехода состояний (в играх - стратегия оппонента) $\mathcal{P}(\cdot|s_0, a_0)$ неизвестно!

Q-values: оценки действий

В случае детерминированной стратегии можно применять жадный алгоритм

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

с использованием $Q - values$.
Функция $Q - values$ оценивает действия:

$$Q^\pi(s_0, a_0) = \sum r(s_l) \cdot \left(\prod \mathcal{P}(s_{j+1}|a_j, s_j) \cdot \pi(a_j|s_j) \right) \cdot \mathcal{P}(s_1|a_0, s_0).$$

Она соответствует выбору действия a_0 в начальном состоянии s_0, а затем применении политики π. Здесь суммирование идет по всем траекториям $s_1, \ldots, s_{l-1}, s_l$ до остановки и всем последовательностям действий $a_1, \ldots, a_{l-1}$, умножение по j от 1 до $l - 1$.
Она связана с V^π соотношением

$$Q^\pi(s_0, a_0) = \sum_{s_1} [r(s_1) + \gamma \cdot V^\pi(s_1)] \cdot \mathcal{P}(s_1|s_0, a_0).$$

Метод Монте-Карло

Для представления оценок действий оптимальной стратегии:

$$Q^*(s_0, a_0) = \mathbb{E}_{S_1 \sim \mathcal{P}} \left[r(S_1) + \gamma \cdot \max_{a_1} Q^*(S_1, a_1) \right]$$

можно использовать случайный выбор $S_1 \sim \mathcal{P}(\cdot|s_0, a_0)$ и обновить значение

$$Q(s_0, a_0) \leftarrow Q(s_0, a_0) + \frac{1}{n+1} \cdot \left[r(S_1) + \gamma \cdot \max_{a_1} Q(S_1, a_1) - Q(s_0, a_0) \right] =$$

$$= \frac{n}{n+1} \cdot Q(s_0, a_0) + \frac{1}{n+1} \cdot \left[r(S_1) + \gamma \cdot \max_{a_1} Q(S_1, a_1) \right].$$

Проблема: элементов $\mathbf{S} \times \mathbf{A}$ может оказаться слишком много, поэтому точные значения $Q - values$ заменяют на их приближение нейросетью.

Достоинства и недостатки нейросетей

Достоинства:

1. Нейросети экстраполируют оценки действий в ранее ненаблюдавшихся состояниях.
2. Обучение нейросетей сводится к умножению матриц (необходимо $O(n^{\log_2 7})$ операций умножения для перемножения $n \times n$-матриц). Применение Глубокого обучения привело к временной сложности $O(n^2)$, а техники Tensor-Train - к $O(n \cdot \log n)$.
3. Этапы обучения и вывода производятся над одной структурой данных (графом с весами).
4. Обучение нейросетей хорошо распараллеливается на GPGPU.

Недостатки:

1. Переобучение (галлюцинации) нейросетей.
2. Необъяснимость оптимальной политики на человеческом языке.
3. Невозможна передача знаний, кроме как дистилляцией.

Идея: сходство между состояниями

Сходство - это бинарная операция $\wedge : L \times L \to L$, задающая структуру нижней полурешетки с наименьшим элементом $\bot$. Элементы L называются **фрагментами**.
Операция сходства должна удовлетворять аксиомам **нижней полурешетки**:

$$x \wedge x = x \tag{1}$$

$$x \wedge y = y \wedge x \tag{2}$$

$$(x \wedge y) \wedge z = x \wedge (y \wedge x) \tag{3}$$

Определение

Порядок *на нижней полурешетке* $\langle L, \wedge \rangle$ *определяется так*

$$x \leq y \Leftrightarrow x \wedge y = x$$

Тривиальное сходство = **пустой фрагмент** $\bot$ есть наименьший элемент:

$$x \wedge \bot = \bot \tag{4}$$

Решетки

Конечную нижнюю полурешетку легко превратить в решетку (с операцией $\vee : L \times L \to L$, где $x \vee y = \bigwedge\{z \in L : x \leq z, y \leq z\}$), добавив наибольший элемент $\top$, если его нет.

$$x \wedge \top = x \tag{5}$$

Дополнительные свойства операций:

$$x \vee x = x$$

$$x \vee y = y \vee x$$

$$(x \vee y) \vee z = x \vee (y \vee x)$$

$$x \vee \top = \top$$

$$x \vee \bot = x$$

$$x \wedge (x \vee y) = x \qquad x \vee (x \wedge y) = x \tag{6}$$

Описать окружность около выпуклого многоугольника

Многоугольники	**goal**	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
Равносторонний треугольник	+	1	0	0	1	1	0	1	1
Прямоугольный треугольник	+	1	0	1	0	0	0	0	0
Квадрат	+	0	1	1	1	1	1	1	1
Прямоугольник	+	0	1	1	1	0	1	1	1
Равнобедренная трапеция	+	0	1	0	1	0	1	1	0
Ромб	-	0	1	0	1	1	1	1	0
Параллелограмм	-	0	1	0	1	0	1	1	0
Прямоугольная трапеция	-	0	1	1	0	0	1	1	0
Равнобедренный треугольник	?	1	0	0	1	0	0	1	0

f_1: треугольник; f_2: четырехугольник; f_3: есть прямой угол; f_4: пара сторон равна; f_5: все стороны равны; f_6: есть пара параллельных сторон; f_7: пара углов равна; f_8: все углы равны.

- Сходство $\{f_1\}$ - любой треугольник можно окружить
- Сходство $\{f_4, f_5, f_7, f_8\}$ - вокруг правильного многоугольника можно
- Сходство $\{f_2, f_3, f_4, f_6, f_7, f_8\}$ - вокруг прямоугольника можно
- Сходство $\{f_2, f_4, f_6, f_7\}$ имеет ромб как контр-пример!

