# **REINFORCEMENT LEARNING LIBRARY**

## Этапы обучения с подкреплением

Реализация обучения с подкреплением выполняется в несколько этапов:

1. инициализация система, которую надо обучить
2. применяется алгоритм обучения с подкреплением
3. анализ полученных результатов
4. оптимизация алгоритма обучения с подкреплением (при необходимости)

Данные этапы выполняются не зависимо от того, какой алгоритм обучения с подкреплением будет применён. Но при этом способ реализации каждого из этапов может быть выбран индивидуально, в зависимости от условий и сложности решаемой задачи.

В библиотеке предлагается реализация этапов 1 и 2, написанная на языке программирования Java. Реализация этапа 1 представляет собой набор объектов и методов, который позволяет инициализировать обучаемую систему. На этапе 2 есть возможность выбора алгоритмов обучения SARSA или Q-Learning. При этом, доступны методы, реализующие основные шаги указанных алгоритмов. Таким образом пользователь получает возможность самостоятельно модифицировать алгоритм обучения в пределах TD-обучения. Это даёт возможность повышать качество обучения, а также не зацикливаться на самостоятельной реализации шагов, которые будут однотипными для алгоритмов TD-обучения. Также некоторые методы пользователь может переопределять. Это предаёт гибкость коду и удобство использования, так как чётко установленных правил инициализации системы обучения или реализации алгоритмов обучения нету.

Этапы 3,4 должны быть реализованы непосредственно пользователем путём анализа полученных в результате этапа 2 данных, а также изменений параметров обучения в целях улучшения качества обучения. Результатами для анализа могут быть, к примеру, скорость обучения, эпизод, на котором был достигнут успех в обучении и другие параметры, которые пользователь выберет или определит самостоятельно.

Стоит отметить, что этапы 1 и 2 зависят от поставленной задачи обучения с подкреплением. Поэтому пользователь должен чётко определить цель обучения, а также что конкретно и чему надо обучить. Только после этого можно переходить к этапу 1. Таким образом можно выделить начальный, или подготовительный, этап для пользователя – постановка задачи обучения. Без корректной постановки задачи переход к реализации обучения с подкреплением является бессмысленным.

Далее будет описана более детально непосредственно структура этапов 1, 2 и её реализация в RL библиотеке. Также будут приведены некоторые примеры реализации этапов 3,4.

## 1 Этап инициализации обучаемой системы

### **Структура этапа инициализации обучаемой системы**

Этап инициализации обучаемой системы следует непосредственно после постановки задачи. Его целью является формулировка постановленной задачи в терминах обучения с подкреплением. Это означает, что даётся чёткое определение:

1. агента - того, кого надо обучить;
2. множества состояний – состояния, в которых потенциально может пребывать агент при заданной внешней среде;
3. множества действий – действия, которые агент потенциально может выполнить, находясь в конкретном состоянии и взаимодействии со внешней средой;

Другие параметры (на пример, вознаграждение, функция ценности) определяются на этапе 2 реализации задач обучения с подкреплением и зависят от выбранного алгоритма.

Таким образом до начала применения алгоритма обучения надо провести инициализацию системы, то есть этап определения обучаемой системы, состоящего из следующих шагов соответственно:

1. инициализация агента
2. инициализация множества состояний
3. инициализация множества действий

Результатом данного этапа будет чёткое определение того, что мы хотим обучить и чему.

### **Реализация этапа инициализации обучаемой системы в RL-библиотеке**

4) выбрать алгоритм, в соответствии с которым будет проводится обучение

5) определить все необходимые параметры в зависимости от выбранного алгоритма

6) описать процесс обучения

7) проанализировать полученные результаты

2) выбирается алгоритм для реализации обучения с подкреплением

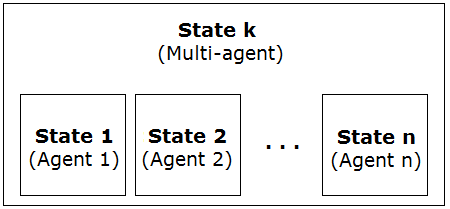
3) определяются параметры обучения для применения выбранного алгоритма обучения

4) применение выбранного алгоритма с определёнными ранее параметрами

Agent:

State:

Состояние мультиагента в конкретный момент времени состоит из состояний каждого отдельного агента в этот момент времени. Множество состояний состоит из множеств состояний каждого входящего в мультиагента агента. А это значит, что множество состояний мультиагента – это все возможные переборы состояний, в которых находятся агенты в конкретный момент времени. При этом количество возможных состояний мультиагента равно количеству всех комбинаций возможных состояний каждого из агентов.

