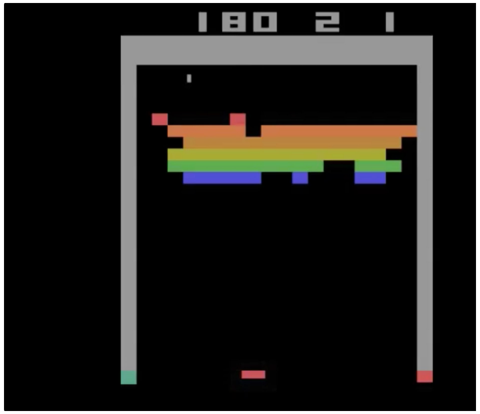
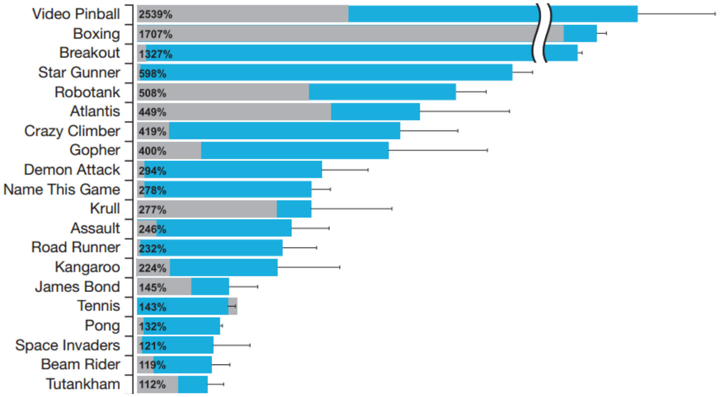
Learning Curriculum Policies for Reinforcement Learning

Sanmit Narvekar and Peter Stone
Department of Computer Science
University of Texas at Austin
{sanmit, pstone} @cs.utexas.edu