Битовое представление обучающей выборки

Метод Монте-Карло дает 2-х мерную таблицу, где строчки соответствуют решениям (состояние s_j + действие = столбец a_r, в котором стоит 1), столбцы f_j - признакам, описывающим ситуации.

S	a_1	...	a_r	$Q(s,a)$	f_1	...	f_j	f_{j+1}	...	f_n
s_1	1	...	0	0.56	0	...	1	1	...	1
⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
s_i	1	...	0	0.64	**1**	...	1	**1**	...	0
⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
s_m	1	...	0	0.82	**1**	...	0	**1**	...	1
s_{m+1}	1	...	0	-0.31	0	...	1	1	...	0
⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
s_{m+k}	1	...	0	-0.77	1	...	1	0	...	1
⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
s_1	0	...	1	0.90	0	...	1	1	...	1
⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋱	⋮	⋱	⋮
s_m	0	...	1	-0.43	1	...	0	1	...	1

Формальное определение кандидатов

Для подмножества $A \subseteq O$ объектов его *общим фрагментом* называется подмножество $A' = \{f \in F : \forall o \in A\,[oIf]\} \subseteq F$. Полагаем $\emptyset' = F$.
На самом деле, это определение совпадает с последовательным вычислением побитового умножения строк, соответствующих отобранным во множество A объектов.
Для подмножества $B \subseteq F$ признаков его *сходством* называется подмножество $B' = \{o \in O : \forall f \in B\,[oIf]\} \subseteq O$. Полагаем $\emptyset' = O$.
Операции $' : 2^O \to 2^F$ и $' : 2^F \to 2^O$ называются *полярами* и задают *соответствие Галуа*.

Определение

Пару $\langle A, B\rangle$ *назовем* ***кандидатом***, *если* $A = B' \subseteq O$ *и* $B = A' \subseteq F$.
Подмножество $A \subseteq O$ *называем* ***списком родителей*** *кандидата, а* $B \subseteq F$ - ***(общим) фрагментом*** *кандидата.*
Равенство $A = B'$ *соответствует принципу* ***исчерпываемости***.

Плотные подмножества решетки

Подмножество элементов $S \subseteq L$ решетки $\boldsymbol{L}$ называется $\wedge$-**плотным**, если для любого элемента $x \in L$ найдется такое подмножество $X \subseteq S$, что $x = \bigwedge X$.
Подмножество элементов $S \subseteq L$ решетки $\boldsymbol{L}$ называется $\vee$-**плотным**, если для любого элемента $x \in L$ найдется такое подмножество $X \subseteq S$, что $x = \bigvee X$.

Лемма

Для решетки кандидатов $\boldsymbol{L}$*, порождаемой обучающей выборкой* $I \subseteq O \times F$*, образ* $g(O) = \{g(o) : o \in O\}$ *отображения* $g : O \to L$*, задаваемого правилом* $g(o) = \langle \{o\}'', \{o\}' \rangle$*, является* $\wedge$*-плотным подмножеством. Образ* $h(F) = \{h(f) : f \in F\}$ *отображения* $h : F \to L$*, задаваемого правилом* $h(f) = \langle \{f\}', \{f\}'' \rangle$*, является* $\vee$*-плотным подмножеством.*

Лемма

Для любой конечной решетки $\boldsymbol{L}$ *любое надмножество всех* $\vee$*-неразложимых элементов образуем* $\vee$*-плотное подмножество, а любое надмножество всех* $\wedge$*-неразложимых элементов образуем* $\wedge$*-плотное подмножество.*

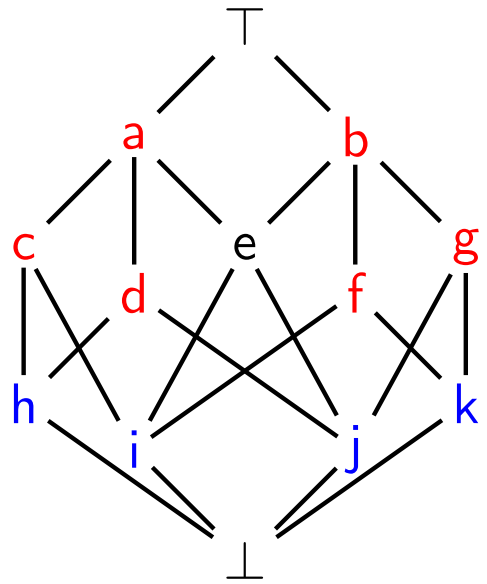
Неразложимые элементы решетки

Определение

Элемент $x \in L$ назовем $\vee$-неразложимым, если $x \neq \bot$ и для любых $y, z \in L$ если $y < x$ и $z < x$, то $y \vee z < x$.

Определение

Элемент $x \in L$ назовем $\wedge$-неразложимым, если $x \neq \top$ и для любых $y, z \in L$ если $x < y$ и $x < z$, то $x < y \wedge z$.



Фундаментальная теорема Р. Вилле

Ganter Bernhard, Wille Rudolf, **Formal Concept Analysis**, Springer, 1999

Теорема

Для произвольной обучающей выборки $I \subseteq O \times F$ множество всех кандидатов образует решетку относительно операций

$$\langle A_1, B_1 \rangle \wedge \langle A_2, B_2 \rangle = \langle (A_1 \cup A_2)'', B_1 \cap B_2 \rangle \tag{7}$$

$$\langle A_1, B_1 \rangle \vee \langle A_2, B_2 \rangle = \langle A_1 \cap A_2, (B_1 \cup B_2)'' \rangle \tag{8}$$

Теорема

Пусть для любой конечной решетки $\langle L, \wedge, \vee \rangle$ найдутся такие два множества O и F с отображениями $g : O \to L$ и $h : F \to L$, что $g(O) \subseteq L$ - $\wedge$-плотное подмножество, а $h(F) \subseteq L$ - $\vee$-плотное подмножество. Тогда, полагая $oIf \Leftrightarrow g(o) \geq h(f)$, мы получим обучающую выборку, решетка кандидатов для которой будет изоморфна исходной решетке $\langle L, \wedge, \vee \rangle$.

