Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play

Desheulin Oleg

Higher School of Economics

https://arxiv.org/pdf/1703.05407.pdf

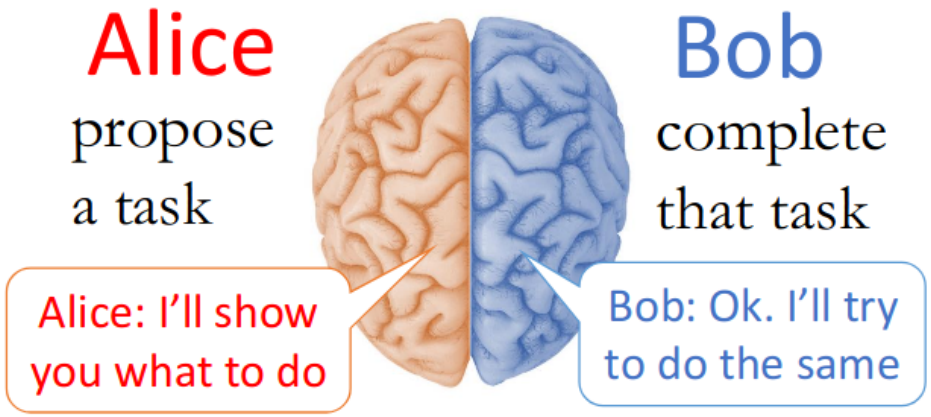
8 февраля 2019 г.

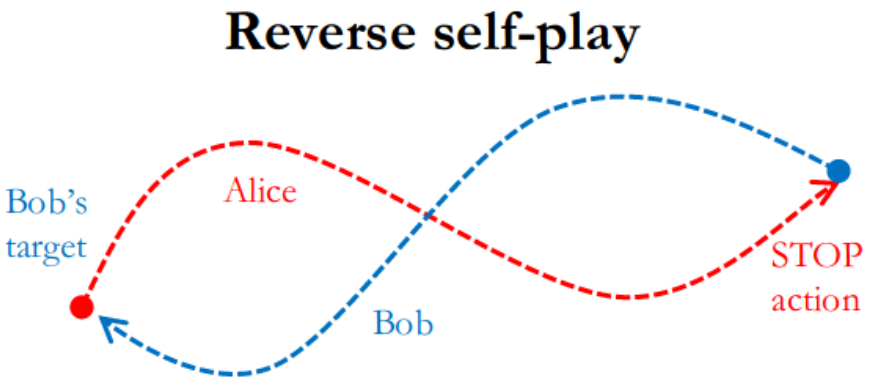
Проблемы которые попробуем решить

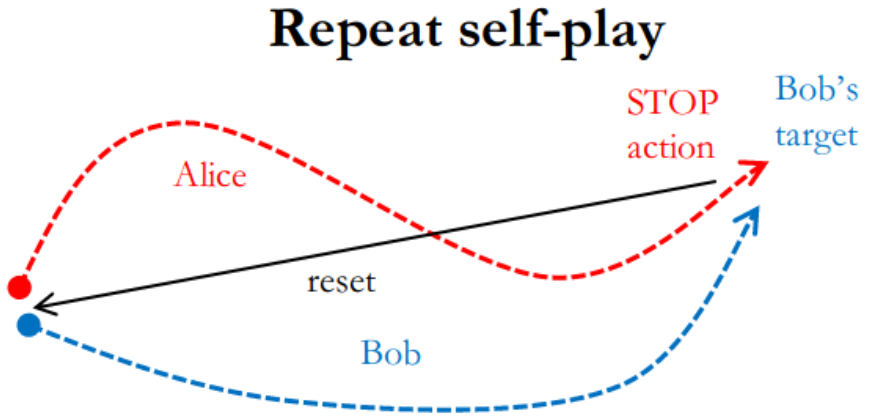
- Обучение агента происходит с нуля, поэтому может потребоваться много итераций
- Если эпизод длится долго - сложно правильно наградить агента, вообще сложно в принципе подобрать хорошую систему наград
- Надо соблюсти баланс между Exploration и Exploitation
- Как выбрать правильный Curricula - последовательность заданий для лучшего обучения агента

Идея

Создадим иллюзию конкуренции и заставим агента играть с самим собой







Reward

Пусть t_A - время, затраченное Алисой до момента STOP, а t_B - время затраченное Бобом на выполнение задания от Алисы. Если Боб не справился, то будем считать $t_B = t_{MAX} - t_A$.