Successes of Reinforcement Learning



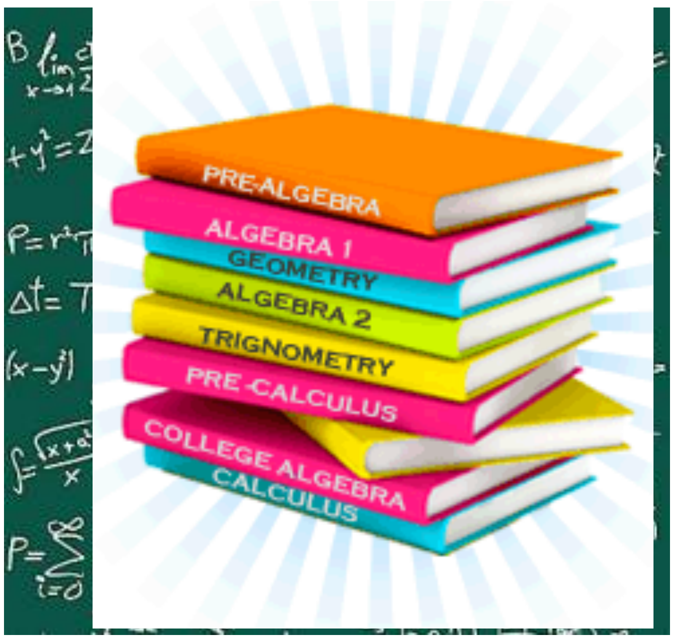


Approaching or passing human level performance

BUT

Can take *millions* of episodes! People learn this <u>MUCH</u> faster

People Learn via Curricula

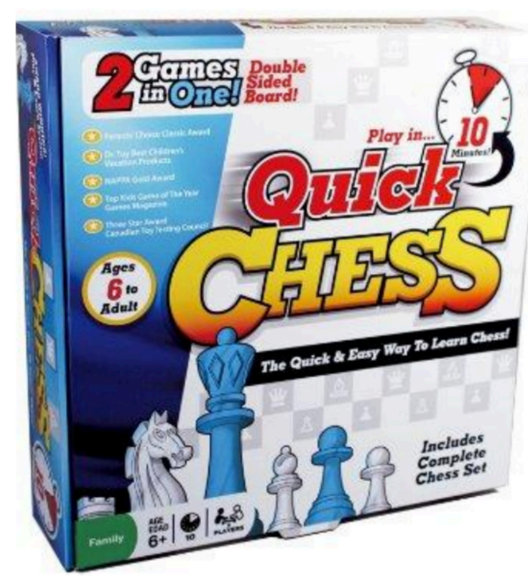


People are able to learn a lot of complex tasks very efficiently

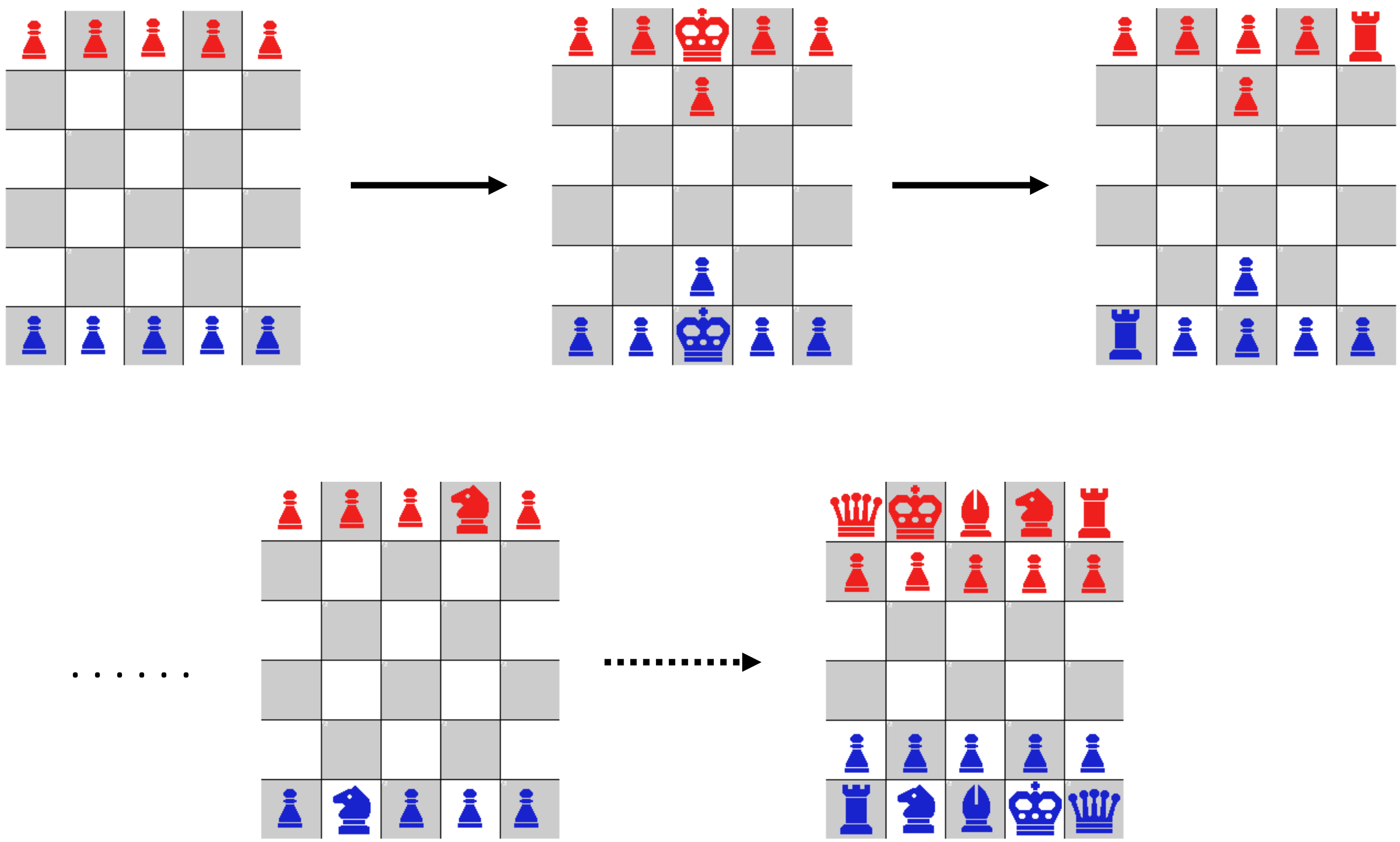
Example: Quick Chess

- Quickly learn the fundamentals of chess

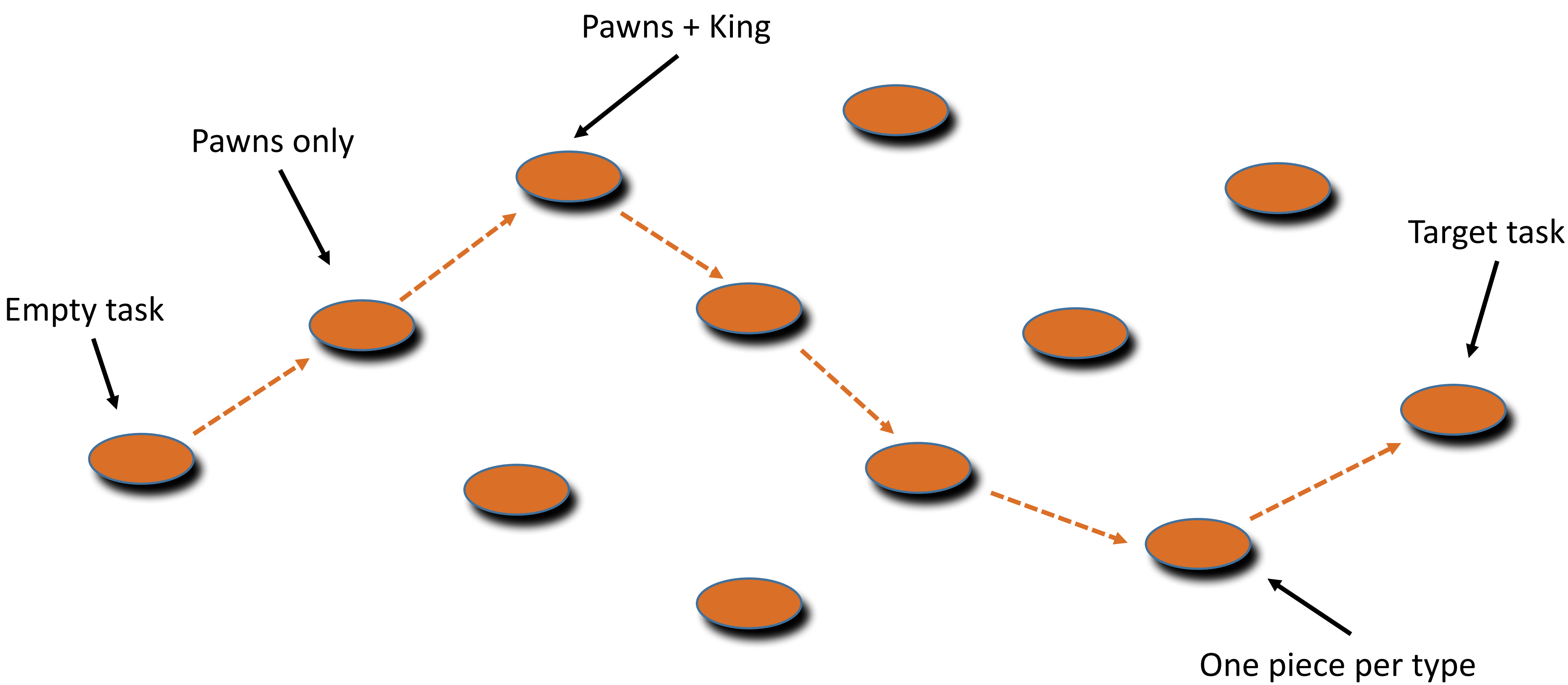
- 5 x 6 board
- Fewer pieces per type
- No castling
- No en-passant



