Shortest-Path constrained Reinforcement Learning for Sparse Reward Tasks

Sungryull Sohn*
University of Michigan
LG AI Research

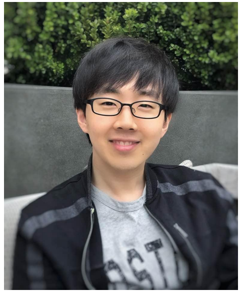
Sungtae Lee*
Yonsei University

Jongwook Choi
University of Michigan

Harm van Seijen
Microsoft Research

Mehdi Fatemi
Microsoft Research

Honglak Lee
LG AI Research
University of Michigan



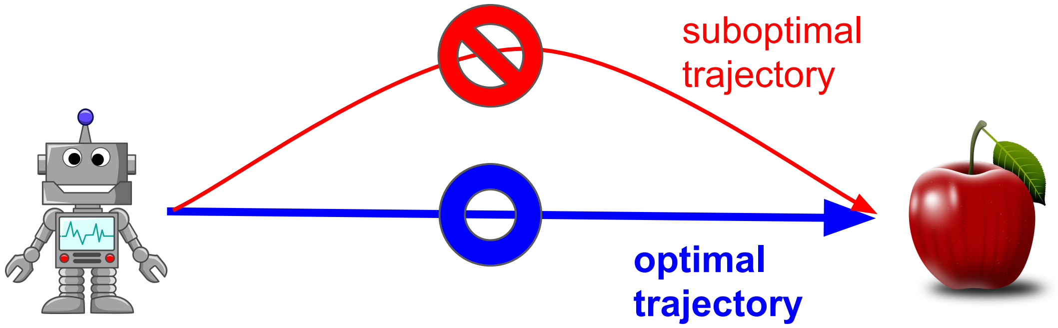
* Equal contributions

Motivation

Model-free RL suffers from the low sample efficiency in sparse reward tasks

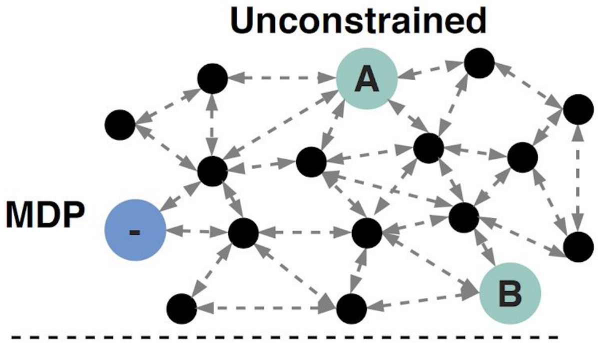
→ We propose to **constrain the policy** to only rollout shortest-path!

- removes the redundancy in the ***agent's transitions***
- improves the sample complexity
- preserves the optimality



Shortest-path constraint

Definition: The policy only rolls out the shortest-path between rewarding states.



Initial state:

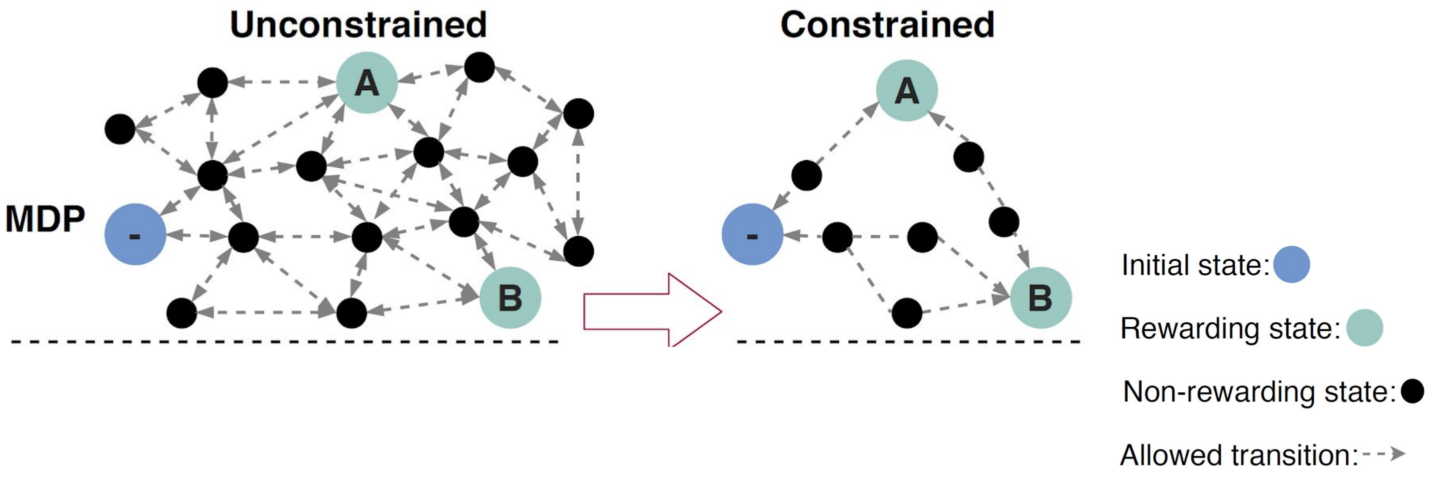
Rewarding state:

Non-rewarding state:

Allowed transition:

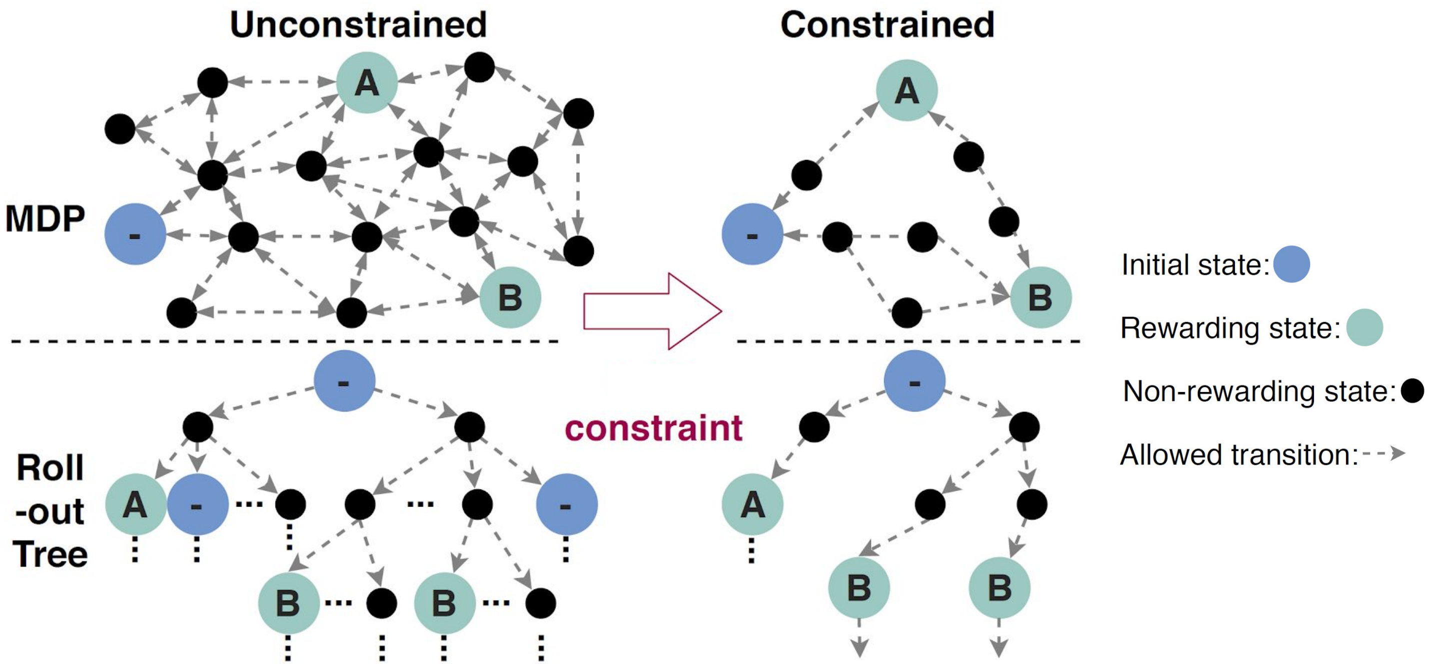
Shortest-path constraint

Definition: The policy only rolls out the shortest-path between rewarding states.



Shortest-path constraint

Definition: The policy only rolls out the shortest-path between rewarding states.