- Alice reward:

$$R_A = \gamma max(t_B - t_A)$$

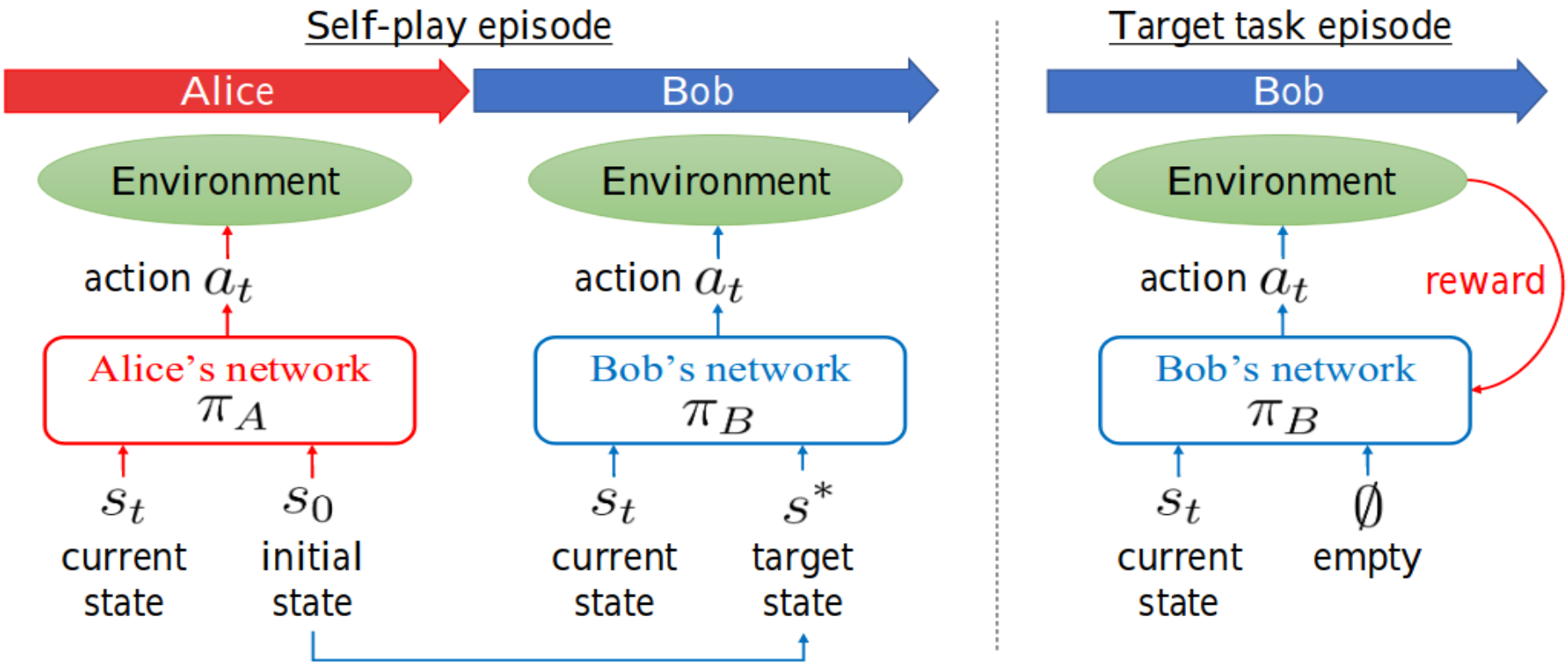
  Алиса пытается найти такое задание на котором Боб фейлится.

- Bob's reward:

$$R_B = -\gamma t_B$$

  Боб пытается максимально быстро выполнить задание.

Параметризация policy



Теоретическое обоснование

Такой способ обучения может быть интерпретирован, как обучение Боба находить кратчайший путь из любого состояния в среды в любое.