Example: Quick Chess

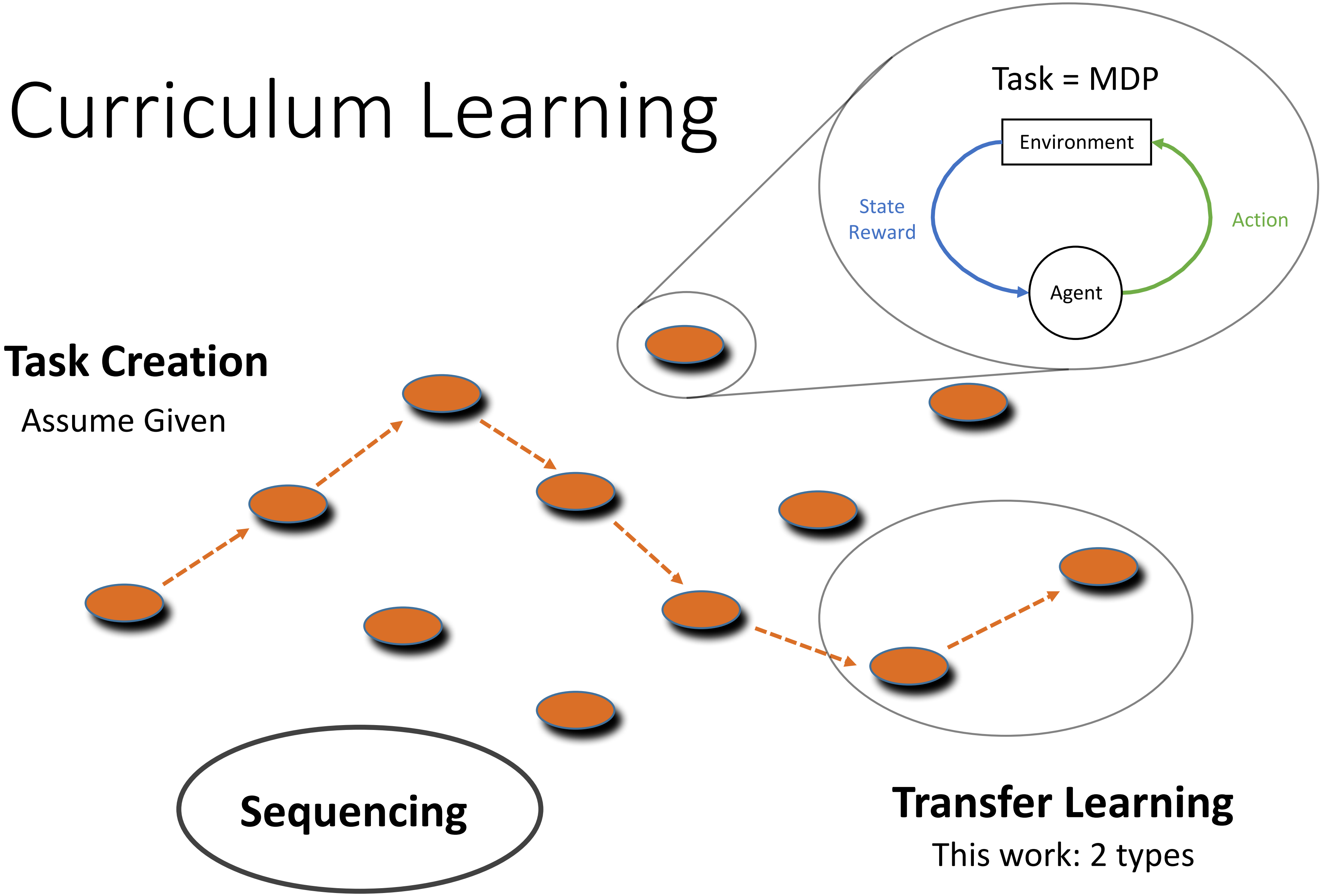


Task Space



- Quick Chess is a curriculum designed for people
- We want to do something similar automatically for autonomous agents

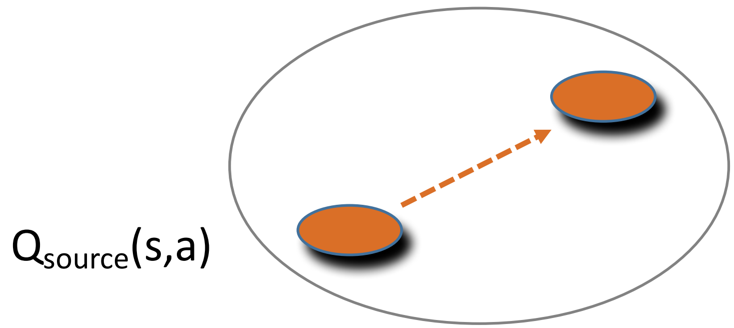
Curriculum Learning



- Curriculum learning is a complex problem that ties task creation, sequencing, and transfer learning

Value Function Transfer

- Initialize Q function in target task using values learned in a source task



- Assumptions:
  - Tasks have overlapping state and action spaces
  - OR an inter-task mapping is provided
    - Existing related work on learning mappings

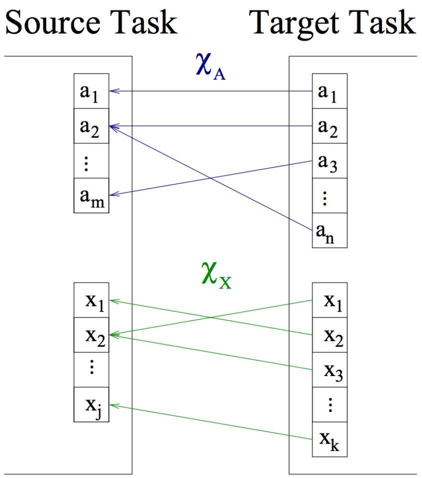


Image credit: Taylor and Stone, JMLR 2009

Reward Shaping Transfer

- Reward function in target task augmented with a shaping reward *f*:

$$\underbrace{r'(s,a,s')}_{\text{New Reward}} = \underbrace{r(s,a,s')}_{\text{Old Reward}} + \underbrace{f(s,a,s')}_{\text{Shaping Reward}}$$

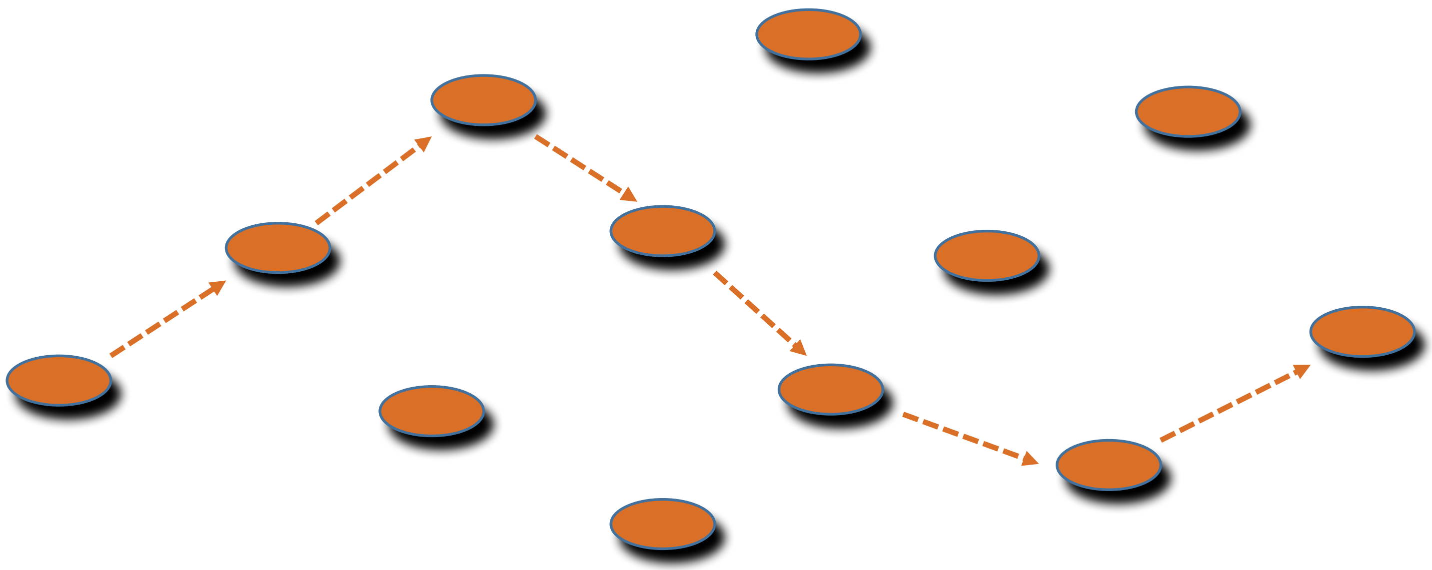
- Potential-based advice restricts f to be difference of potential functions:

$$f(s,a,s') = \Phi(s',\pi(s')) - \Phi(s,a)$$

- Use the value function of the source as the potential function:

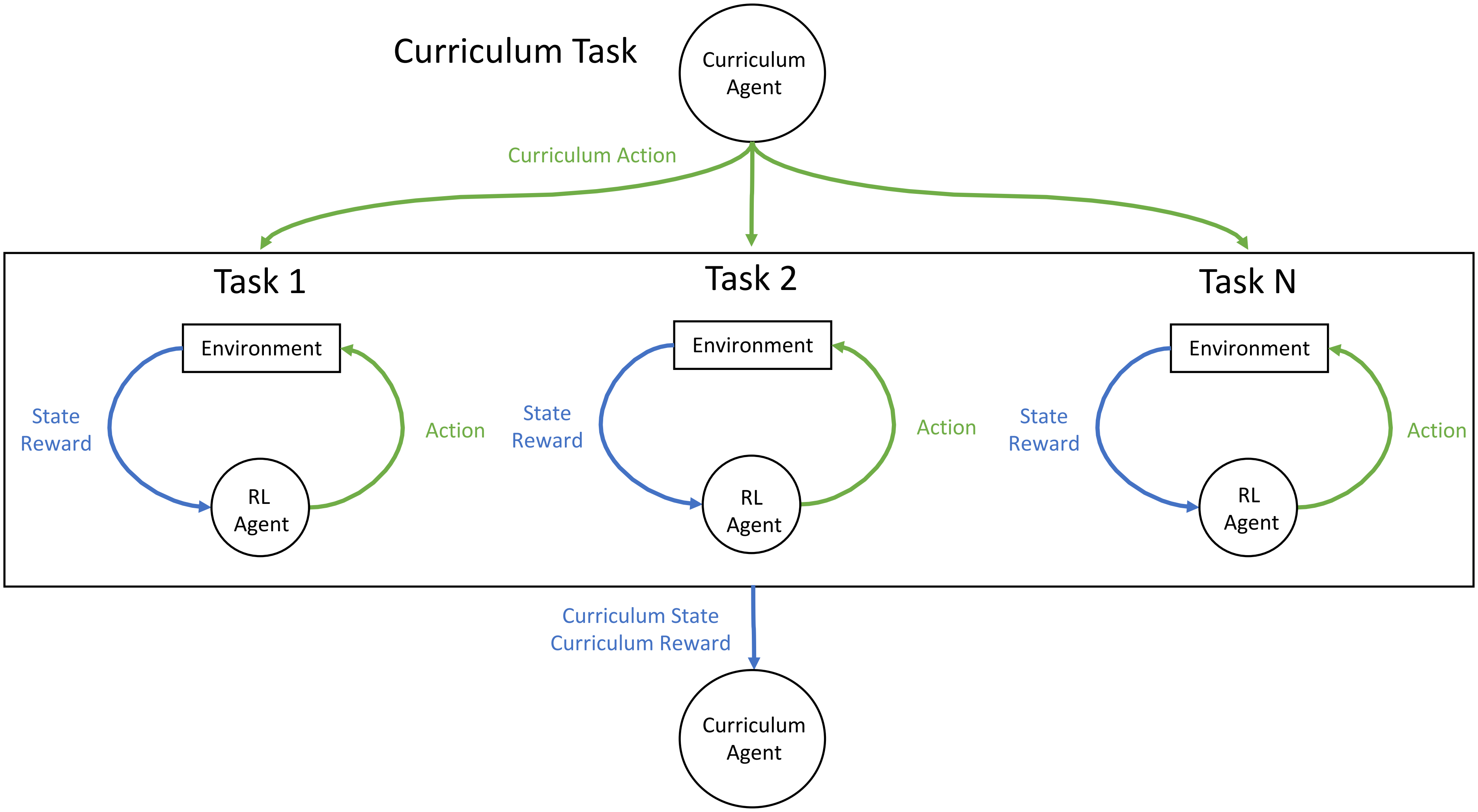
$$\Phi(s,a) = Q_{source}(s,a)$$

The Problem: Autonomous Sequencing

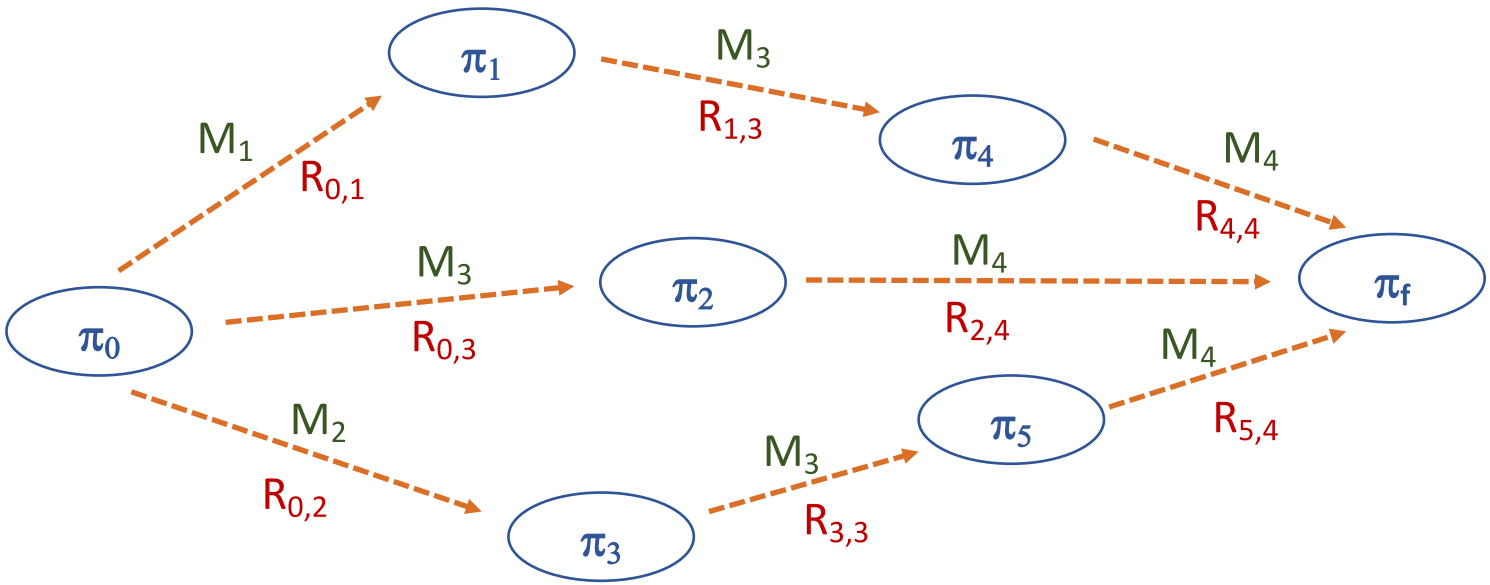


- **Existing work heuristic-based**, such as examining performance on the target task, and using heuristics to select next task
- In this work, we use **learning to do sequencing**