Простейший вариант: $g = id : L \to L$ и $h = id : L \to L$.

Булева алгебра: экспоненциальный взрыв

Пусть $O = \{o_1, o_2, \ldots, o_n\}$ - множество объектов, а $F = \{f_1, f_2, \ldots, f_n\}$ - множество признаков, и обучающая выборка задается формулой $o_i I f_j \Leftrightarrow i \neq j$:

$O \mid F$	f_1	f_2	$\ldots$	f_n
o_1	0	1	$\ldots$	1
o_2	1	0	$\ldots$	1
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
o_n	1	1	$\ldots$	0

Ясно, что любая пара $\langle O \setminus \{o_{j_1}, \ldots, o_{j_k}\}, \{f_{j_1}, \ldots, f_{j_k}\}\rangle$ будет кандидатом, поэтому мы имеем Булеву алгебру всех 2^n подмножеств (=битовых строк). При $n = 32$ обучающая выборка занимает $n \cdot n = 2^{10}$ бит $= 128$ байт. Но чтобы записать результат требуется $n \cdot 2^n = 2^{37}$ бит, т.е. ровно 16 Гигабайт. При $n = 64$ обучающая выборка занимает $n^2 = 2^{12}$ бит $= 512$ байт, но для сохранения всех кандидатов нужно $n \cdot 2^n = 2^{70}$ бит!

Подрешетка, изоморфная Булевой алгебре

Рассмотрим серию испытаний Бернулли $\delta_{1,1}, \ldots, \delta_{m,n}$ с вероятностью успеха $p = \left(1 - \frac{1}{m}\right)$, заполняющую обучающую выборку $I \subseteq O \times F$, где $O = \{o_1, \ldots, o_m\}$ и $F = \{f_1, \ldots, f_n\}$.
Обозначим через c_i столбец высоты m, где (m-1) позиция заполнены единицами, а единственный ноль встречается на i-ой позиции. Значение для параметра p определяется как оценка максимального правдоподобия, чтобы обеспечить в среднем ровно один ноль в столбце высоты m. Тогда вероятность $\mathbb{P}[f_j = c_i]$ получить столбец c_i равна

$$s = p^{m-1} \cdot (1 - p) = \left(1 - \frac{1}{m}\right)^{m-1} \cdot \frac{1}{m}$$

для любых i и j, где события $[f_j = c_i]$ и $[f_k = c_l]$ независимы для i, k, j и l.

Теорема

При $m \to \infty$ и $n \geq e \cdot m \cdot \ln m$ для вероятности порождения m-мерной булевой алгебры как решётки кандидатов имеем

$$\lim_{m \to \infty} \mathbb{P}\left[\forall j \exists i \left(f_j = c_i\right)\right] \geq 1 - m^{(1 - e \cdot m \cdot s)} \to 1.$$

Переобучение: фантомные сходства

Пусть $O = \{o_1 = B737MAX, o_2 = MC21, o_3 = SJ100, o_4 = A350\}$ будет множеством самолетов без сертификата летной годности, каждый из которых описывается проблемами из списка $F = \{f_1 = \text{оперение}, f_2 = \text{двигатель}, f_3 = \text{нацарапанное ругательство}\}$:

$O \mid F$	f_1	f_2	f_3
o_1	1	0	0
o_2	1	0	1
o_3	0	1	1
o_4	0	1	0

Если рассмотреть непустые сходства не менее двух объектов, то мы получим две «настоящие» причины: $\langle \{o_1, o_2\}, \{f_1\} \rangle$ «самолет с проблемным оперением нельзя допускать» и $\langle \{o_3, o_4\}, \{f_2\} \rangle$ «самолет с проблемным двигателем нельзя допускать», и одно «фантомное» сходство $\langle \{o_2, o_3\}, \{f_3\} \rangle$ «самолет с ругательством нельзя допускать». Последний кандидат возник из-за случайного совпадения подмножества признаков $\{f_3\}$ у двух примеров o_2 и o_3, каждый из которых имеет свою отличную от других «настоящую» причину.

Фантомные сходства неустранимы

Теорема

Для $p \geq (-\ln(1-\varepsilon)/n)^{1/b}$ вероятность появления фантомного сходства b случайных p-примеров не меньше, чем $\varepsilon > 0$.

Пусть число n обозначает количество сопутствующих признаков, которыми мы ограничиваемся. Для каждого контр-примера или обучающего примера образуем последовательность n испытаний Бернулли с одинаковой вероятностью успеха p, причём последовательности для разных объектов независимы. Число m будет равно числу контр-примеров.

Теорема

При числе сопутствующих признаков $n \to \infty$ и вероятности появления этих признаков у контр-примеров и обучающих примеров, равной $p = \sqrt{\frac{a}{n}}$ $(a \leq 1)$, вероятность возникновения фантомного сходства двух обучающих примеров, не устранённого никаким из $m = c \cdot \sqrt{n}$ контр-примеров, будет стремиться к

$$1 - e^{-a} - a \cdot e^{-a} \cdot \left[1 - e^{-c \cdot \sqrt{a}}\right] > 0.$$

Переобучение неустранимо: эксперименты

Iakimova, L.A. **The Study of Retraining in Algebraic Machine Learning** // Pattern Recognition and Image Analysis, Vol. 33 (2023), No. 3, p. 350–353
Были описаны результаты экспериментального исследования вопроса о подозрительных на «фантомность» гипотез:

- Массив взят из Репозитория данных для тестирования алгоритмов машинного обучения
  https://archive.ics.uci.edu/dataset/73/mushroom
- Оцифрованная книга: *Lincoff, G.H.* **The Audubon Society Field Guide to North American Mushrooms**. – NY: Knopf, 1981. – 926 pp.
- Содержит описания 8124 грибов двух видов (съедобные и ядовитые).
- Содержит описания 4208 съедобных и 3916 ядовитых грибов.
- Каждый пример описывался 22 признаками, описывающие различные характеристики грибов. Эти признаки - номинальные, принимающие одно из нескольких значений.

