# Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play

D. Sharafyan

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

# Table of contents

1. Asymmetric self-play

2. Policy gradient

3. Experiments

# Motivation

Cons of supervised approaches to reinforcement learning

- requires a huge number of episodes to learn
- agent learns the environment at the same time as it tries ti understand which trajectories lead to rewards

The idea is to use unsupervised training for an agent, so it can learn the environment without any external reward

# Self-play

Agent has two separate "minds": Alice and Bob
During self-play episodes

- Alice's job is to propose a task to Bob
- Bob's job is to complete given task

When presented with a target task episode, Bob is used to perform it

# Self-play

Given approach restricted to two classes of environment:

- reversible
- ones that can be reset to their initial state

# Self-play

Alice starts at state $s_0$ and proposes task by doing it, i.e. executing a sequence of actions that takes the agent to a state $s_t$. Then she outputs a STOP action and hands control over to Bob.
Now there's two options

- if environment is reversible then Bob's task is to return from $s_t$ to $s_0$
- else Alice's STOP action reinitializes environment and Bob's task is to repeat Alice's task, i.e. reach $s_t$ from $s_0$

# Rewards for Bob and Alice

Let $t_A$ be the time until Alice performs STOP action and $t_B$ is the time taken by Bob to complete his task. The total length of episode is limited to $t_{Max}$, so if Bob fails to complete his task in time we set $t_B = t_{Max} - t_A$ Bob's reward is

$$R_B = -\gamma t_B$$

And Alice's reward is

$$R_A = \gamma \max(0, t_B - t_A)$$

So Alice is trying to find the simplest task that Bob cannot complete

# Parametrizing Alice and Bob's actions

In Alice's case the function will be

$$a_A = \pi_A(s_t, s_0)$$

where $s_0$ is the observation of the initial state of the environment and $s_t$ is the observation of current state. In Bob's case the function will be

$$a_B = \pi_B(s_t, s^*)$$

where $s^*$ is the target state that Bob has to reach
In all experiments authors use policy gradient with a baseline for optimizing policies.

# Table of contents

1. Asymmetric self-play

2. Policy gradient

3. Experiments

# Definitions

$$Q(s,a) = \mathrm{E}_{s_{t+1},a_{t+1},\dots}\left[\sum_{l=0}^{\infty}\gamma^{l} r_{t+l}\right]$$

$$V_{\pi}(s_t) = \mathrm{E}_{a_t,s_{t+1},\dots}\left[\sum_{l=0}^{\infty}\gamma^{l} r_{t+l}\right]$$

$$A_{\pi}(s,a) = Q_{\pi}(s,a) - V_{\pi}(s)$$

where $a_t \sim \pi(a_t|s_t)$, $s_{t+1} \sim P(s_{t+1}|s_t,a_t)$

# Policy objective function

- Given policy $\pi_\theta(a|s)$ with parameters $\theta$, find best $\theta$
- So we need somehow measure the quality of policy $\pi_\theta(a|s)$
- We can use expected discounted reward

$$J(\theta) = \mathrm{E}_{s \sim p(s), a \sim \pi_\theta(s|a)} Q(s, a) = \int\limits_s p(s) \int\limits_a \pi_\theta(a|s) Q(s, a) dads$$

$$\begin{aligned} \nabla J(\theta) &= \int\limits_s p(s) \int\limits_a \nabla \pi_\theta(a|s) Q(s, a) dads \\ &= \int\limits_s p(s) \int\limits_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) Q(s, a) dads \\ &= \mathrm{E}_{s \sim p(s), a \sim \pi_\theta(s|a)} \nabla \log \pi_\theta(a|s) Q(s, a) \end{aligned}$$

# REINFORCE

```
Initialize NN weights $\theta_0$

for i = 1 to m do
    Sample N sessions Z under current $\pi_\theta(a|s)$
    Evaluate policy gradient
        $$\nabla J = \frac{1}{N}\sum_{i=0}^{N}\sum_{s,a\in Z_i}\nabla \log \pi_\theta(a|s)Q(s,a)$$
    Update weights
        $$\theta_{i+1} = \theta_i + \alpha\nabla J$$
```

# Policy gradient with baseline

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

where $b(s_t, \theta)$ is a baseline function, computed via an extra head on the network producing the action probabilities.