Sequencing as an MDP

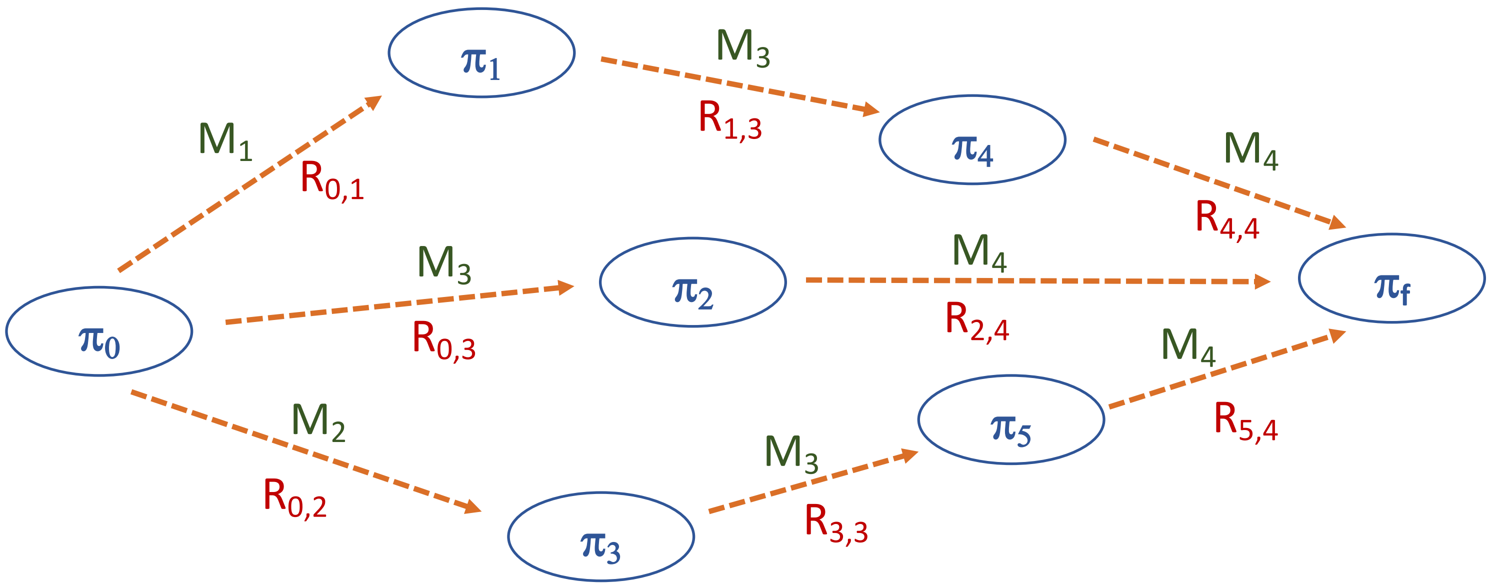


Sequencing as an MDP



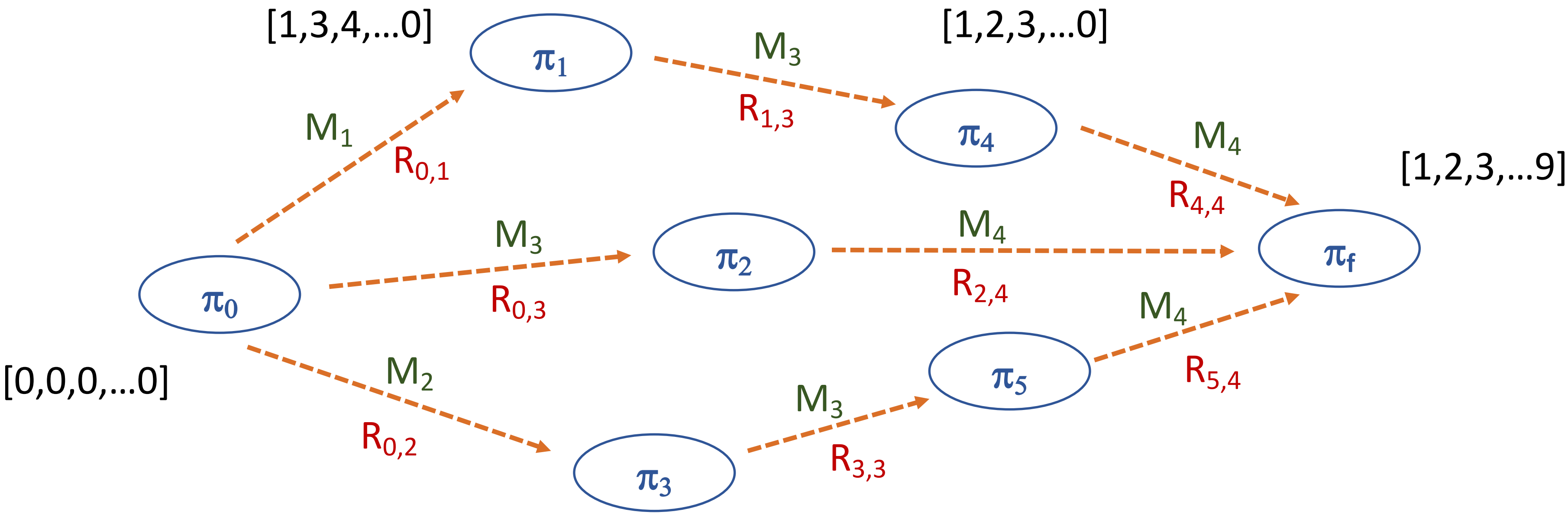
- **State space S^C**: All policies π_i an agent can represent
- **Action space A^C**: Different tasks M_j an agent can train on
- **Transition function $p^C(s^C,a^C)$**: Learning task a^C transforms an agent's policy s^C
- **Reward function $r^C(s^C,a^C)$**: Cost in time steps to learn task a^C given policy s^C

Sequencing as an MDP



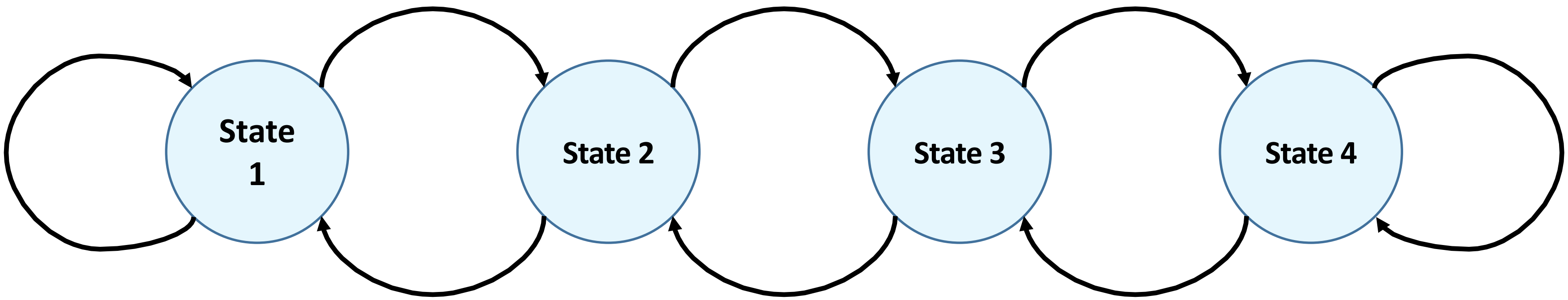
- A policy $\pi^C: S^C \rightarrow A^C$ on this curriculum MDP (CMDP) specifies which task to train on given learning agent policy π_i
- Essentially training a teacher
- How to do learning over CMDP?
- How does CMDP change when transfer method changes?

Learning in Curriculum MDPs



- Express raw CMDP state using the weights of base agent's VF/policy
- Extract features so that similar policies (CMDP states) are "close" in feature space