При случайном разделении массива на обучающую и тестовую выборки (с вероятностью 1/2) количество поганок, предсказанных как съедобные грибы, составило от 7 до 22. В некоторых экспериментах доля ошибок превысила 1%.

Операции «Замыкай-по-одному»

Идея: порождать только случайную подвыборку кандидатов с помощью случайных блужданий по решетке кандидатов.
Операция **замыкай-по-одному-вниз** на кандидате $\langle A, B\rangle$ и объекте $o \in O$ порождает кандидат

$$CbODown(\langle A, B\rangle, o) = \langle A, B\rangle \wedge \langle \{o\}'', \{o\}'\rangle = \langle (A \cup \{o\})'', B \cap \{o\}'\rangle.$$

Операция **замыкай-по-одному-вверх** на кандидате $\langle A, B\rangle$ и признаке $f \in F$ порождает кандидат

$$CbOUp(\langle A, B\rangle, f) = \langle A, B\rangle \vee \langle \{f\}', \{f\}''\rangle = \langle A \cap \{f\}', (B \cup \{f\})''\rangle.$$

Ускорение вычислений:
Если $o \in A$, то $CbODown(\langle A, B\rangle, o) = \langle A, B\rangle$. Аналогично, если $f \in B$, то $CbOUp(\langle A, B\rangle, f) = \langle A, B\rangle$.

Спаривающая цепь Маркова

Состоянием изменяемых переменных в цикле (= состоянием цепи Маркова) является упорядоченная пара кандидатов $\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle$.

Определение

Порядок *на кандидатах:* $\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle$, *если* $B_1 \subseteq B_2$.

Первоначально меньший кандидат совпадает с наименьшим кандидатом $Min := \langle O, O'\rangle$, а больший - с наибольшим $Max := \langle F', F\rangle$.
В цикле к обоим кандидатам применяется одна и та же операция *CbODown* с выбранным объектом, или *CbOUp* с выбранным признаком.

Лемма

Для всякой упорядоченной пары кандидатов $\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle$ *и любого* $o \in O$ *имеем* $CbODown(\langle A_1, B_1\rangle, o) \leq CbODown(\langle A_2, B_2\rangle, o)$.
Для всякой упорядоченной пары кандидатов $\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle$ *и любого* $f \in F$ *имеем* $CbOUp(\langle A_1, B_1\rangle, f) \leq CbOUp(\langle A_2, B_2\rangle, f)$.

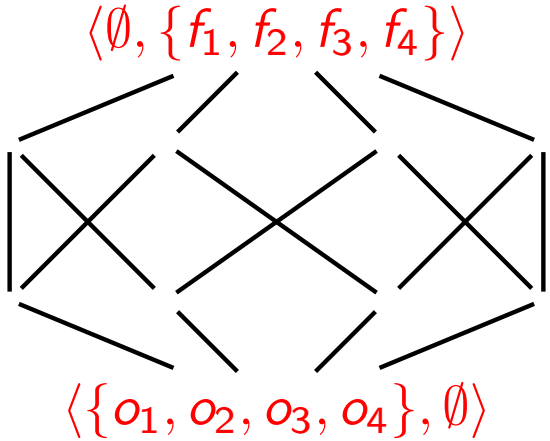
Процесс останавливается, когда меньший кандидат совпадет в большим. Тогда этот общий кандидат и выдается алгоритмом 1.

Алгоритм спаривающей цепи Маркова

Алгоритм

Data: *множество обучающих (+)-примеров; внешние функции* $CbOUp(\ ,\ )$ *и* $CbODown(\ ,\ )$ *операций «замыкай-по-одному»*
Result: *кандидат* $\langle A, B\rangle$
$O :=$ *(+)-примеры,* $F :=$ *признаки;* $I \subseteq O \times F$ *- обучающая выборка из (+)-примеров* $R := O \cup F$; $Min := \langle O, O'\rangle$; $Max := \langle F', F\rangle$
while $(Min \neq Max)$ **do**
  Выбираем случайный элемент $r \in R$
  if $(r \in O)$ **then**
    $Min := CbODown(Min, r)$; $Max := CbODown(Max, r)$
  end
  else
    $Min := CbOUp(Min, r)$; $Max := CbOUp(Max, r)$
  end
end
$\langle A, B\rangle := Min$

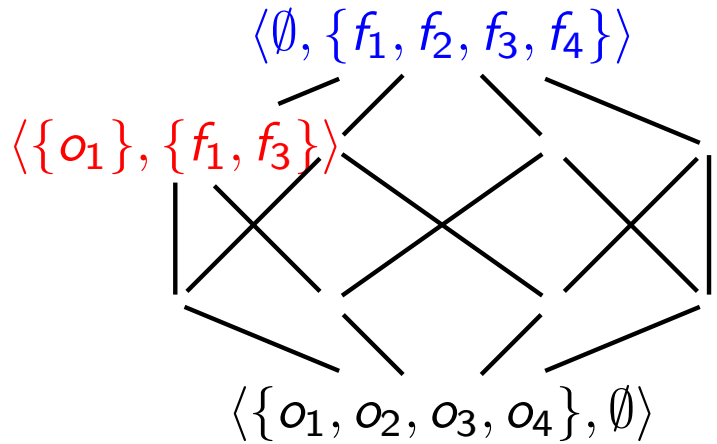
Как работает спаривающая цепь Маркова: шаг 0