# Table of contents

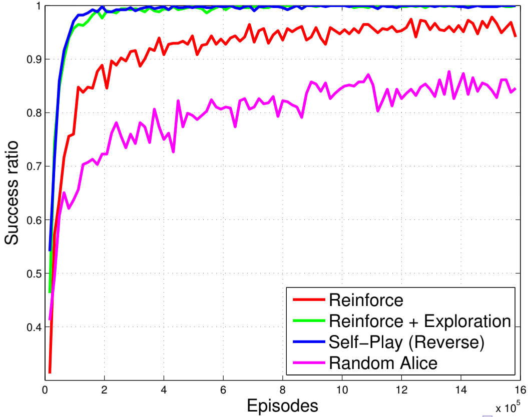
1. Asymmetric self-play

2. Policy gradient

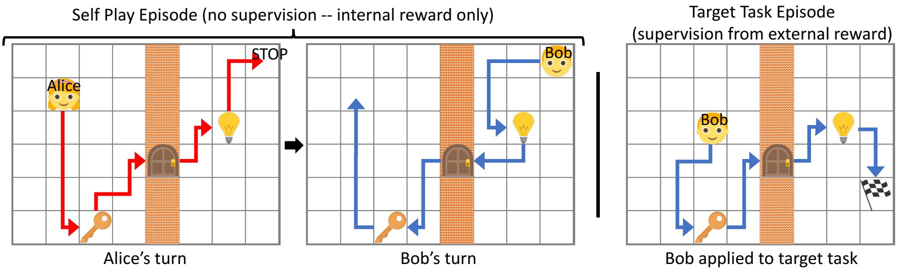
3. Experiments

# Long hallway

The environment consists of $M$ states $\{s_1, s_2, ..., s_M\}$ arranges as a chain. There is 3 actions: "left", "right", "stop". The goal is to reach target state and execute stop action. Here $M = 25$ and $t_{Max} = 30$. Exploration gives bonus reward $\frac{\alpha}{N_s}$ where $N_s$ is the number of times the agent has been in state $s$

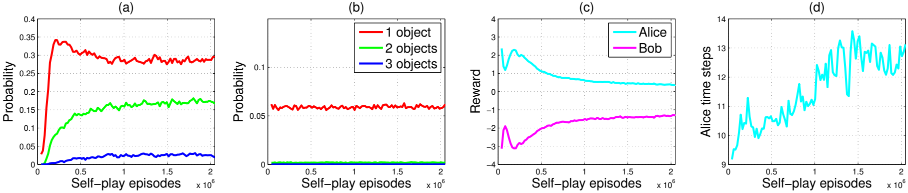


# Mazebase: Light key

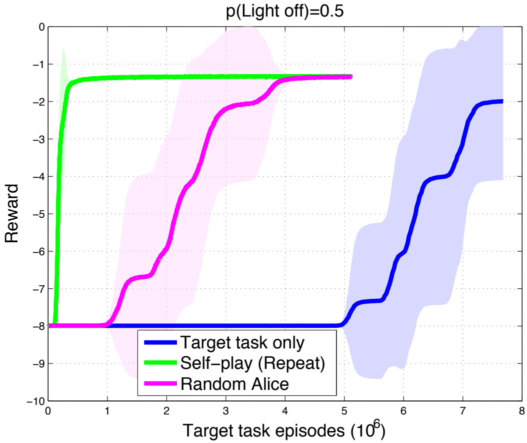

Self Play Episode (no supervision -- internal reward only)
Target Task Episode
(supervision from external reward)
Alice
STOP
Bob
Bob
Alice's turn
Bob's turn
Bob applied to target task

The maze contains a light switch, a key and a wall with a door. An agent can open or close the door by toggling the key switch, and turn on or off light with the light switch. When the light is off, the agent can only see the (glowing) light switch.

# Mazebase results

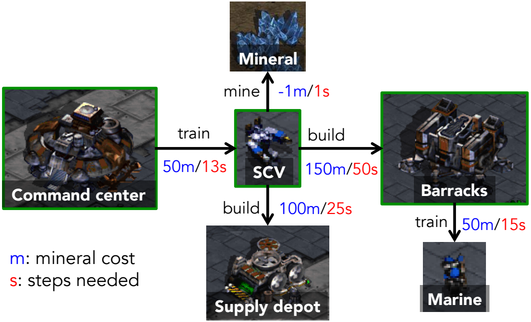


# Mazebase results



# StarCraft: Training Marines

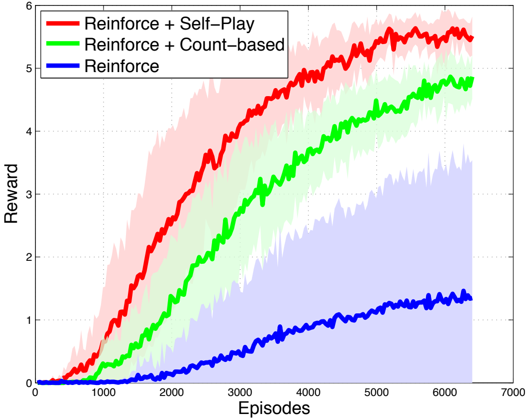
The target task is to build Marine units



m: mineral cost
s: steps needed

# StarCraft results

Count-based exploration is similar to the hallway experiment.


Reinforce + Self−Play
Reinforce + Count−based
Reinforce
Reward
Episodes