Example: Discrete Representations



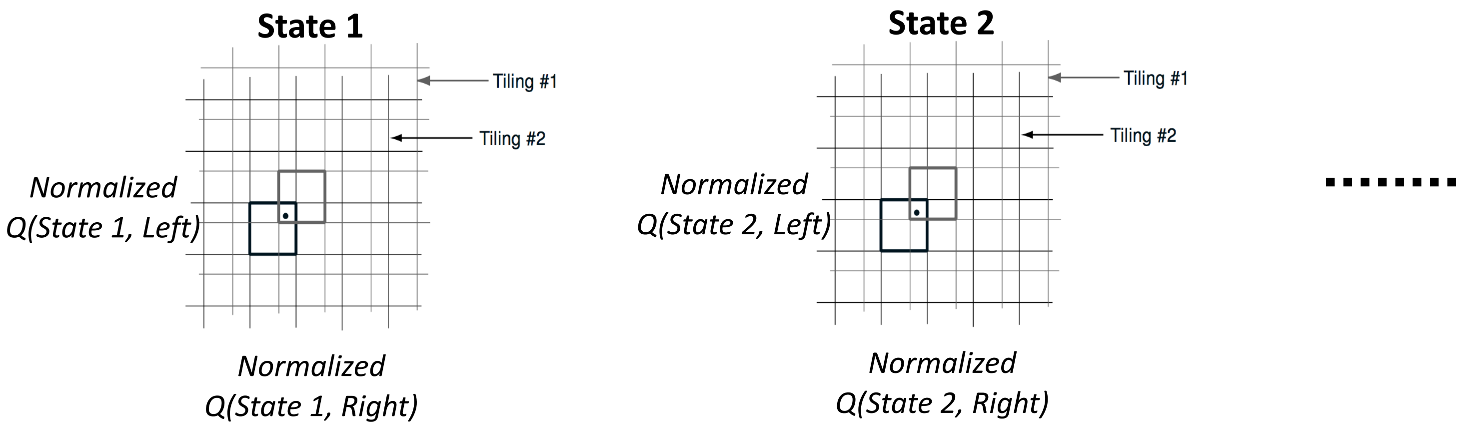
CMDP State 1			
	Left	Right	Policy
State 1	0.3	0.7	→
State 2	0.1	0.9	→
State 3	0.4	0.6	→
State 4	0.0	1.0	→

CMDP State 2			
	Left	Right	Policy
State 1	0.2	0.8	→
State 2	0.2	0.8	→
State 3	0.2	0.8	→
State 4	0.3	0.7	→

CMDP State 3			
	Left	Right	Policy
State 1	0.7	0.3	←
State 2	0.9	0.1	←
State 3	0.6	0.4	←
State 4	0.0	1.0	→

- CMDP states 1 and 2 encode very similar policies, and should be close in CMDP representation space

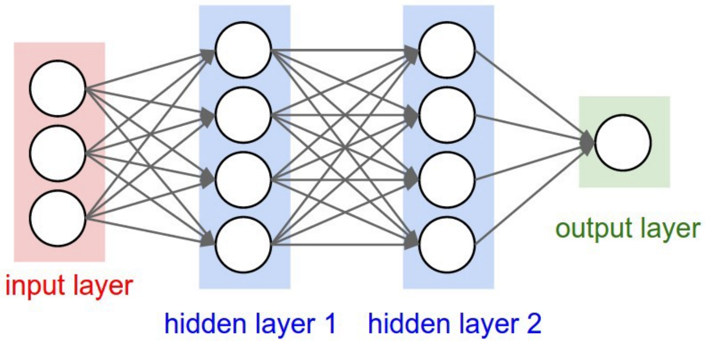
Example: Discrete Representations

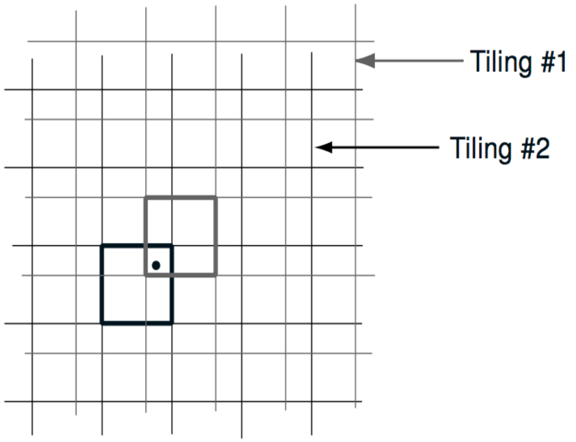


- One approach: use tile coding
- Create a separate tiling on a state-by-state level
- When comparing CMDP states, the more similar the policies are in a primitive state, the more common tiles will be activated
- Each primitive state contributes equally towards the similarity of the CMDP state

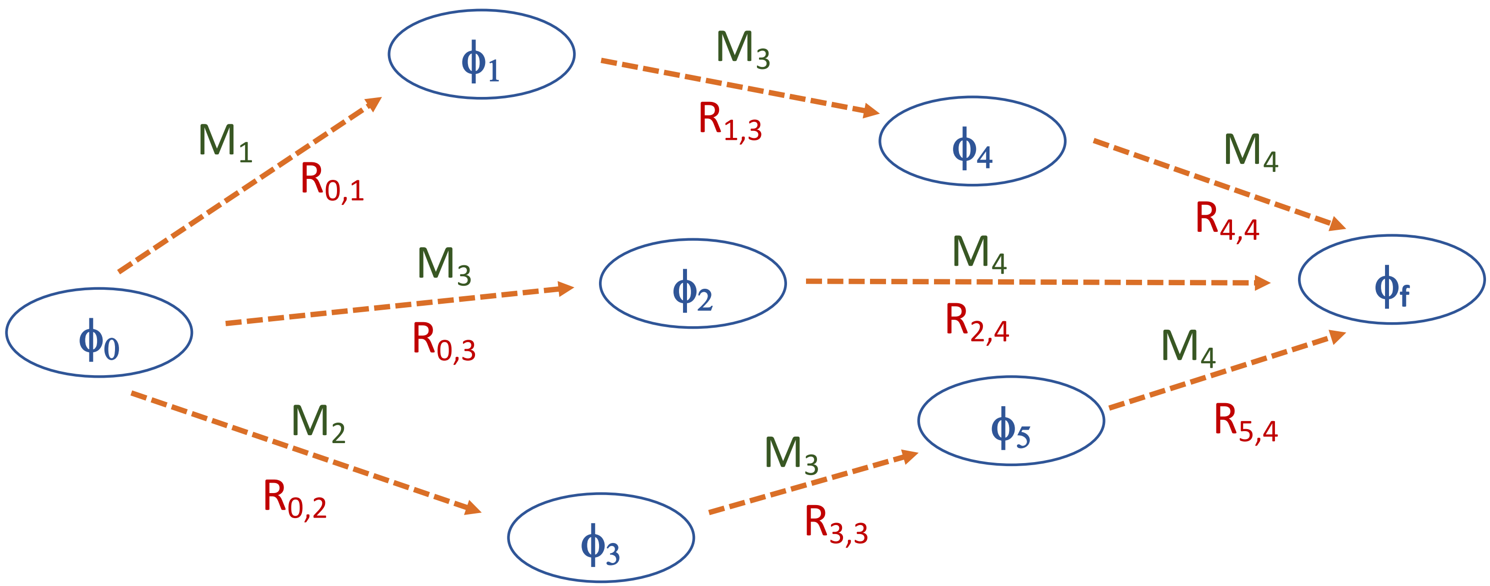
Continuous CMDP Representations

- In continuous domains, weights are not local to a state
- Needs to be done separately for each domain
  - Neural networks
  - Tile coding
  - Etc…
- If the base agent uses a linear function approximator, one can use tile coding over the parameters as before





Changes in Transfer Algorithm



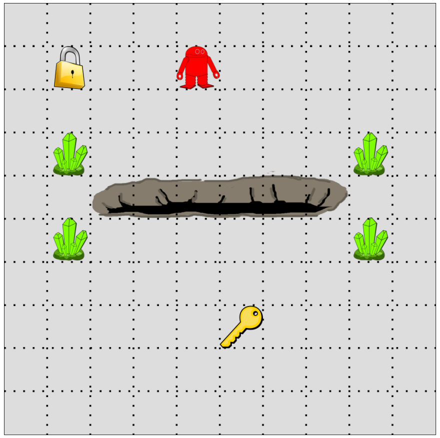
- Transfer method directly affects CMDP state representation and transition function
- CMDP states represent "states of knowledge," where knowledge represented as VF, shaping reward, etc.
- Similar process can be done if knowledge parameterizable