Пример

верхний	f_1	f_2	f_3	f_4	нижний	f_1	f_2	f_3	f_4
o_1	1	0	1	0	o_1	***1***	***0***	***1***	***0***
o_2	1	0	0	1	o_2	***1***	***0***	***0***	***1***
o_3	0	1	1	0	o_3	***0***	***1***	***1***	***0***
o_4	0	1	0	1	o_4	***0***	***1***	***0***	***1***

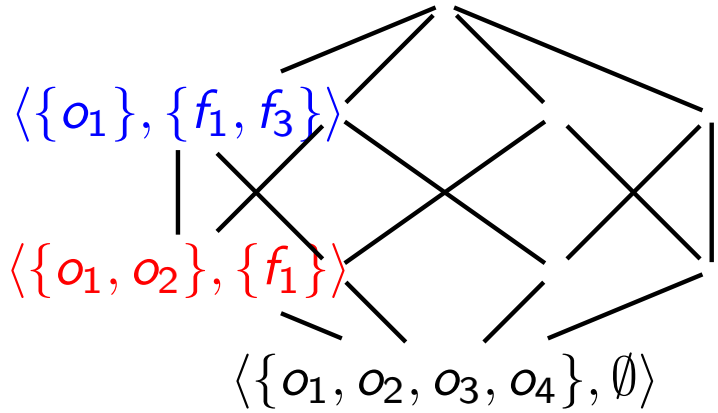
Как работает спаривающая цепь Маркова: выбор o_1



Пример

верхний	f_1	f_2	f_3	f_4	нижний	f_1	f_2	f_3	f_4
o_1	***1***	***0***	***1***	***0***	o_1	***1***	***0***	***1***	***0***
o_2	1	0	0	1	o_2	***1***	***0***	***0***	***1***
o_3	0	1	1	0	o_3	***0***	***1***	***1***	***0***
o_4	0	1	0	1	o_4	***0***	***1***	***0***	***1***

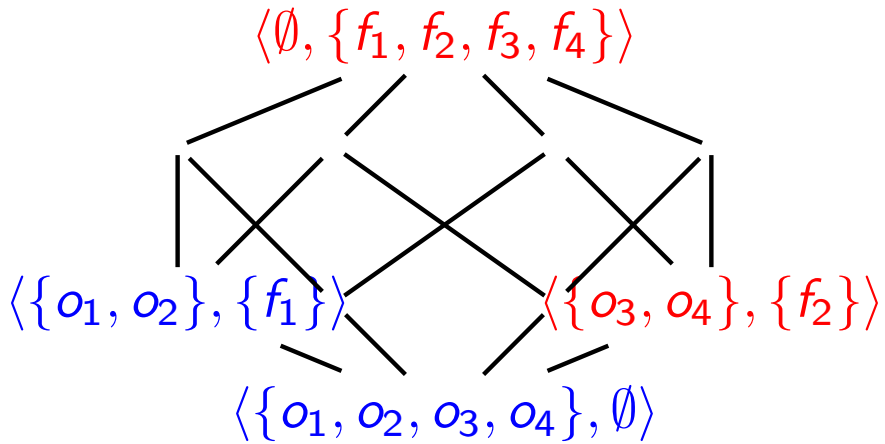
Как работает спаривающая цепь Маркова: выбор o_2



Пример

верхний	f_1	f_2	f_3	f_4	нижний	f_1	f_2	f_3	f_4
o_1	***1***	***0***	***1***	***0***	o_1	***1***	***0***	***1***	***0***
o_2	***1***	***0***	***0***	***1***	o_2	***1***	***0***	***0***	***1***
o_3	0	1	1	0	o_3	***0***	***1***	***1***	***0***
o_4	0	1	0	1	o_4	***0***	***1***	***0***	***1***

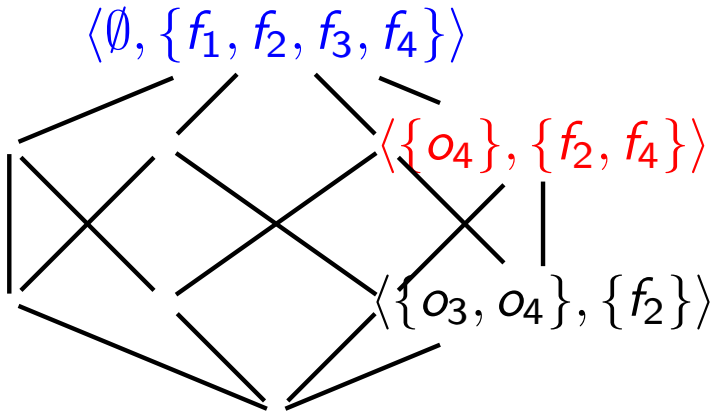
Как работает спаривающая цепь Маркова: выбор f_2



Пример

верхний	f_1	f_2	f_3	f_4	нижний	f_1	f_2	f_3	f_4
o_1	1	0	1	0	o_1	1	0	1	0
o_2	1	0	0	1	o_2	1	0	0	1
o_3	0	1	1	0	o_3	***0***	***1***	***1***	***0***
o_4	0	1	0	1	o_4	***0***	***1***	***0***	***1***