Optimality guarantee

Then, for any MDP with “*mild stochasticity*”

Theorem 1 : Shortest-path constraint preserves optimality.

INTRACTABLE!

k-shortest-path constraint: SP constraint is applied to **the sub-trajectory with length ≤ k**

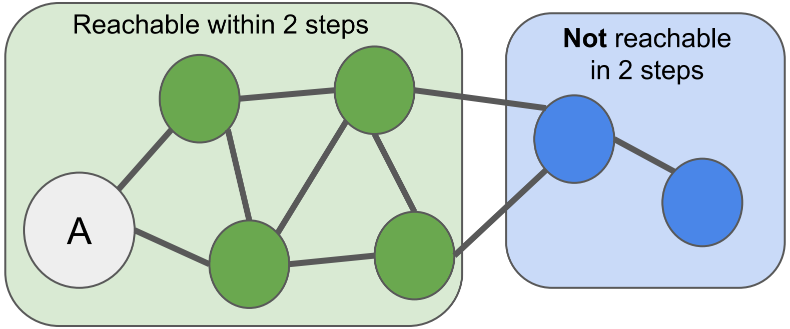
Theorem 2 : k-shortest-path constraint preserves optimality.

TRACTABLE!

Implementation

We use *reachability network* (RNet) [Savinov et al., 2018] to implement k-SP constraint.

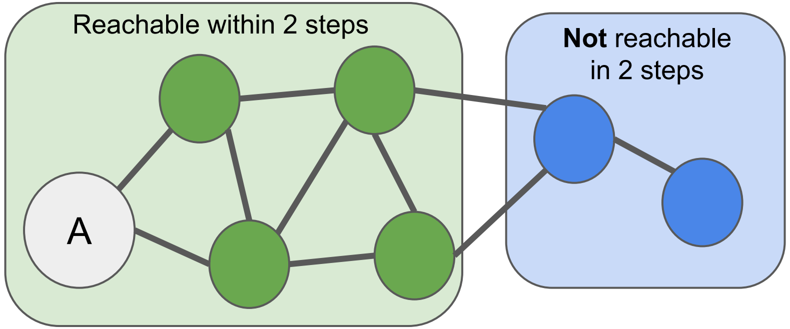
- RNet learns to predict whether *a state is reachable from another within k steps*



Implementation

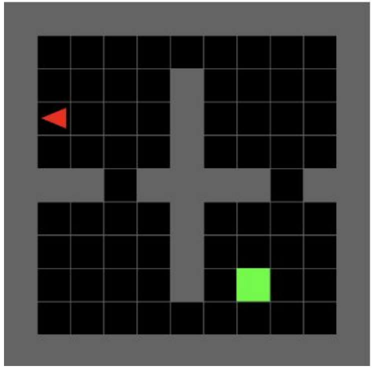
We use *reachability network* (RNet) [Savinov et al., 2018] to implement k-SP constraint.

- RNet learns to predict whether *a state is reachable from another within k steps*
- We apply RNet to the agent's sub-trajectory $[s_{t-k}, \ldots, s_t]$ to test if it's a shortest-path
  - Property: a path is a shortest-path if temporal length = (spatial) length
- RNet can be trained from the agent's experience without any extra supervision.