Theorem (Optimal Bob)

Если π_A и π_B находятся в балансе (Боб не может быть улучшен без изменений в Алисе и наоборот), то π_B - π_{fast}: позволяет за наиболее котороткое время добраться из любого s_t в любой s_*.

- В балансе Алиса получает нулевой reward: $R_A = 0$
- Если Боб неоптимален в задании (s_t, s_*) и Алиса его предлагает Бобу с $p \neq 0$ - Боба можно улучшить.
- Если же Алиса никогда не предлагает (s_t, s_*), то она может предложить это задание с такой же последовательностью действий, как в π_{fast} и получить тогда $R_A > 0$.

Эксперименты: policy gradient

Пусть $R = \sum_{t=t_0}^{T} r_{s_t}$, тогда policy network θ и value network θ_ν:

$$d\theta \leftarrow d\theta + \underbrace{\nabla_\theta \log \pi(a_i|s_i;\theta)(R - V(s_i;\theta_v))}_{\text{REINFORCE policy update}}$$

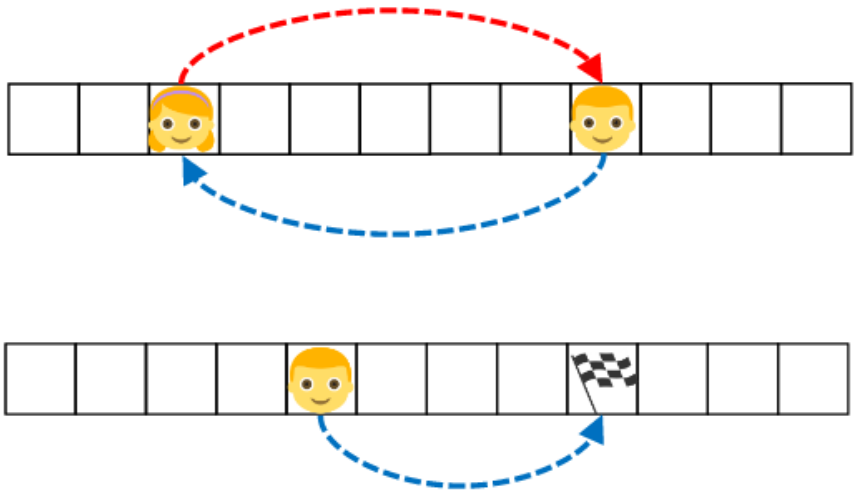
$$d\theta_v \leftarrow d\theta_v + \underbrace{\partial(R - V(s_i;\theta_v))^2}_{\text{advantage}}/\partial\theta_v$$

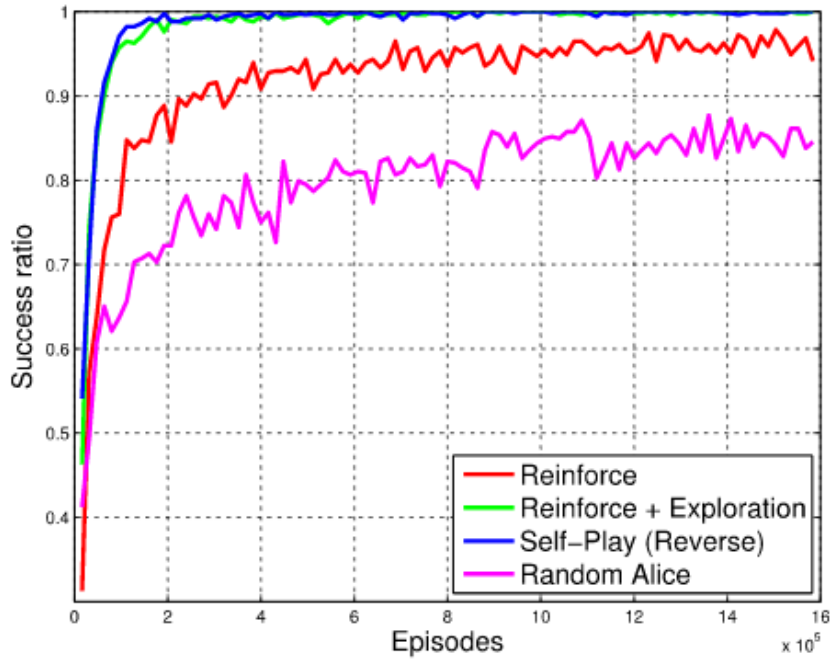
Предложенный Policy gradient (одна нейросеть θ с двумя выходами):

$$\Delta\theta = \sum_{t=1}^{T} \left[\frac{\delta \log \pi_\theta(a_t|s_t)}{\delta\theta} \left(\sum_{i=t}^{T} r_i - b_\theta(s_t) \right) - \lambda \frac{\delta}{\delta\theta} \left(\sum_{i=t}^{T} r_i - b_\theta(s_t) \right)^2 \right]$$