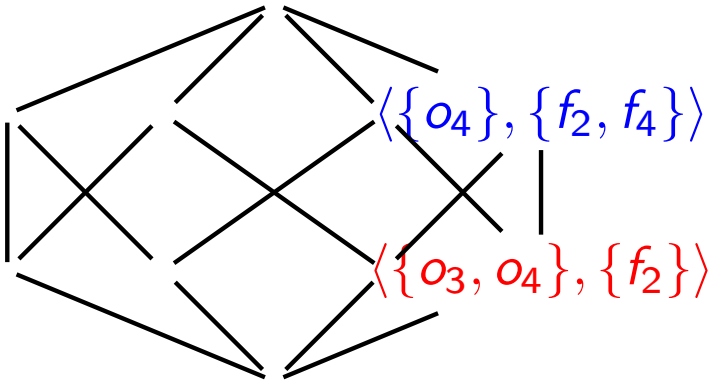
Как работает спаривающая цепь Маркова: выбор o_4



Пример

верхний	f_1	f_2	f_3	f_4	нижний	f_1	f_2	f_3	f_4
o_1	1	0	1	0	o_1	1	0	1	0
o_2	1	0	0	1	o_2	1	0	0	1
o_3	0	1	1	0	o_3	***0***	***1***	***1***	***0***
o_4	***0***	***1***	***0***	***1***	o_4	***0***	***1***	***0***	***1***

Как работает спаривающая цепь Маркова: выбор o_3



Пример

верхний	f_1	f_2	f_3	f_4	нижний	f_1	f_2	f_3	f_4
o_1	1	0	1	0	o_1	1	0	1	0
o_2	1	0	0	1	o_2	1	0	0	1
o_3	***0***	***1***	***1***	***0***	o_3	***0***	***1***	***1***	***0***
o_4	***0***	***1***	***0***	***1***	o_4	***0***	***1***	***0***	***1***

Остановка спаривающей цепи Маркова

Определение

Состояние вида $\langle A_1, B_1 \rangle = \langle A_2, B_2 \rangle$ *спаривающей цепи Маркова для совпадающей пары кандидатов называется* ***эргодическим****. Состояние вида* $\langle A_1, B_1 \rangle < \langle A_2, B_2 \rangle$ *называется* ***невозвратным****.*

Теорема

Вероятность того, что состояние цепи Маркова в момент времени t *окажется невозвратным, стремится к нулю, когда* $t \to \infty$*.* □

Следствие

Алгоритм спаривающей цепи Маркова останавливается с вероятностью 1.

Алгоритм индуктивного обобщения

Алгоритм

Data: *множество обучающих (+)- и (-)-примеров; число N ВКФ-гипотез*
Result: *выборка S ВКФ-гипотез*
$O :=$ *(+)-примеры,* $F :=$ *признаки;* $I \subseteq O \times F$ *- обучающая выборка для (+)-примеров;* $C :=$ *(-)-примеры;* $S := \emptyset$; $i := 0$
while $(i < N)$ **do**
  породить кандидата $\langle A, B \rangle$ *с помощью цепи Маркова*
  hasObstacle := ***false***
  for $(c \in C)$ **do**
    if $(B \subseteq \{c\}')$ **then**
      hasObstacle := ***true***
    end
  end
  if (*hasObstacle* = ***false***) **then**
    $S := S \cup \{\langle A, B \rangle\}$
    $i := i + 1$
  end
end

Алгоритм предсказания по аналогии

Определение

Объект o, описываемый фрагментом $o' \subseteq F$ (множеством признаков), ***предсказывается положительным*** *с помощью ВКФ-гипотезы $\langle A, B \rangle$, если $B \subseteq \{o\}'$.*

Алгоритм

Data: *расширенная выборка S^+ ВКФ-гипотез, файл тестовых (?)-примеров*
Result: *предсказанное целевое свойство у каждого из (?)-примеров*
$X :=$ *(?)-примеры*
for $(o \in X)$ **do**
  $PredictPositively(o) :=$ ***false***
  for $(\langle A, B \rangle \in S^+)$ **do**
    if $(B \subseteq \{o\}')$ **then**
      $PredictPositively(o) :=$ ***true***
    end
  end
end

Демонстрация: массив Mushroom

https://archive.ics.uci.edu/dataset/73/mushroom

- Исходные данные включают описания 8124 грибов, разделенные на две категории (съедобные и ядовитые). Мы случайным образом разделили их на обучающую и тестовую выборки.
- Обучающая выборка содержит 4032 объекта, из которых 2088 (+)-объектов (съедобные грибы).
- Тестовая выборка содержит 2120 (+)- и 1972 (-)-примеров (ядовитые грибы).
- Каждый пример описывался 22 признаками. ВКФ-система закодировала эти признаки битовыми строками длины 110 бит.
- Точность предсказания ВКФ-системы достигла 100% для 50/100 ВКФ-гипотез (для причин съедобности/ядовитости, соответственно).

Необходимое число ВКФ-гипотез

Определение

*Объект o назовем ε-**важным**, если суммарная вероятность появления таких ВКФ-гипотез $\langle A, B\rangle$, что $B \subseteq \{o\}'$ будет больше ε.*

Теорема

Для n признаков и любых $\varepsilon > 0$ и $1 > \delta > 0$ достаточно породить

$$N \geq \frac{n \cdot \ln 2 - \ln \delta}{\varepsilon}$$

ВКФ-гипотез, чтобы вероятностью $> 1 - \delta$ все ε-важные объекты могли быть предсказаны положительно.

Для массива Mushroom из UCI ML Repository результаты следующие:

1. Из выборки, содержащей 2088 съедобных и 1944 ядовитых грибов, порождается более 139 000 различных кандидатов без контр-примеров.
2. Для кодирования $n = 110$ бинарными атрибутами достаточно $N = 50$ ВКФ-кандидатов без контр-примеров, чтобы с доверием 0.95 безошибочно классифицировать все тестовые примеры.