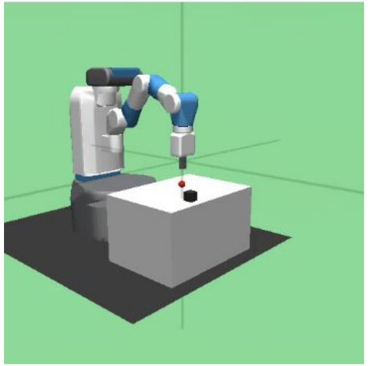
Experiment - domains

MiniGrid



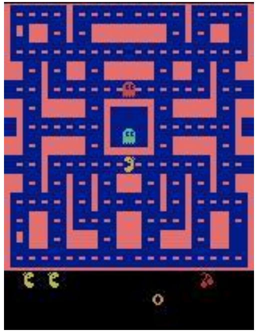
DMLab

Fetch



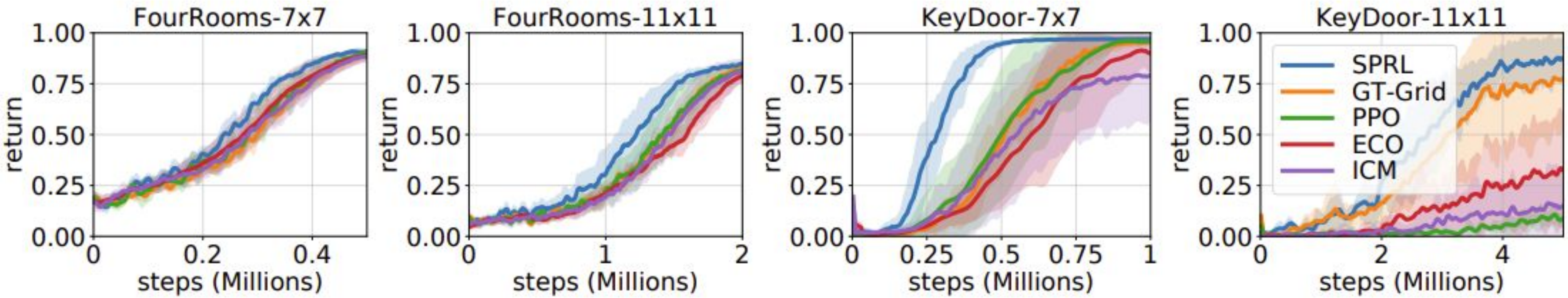
ATARI

Montezuma



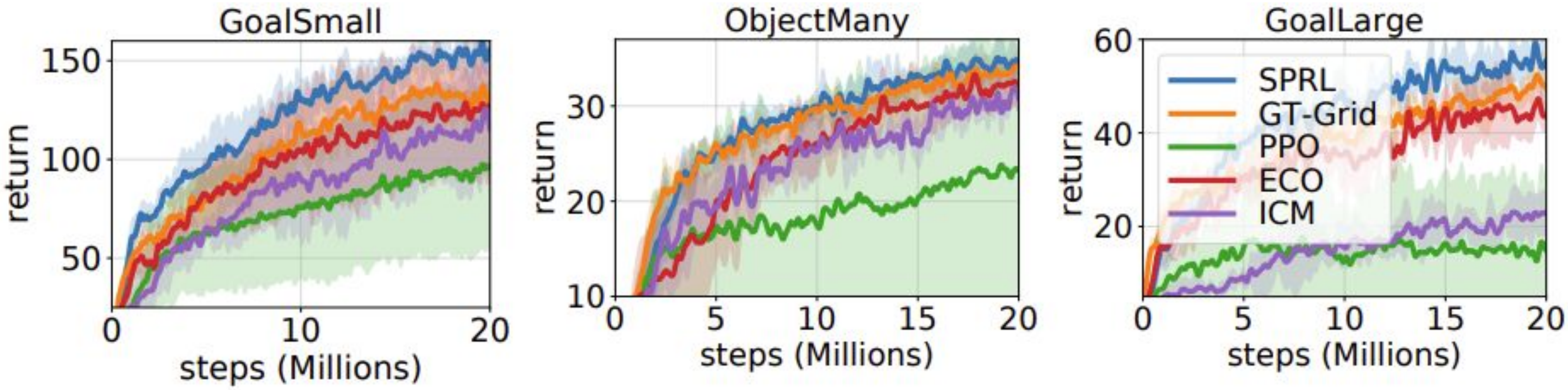
MsPacman

Experiment - Minigrid



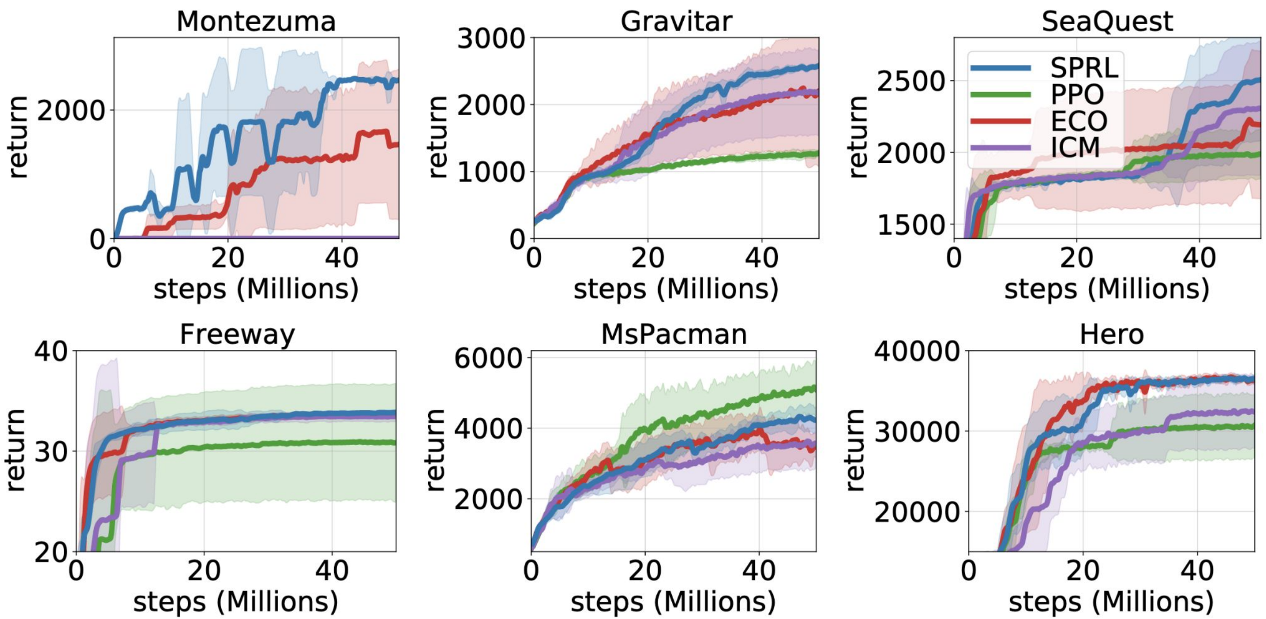
- SPRL even outperforms GT-Grid, an upper-bound of novelty-seeking exploration methods.
- SPRL improves
  - exploration by suppressing unnecessary explorations.
  - exploitation by reducing the policy search space.

Experiment - DMLab



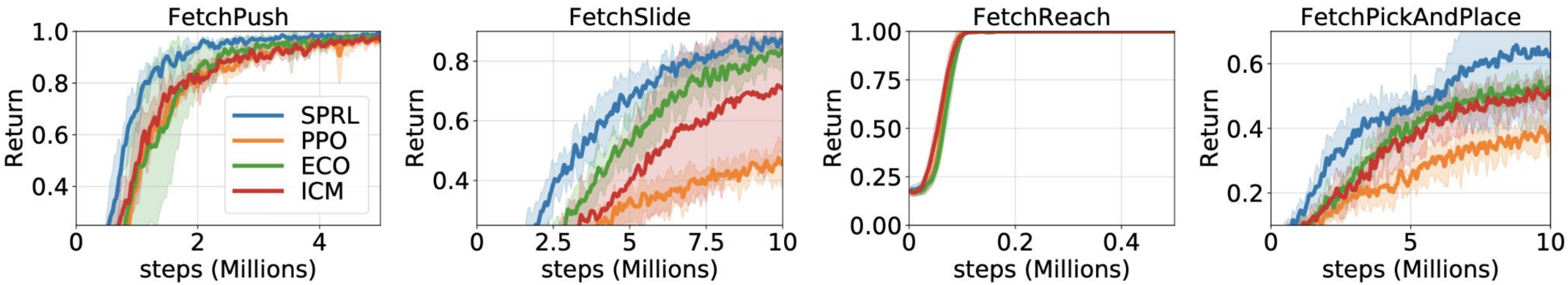
- SPRL outperforms GT-Grid in DMLab.
- SPRL has the largest improvement in GoalLarge task, where both the map layout is largest and the reward is most sparse.

Experiment - ATARI



- Evaluated on 6 tasks : 2 Sparse reward tasks, 1 Dense reward tasks, 3 Non-navigational tasks
- SPRL outperforms the baselines in 5 out of 6 tasks except for Ms.Pacman, a dense reward task.
- The difference between SPRL and PPO is the largest on non-navigational tasks.

Experiment - Fetch



- SPRL outperforms the baselines even in **continuous control tasks**.
- Reachability network reaches an accuracy over 95% before 1M steps.

Conclusion

- We proposed a novel constraint on policy that improves the sample-efficiency of any model-free RL method

- SPRL outperforms strong novelty-seeking exploration baselines

- SPRL opens up a novel direction to improve sample efficiency in reinforcement learning