Эксперименты: Long Hallway

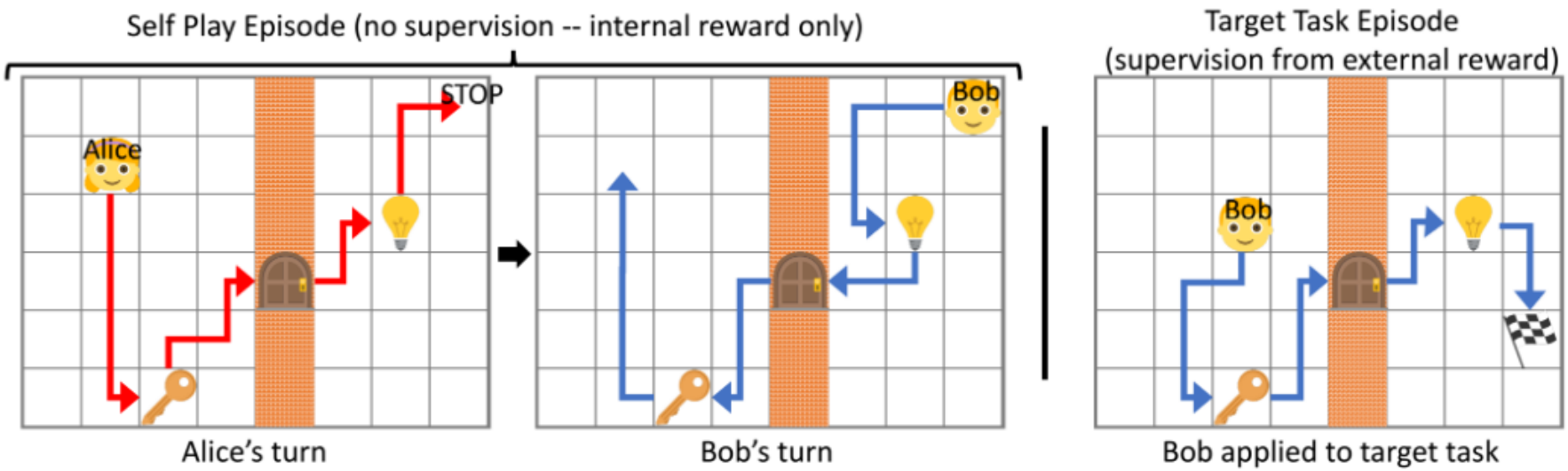
- Агент ходит по коридору
- Реверсивная среда
- Даже обычный Reinforce хорошо работает