Минимизация эмпирического риска

Так как априорная ВПК-оценка для N сильно завышена , предлагается попеременно запускать алгоритм индуктивного порождения гипотез (удваивая каждый раз их число) и следующего алгоритма

Data: расширенная выборка S^+ гипотез, файл $(+)$-примеров
Result: значение k эмпирического риска
$O := (+)$-примеры;
$k := 1$; $d = |O|$;
for $(o \in O)$ **do**
  for $(\langle A, B\rangle \in S^+)$ **do**
    if $(B \subseteq \{o\}')$ **then**
      $k = k - \frac{1}{d}$;
      break;
    end
  end
end

до тех пор, пока эмпирический риск не прекратит уменьшаться. Это - вариант метода минимизации эмпирического риска В.Н. Вапника - А.Я. Червоненкиса.

Исходный пример: «Крестики-нолики»

По-американски она называется **"tic-tac-toe"**, в британском содружестве - **"noughts and crosses"**.
Доски для игры типа три-в-ряд обнаруживаются в раскопках античного Египта (датируются около 1300 д.н.э.), где они встречаются на крышах домов (они плоские, там хозяева принимали гостей).
Комбинаторика и симметрии
Хотя число состояний в игре 5477 (не могут одновременно оба игрока выиграть), симметрии (т.е. вращения и отражения) приводят к существенному сокращению: существует только 138 окончательных позиций. Комбинаторные исследования показывают, что (когда "X"делает первый ход) результаты таковы:

- 91 позиций, в которых выиграл (X);
- 44 позиций, в которых выиграл (O);
- 3 ничейных позиций (часто называемых "cat's game")

Мы будем (для наглядности) игнорировать симметрии.

Совершенная стратегия для первого игрока

В 1972 году Ньюэлл и Саймон выписали упорядоченный список правил для tic-tac-toe.

1. Если игрок имеет две клетки в ряд, а третья свободна, то он должен занять эту клетку.
2. Если оппонент имеет две клетки в ряд, то игрок должен занять третью (свободную) клетку, чтобы блокировать непосредственный выигрыш оппонента.
3. Занять такую клетку, когда у игрока будет 2 способа выиграть (две незаблокированных линий из 2 клеток).
4. Если у оппонента есть единственный способ разветвиться, то игрок должен блокировать его. Иначе, игрок должен блокировать то разветвление, которое одновременно строит незаблокированную линию из 2 его клеток.
5. Игрок занимает свободную центральную клетку.
6. Если оппонент занял угол, то игрок занимает свободный противоположный угол.
7. Игрок занимает свободный угол.
8. Игрок занимает свободную середину стороны..

Математика vs психология

В 1975 году А. Ньюэлл и Г. Саймон получили премию Тьюринга за создание системы General Problem Solver.
Впоследствии Герберт Саймон применил GPS для исследования поведения нерациональных экономических субъектов, за что его в 1978 году наградили Нобелевской премией по экономике.
В частности, он обнаружил, что:
Первый игрок ("X") имеет 3 стратегически различных позиции для первого хода (из-за симметрий): угол, центр, или середину ребра.
Игрок может выиграть или не проиграть, начиная с любого стартового хода; однако, выбор угла дает оппоненту наименьшее число ходов, которые тот должен сделать, чтобы избежать поражения.
Можно было бы предполагать, что выбор угла - лучший старт для X, однако исследования Г. Саймона показали, что нерациональные игроки O чаще проигрывают при занятии центра на первом шаге для X.

Современные подходы

Рассматривают игру tic-tac-toe как задачу обучения с покреплением (Reinforcement Learning).

1. Используют метод Монте-Карло на частичных деревьях игры, чтобы оценить $Q - values$ для выбора оптимальной стратегии.
2. Используют аппроксимацию $Q - values$ с помощью нейросети.

Сейчас более популярен нейросетевой подход, после успеха AlphaGo в 2015 году, результат которой обеспечил переключение от первого варианта на второй.
Мы будем моделировать первый подход (через оценивание методом Монте-Карло), дополнив его вычислением сходств между состояниями, в которых применялось одно и то же действие.
Доводы за наше решение:

1. Есть вопросы об адекватности аппроксимации $Q - values$ нейросетями.
2. Статистика метода Монте-Карло легко собирается (и порождает доверительные интервалы).
3. Есть ВКФ-метод обучения, основанный на операции сходства.

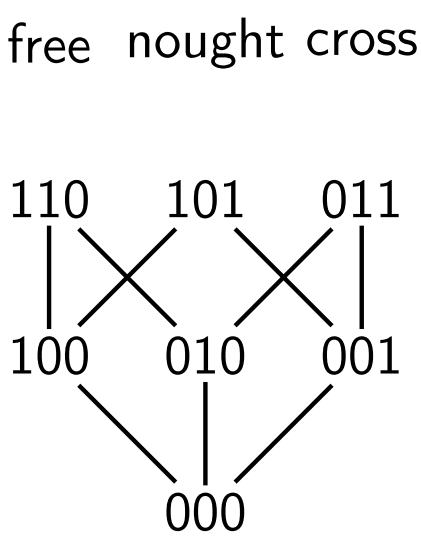
Кодирование состояний

Обозначим состояния ячеек через:

- 011 - X - Cross;
- 101 - O - Nought;
- 110 - - Free.

Введём промежуточные состояния (для сходства их состояний):

- 001 - ? - Occupied;
- 010 - % - non-Nought;
- 100 - # - non-Cross;
- 000 - * - Any.



Одно из правил Ньюэлла и Саймона

Rule: If the player has two in a row, he/she can place a third to get three in a row.
Action: The first player can occupy the bottom right angle to immediatedly win in situations covered by generalized state.

011	000	000		X	*	*
000	011	000		*	X	*
000	000	110		*	*	

При конкатенации битовых строк для состояний клеток слева-направо сверху-вниз имеем
011000000000011000000000110

Правила как сходства конкретных действий

Допустим, что действие "занять правый нижний угол"применялось при трёх разных состояниях доски.
Находя их сходство, получим правило "Занимай правый нижний угол, если он свободен, на диагонали стоят два крестика, в позициях (0,1), (0,2) и (1,0) нет крестиков, а в остальных позициях может стоять что угодно!"

