Scaling Multi-Agent Reinforcement Learning with Selective Parameter Sharing

Filippos Christianos, Georgios Papoudakis, Arrasy Rahman, Stefano V. Albrecht

contact: f.christianos@ed.ac.uk

THE UNIVERSITY of EDINBURGH informatics

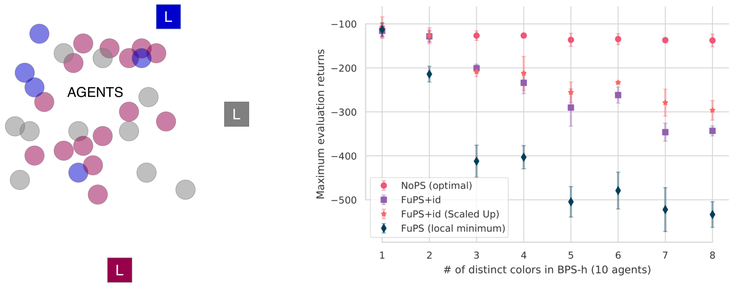
Summary

- **Problem:** Multi-agent reinforcement learning (MARL) with many (possibly heterogeneous) agents.
- **Contribution:** We evaluate typical methods of parameter sharing between agents (i.e. agents using the same sets of parameters for their policies). We find that parameter sharing can be detrimental to reinforcement learning in certain conditions. To address this, we propose a method that determines clusters of agents that can share parameters for improved learning performance.
- **Evaluation:**
  - Evaluated in four multi-agent environments (and over ten task variations). The environments contained groups of agents with different observation/action spaces or goals. We tested tasks with up to 200 agents.
  - Outperforms previously used methods of parameter sharing including fully sharing parameters or appending agent indices to the agents' observation – and not sharing parameters at all.
  - In many tested environments, parameter sharing with our method learns in fewer steps and/or converges to higher returns than the baselines.

Motivation

Parameter Sharing between agents has been used a lot in the literature. Intuitively, when multiple agents need to learn similar things, then shared parameters means less parameters to train = faster learning.

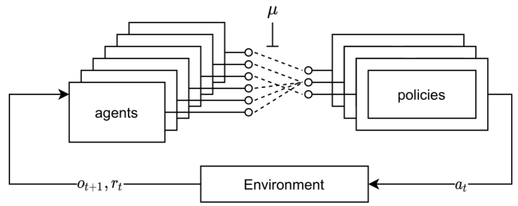
However, when agents differ, parameter sharing can have a detrimental effect on learning. Even on a toy environment, naively sharing all parameters leads to decreased final returns.



Methodology

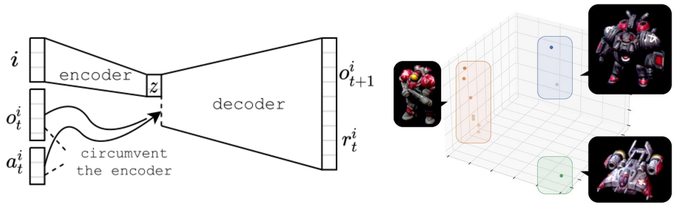
Can we benefit from parameter sharing AND make sure it does not lead to lower overall returns?

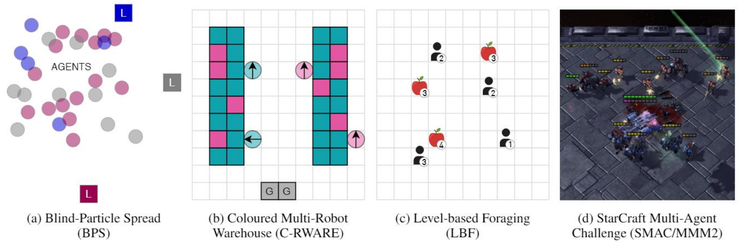
In an environment with N agents we can train K parameterized policies (K < N) and assign them to groups of agents.
Each agent group learns on and uses their respective parameter sets.



How can we decide to partition the agents?

We propose the use of an encoder-decoder model (similar to a VAE). We can make it so that it encodes an agent index to an embedding space, and then use K-means partitioning.

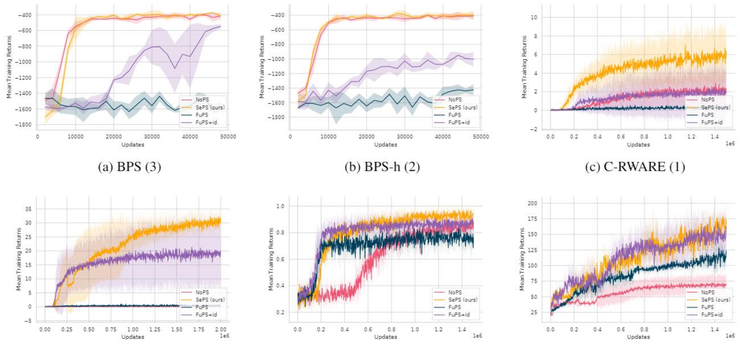




(a) Blind-Particle Spread (BPS) (b) Coloured Multi-Robot Warehouse (C-RWARE) (c) Level-based Foraging (LBF) (d) StarCraft Multi-Agent Challenge (SMAC/MMM2)

Results

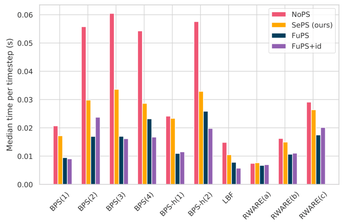
It turns out that Selective Parameter Sharing (SePS) can improve both final converged returns and how fast it converges (depending on the environment)



(a) BPS (3) (b) BPS-h (2) (c) C-RWARE (1)

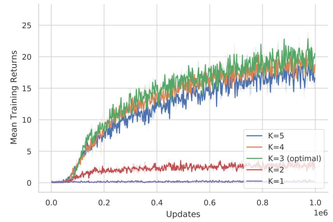
(d) C-RWARE (3) (e) LBF (f) SMAC (MMM2)

Is it reliable? When to use SePS?



Significantly decreases training speed over no parameter sharing when there are many agents.

We could not run 200+ agents without using some form of parameter sharing.



K is easy to find or tune. Some degree of homogeneity must still exist in the environment. Otherwise SePS defaults to no parameter sharing.