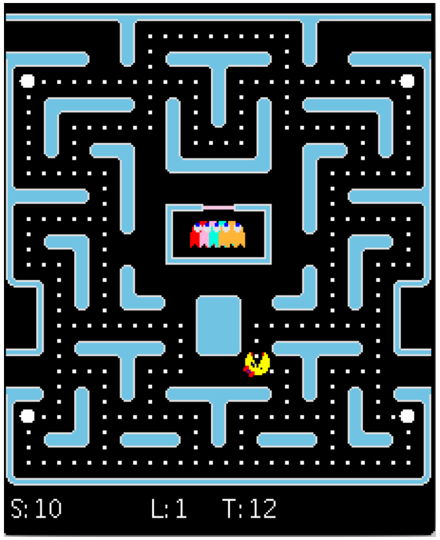
Experimental Results

- Evaluate whether curriculum policies can be learned

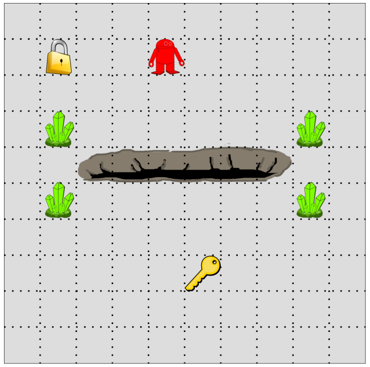
- Grid world
  - Multiple base agents
  - Multiple CMDP state representations

- Pacman
  - Multiple transfer learning algorithms
  - How long to train on sources?





Grid world Setup



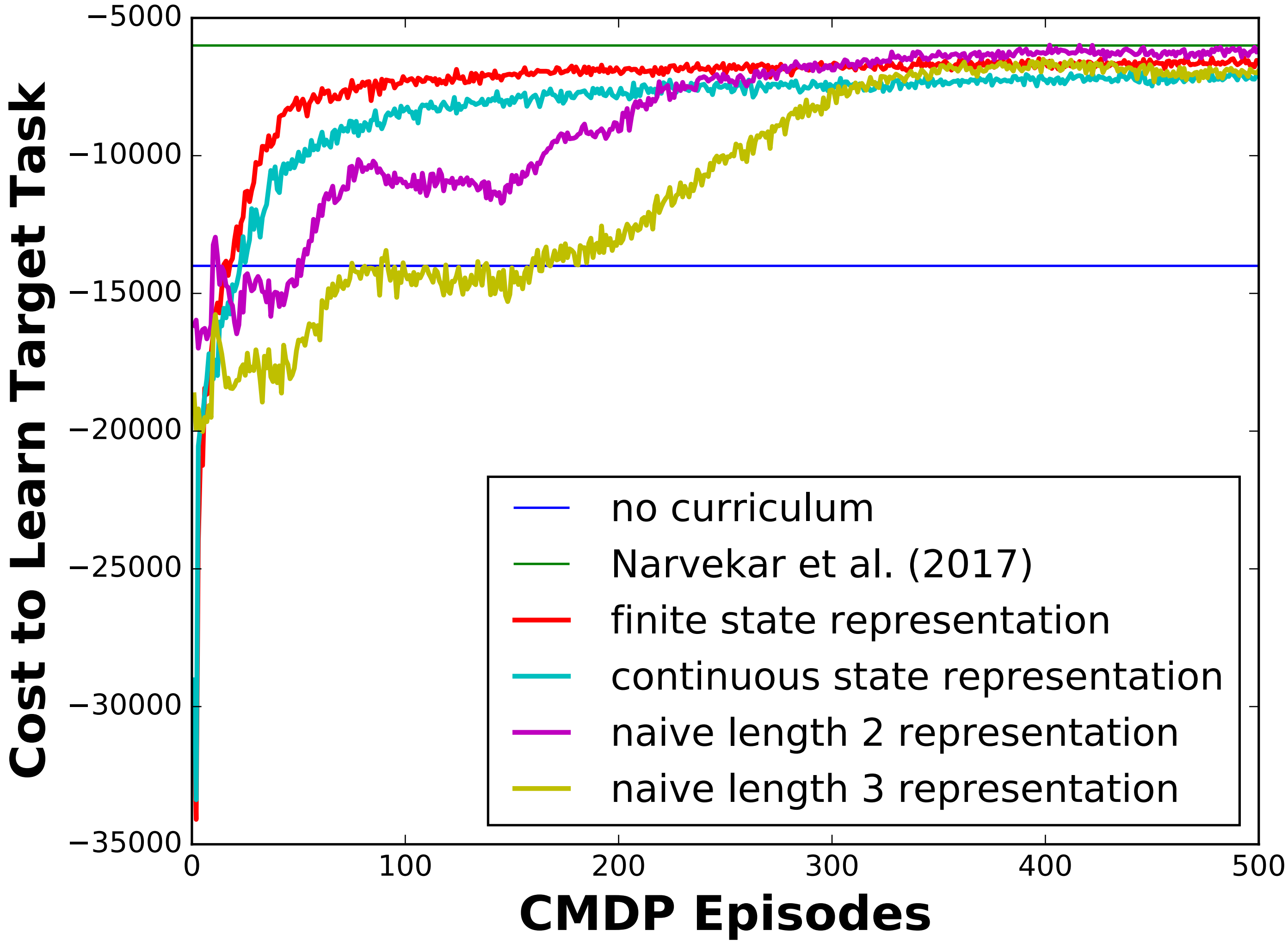
Agent Types

- Basic Agent
  - State: Sensors on 4 sides that measure distance to keys, locks, etc.
  - Actions: Move in 4 directions, pickup key, unlock lock
- Action-dependent Agent
  - State difference: weights on features are shared over 4 directions
- Rope Agent
  - Action difference: Like basic, but can use rope action to negate a pit