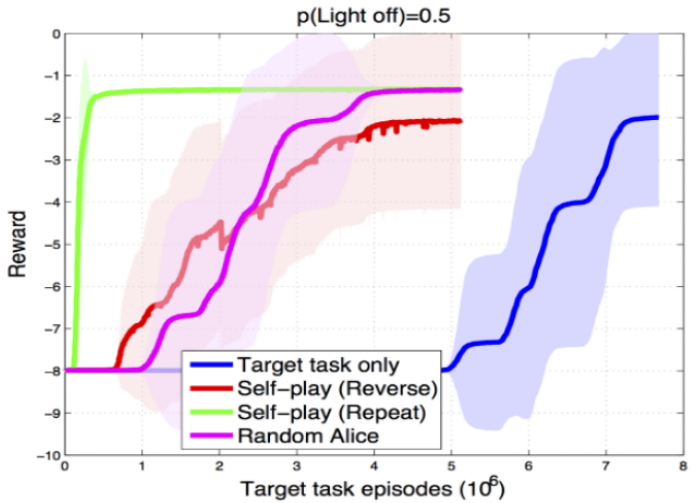
Эксперименты: Mazebaze: Light Key

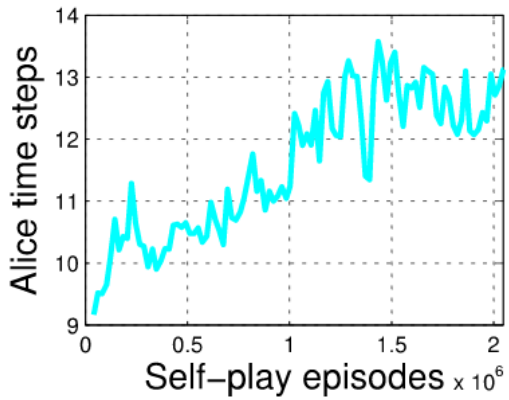
- Надо дойти до флага, найдя ключ и включив свет
- Здесь показан reverse self-play

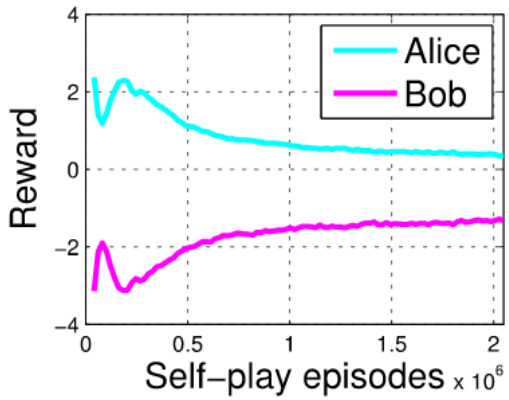


Эксперименты: Mazebaze: Light Key

- Repeat self-play очень хорош
- Хорошо видно как учится Алиса и со временем становится сложнее, получая все меньший ревард с развитием Боба.

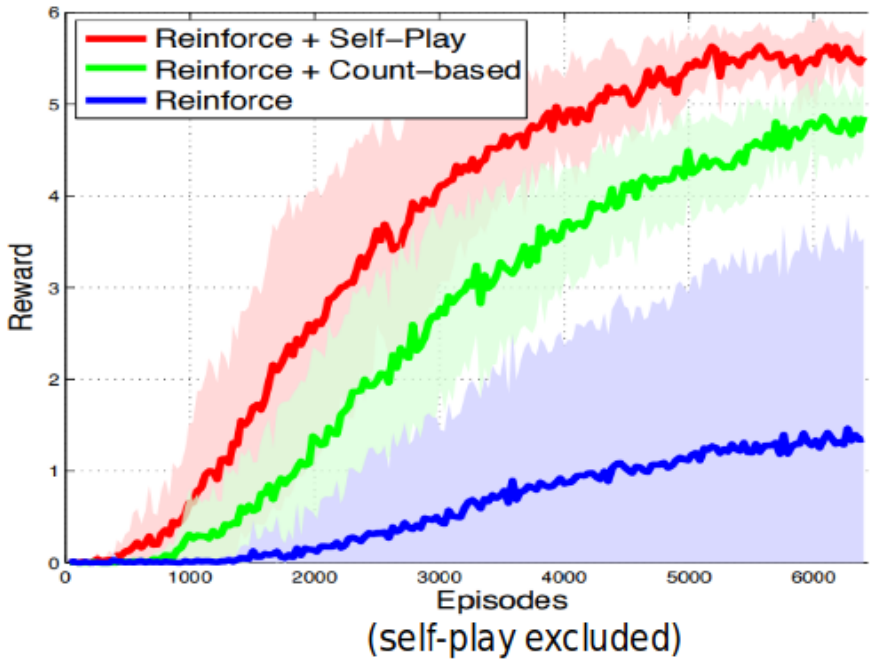






Эксперименты: StarCraft

- Необходимо котролировать среду с многими юнитами. Однако, никаких сражений, только развитие.
- Бобу нужно создать столько же юнитов и сохранить столько же минералов, сколько есть у Алисы.



Выводы

- Агент изучает среду с помощью self-play и его сильно проще дообучить на target task
- Метод хорошо работает, как с дискретными, так и с более сложными средами
- Хороший способ задать reward в средах с только одним агентом