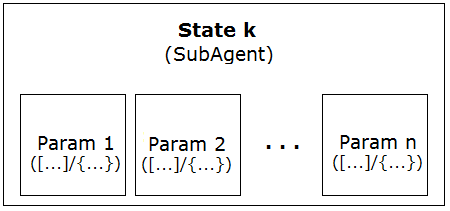


В свою очередь состояние агента в конкретный момент времени описывается одним или несколькими разными параметрами. Каждый из этих параметров имеет свою область значений. Они могут задаваться либо как интервальная разбивка с указанием шага между значениями, либо просто как набор значений.

Рассмотрим, к примеру, задачу балансировки стержня на двигающейся тележке. Агент – система «тележка-стержень». Действия, составляющие множество состояний, представляют собой сдвиг влево, сдвиг вправо, остановка на месте. Состояние такой системы описывается с помощью четырёх параметров: позиция тележки на ограниченной платформе, угол отклонения стержня от вертикальной оси, ускорением тележки и круговым ускорением стержня. Соответственно множество состояний – это перебор всех комбинаций возможных значений каждого из этих параметров. В процессе обучения к агенту прикладываются внешние возмущения, на которые он реагирует, выполняя действия и, таким образом, изменяя своё состояние.

Возьмём первый параметр, то есть позицию тележки. Допустим, платформа представляет собой горизонтальную ось, длинной 20 делений ([-10,10]). Вначале обучения система «тележка-стержень», которая является агентом в данной задаче, находится в центре платформы, то есть на нулевой позиции. Как только внешняя среда приложит своё воздействие к тележке, последняя может сдвинутся на определённую позицию вправо, влево или не изменить своего положения. Это означает, что система будет пребывать около одного из делений. Поэтому считается, что позиция тележки – это ближайшее значение ближайшего деления к текущей реальной позиции тележки. Так как важность данного параметра проявляется в том, что тележка не должна выехать за пределы платформы, то в более детальной разбивке нету потребности. Исходя из этого множество значений данного параметра можно задать либо как разбивку по интервалу [-10,10] с шагом в 1 деление, либо как множество {-10,-9,-8, … , 0, … , 8,9,10}. В данном случае второй вариант применить можно только потому, что значений не много и перечислить их не сложно. Если же рассматривать второй параметр состояний, то есть угол отклонения стержня от оси, то от точности определения значений угла в конкретный момент времени зависит качество и скорость обучения (без учёта влияния на данные качественные оценки выбранного алгоритма и других параметров обучения). Поэтому надо брать как можно более детальную (в разумных пределах) разбивку угла на интервале, ограниченном граничными условиями. В этом случае вариант задания параметра состояния в виде просто перечисления является неудобным и сложным в представлении.

Понятное дело, что в программе в обоих случаях элементы будут представляться в виде перечисления, но в случае разбивки с шагом по интервалу в определение точных значений параметра пользователем нету необходимости, а это очень удобно в случае с большим количеством значений с постоянным шагом. Если изменения значений нельзя описать шагом, то надо использовать второй случай описания, то есть перечисление на прямую.



***Действия пользователя по написанию программы с использование Обучения с Подкреплением***

1) *Определяет что конкретно собирается обучать, указывая только имя мульти-агента*. При этом он не определяет параметры, необходимые для обучения: множество состояний и множество действий.

Основные инструменты для определения мульти-агента:

- класс, с помощью которого реализуется инициализация агента: /\*…\*/

- конструктор класса, который отвечает за инициализацию агента со входящим параметром «имя мульти-агента»: /\*…\*/

- методы для дальнейшего определения параметров обучения для данного мульти-агента: /\*…\*/

2)