X		O		X	O			X	#	#
O	X		$\wedge$	O	X	X	=	O	X	%
X	O			O				?	#	
X	#	#		X	O	O		X	#	#
O	X	%	$\wedge$		X	O	=	#	X	*
?	#				X			*	*	

Кодирование последнего сходства имеет представление
011 100 100 100 011 000 000 000 110,
что отличается от соответствующего варианта
011 000 000 000 011 000 000 000 110
правила Ньюэлла и Саймона только в 3 позициях!

Игра HoMM3: постановка задачи

Применить описанную систему к поиску стратегии в игре Heroes of Might and Magic 3

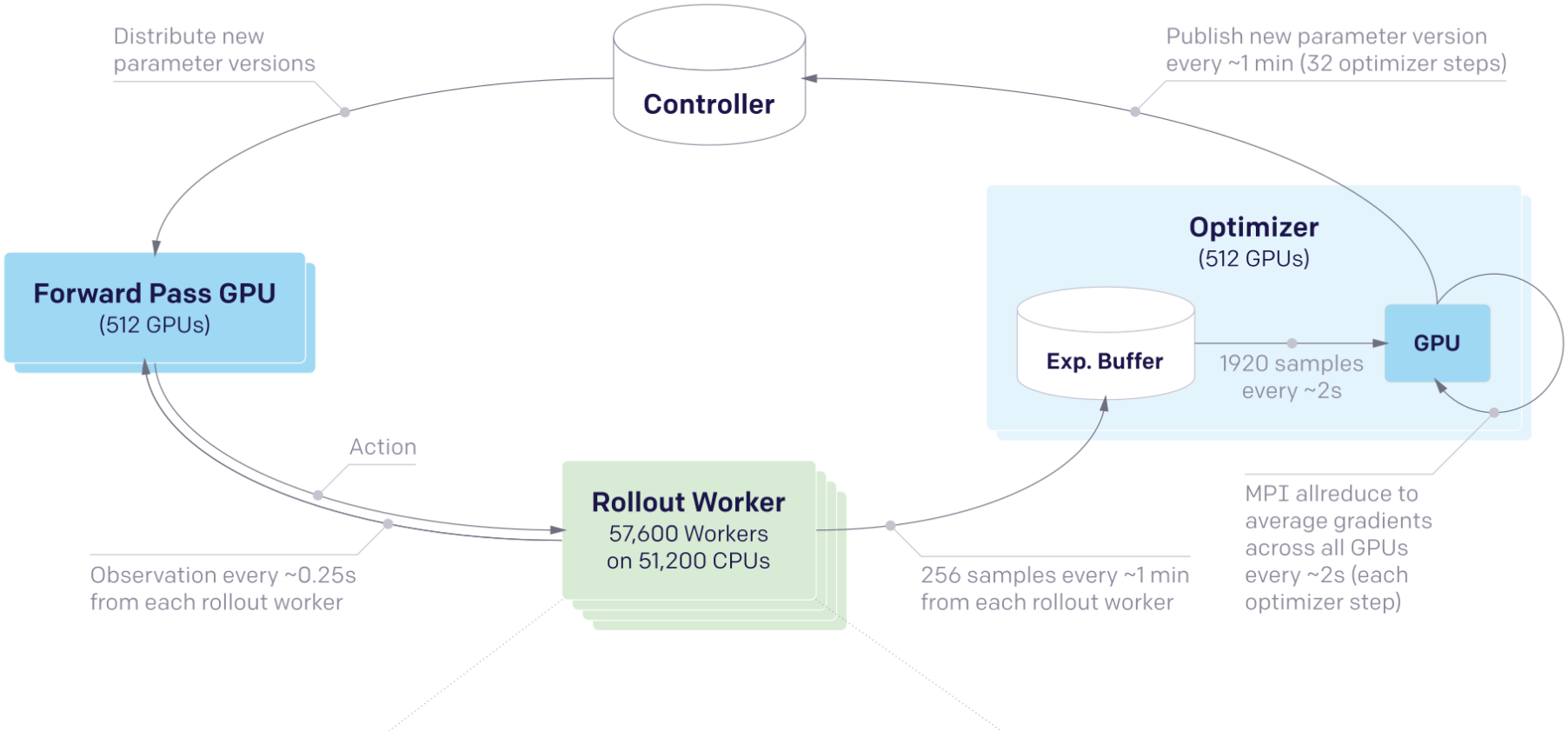
1. Имеются исходные тексты на C++;
2. Имеются отдельные стратегии (в том числе, фанатские) для
   - разведка: движение по карте и захват внешних ресурсов;
   - финансирование: строительство зданий в городе и найм героев и войск;
   - бои: штурм городов и бои на местности разных типов;
3. Горизонт планирования компьютера только один игровой день, а войска и герои поступают раз в неделю.
4. Проводятся турниры для HotA (фанатское дополнение к HoMM3).

Это может привести к обобщению теории обучения с подкреплением для игр с частичной наблюдаемостью (туман войны), но следующее доставляет достаточную информацию:

1. заклинание "обзор воздуха": наличие городов, героев и войск;
2. заклинание "обзор земли": наличие сокровищ и шахт;
3. информация из гильдии воров: противники, их ресурсы, лучшие войска

А что на Западе?

OpenAI использовала следующую конфигурация вычислительного оборудование для обучения стратегии игры Dota 2 (декабрь 2019)



Взято из: OpenAI et al. Dota 2 with Large Scale Deep Reinforcement Learning // https://arxiv.org/abs/1912.06680