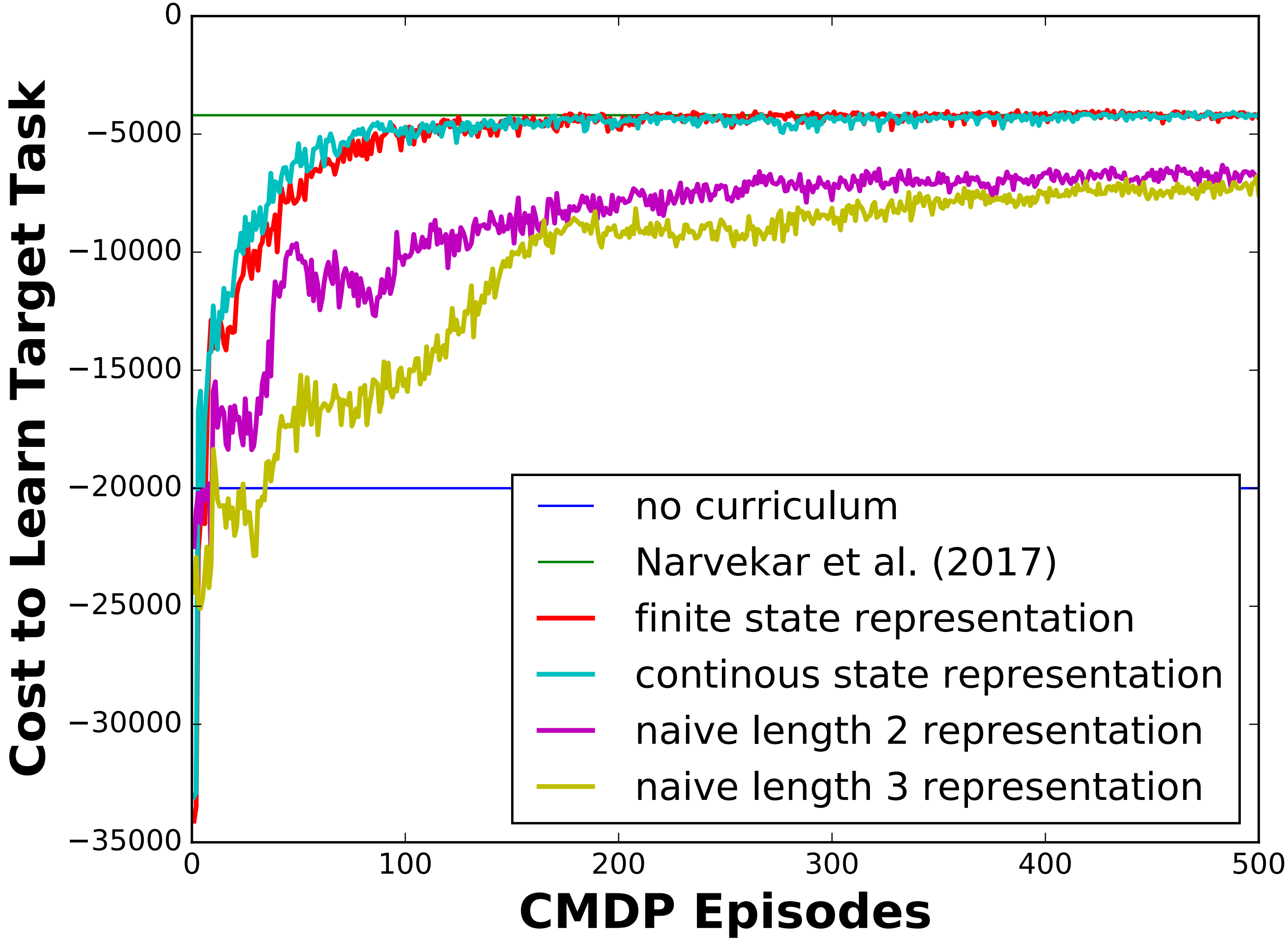
CMDP Representations

- Finite State Representation
  - For discrete domains, groups and normalizes raw weights state-by-state to form CMDP features
- Continuous State Representation
  - Directly uses raw weights of learning agent as features for CMDP agent

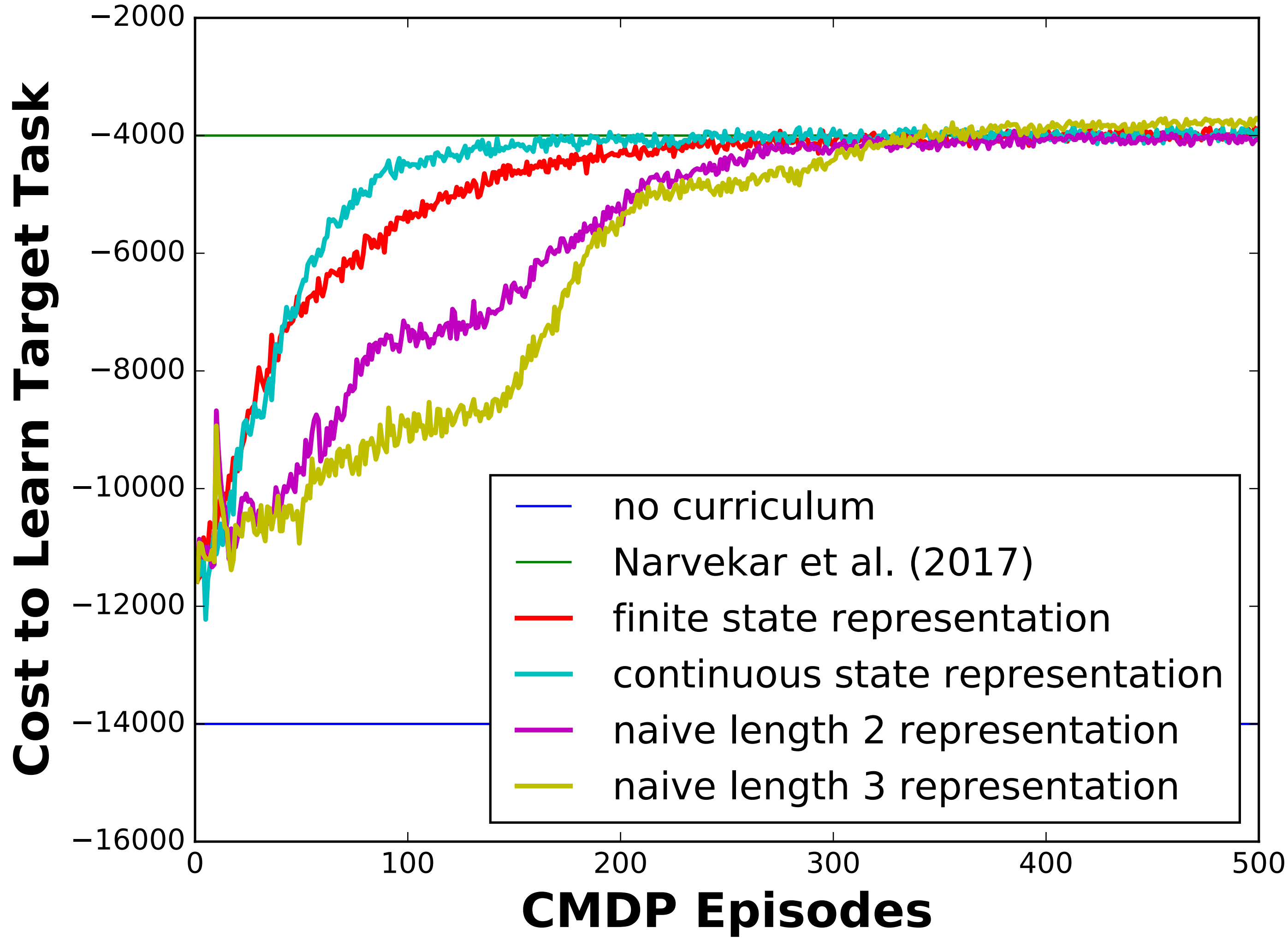
Basic Agent Results



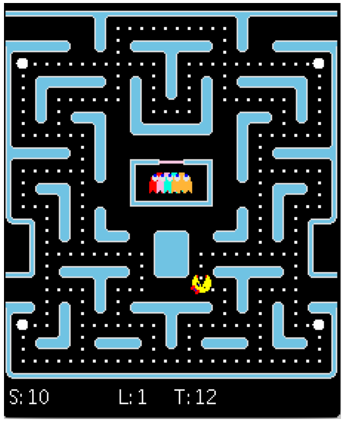
Action-Dependent Agent Results



Rope Agent Results



Pacman Setup



Agent Representation

- Action-dependent egocentric features

CMDP Representation

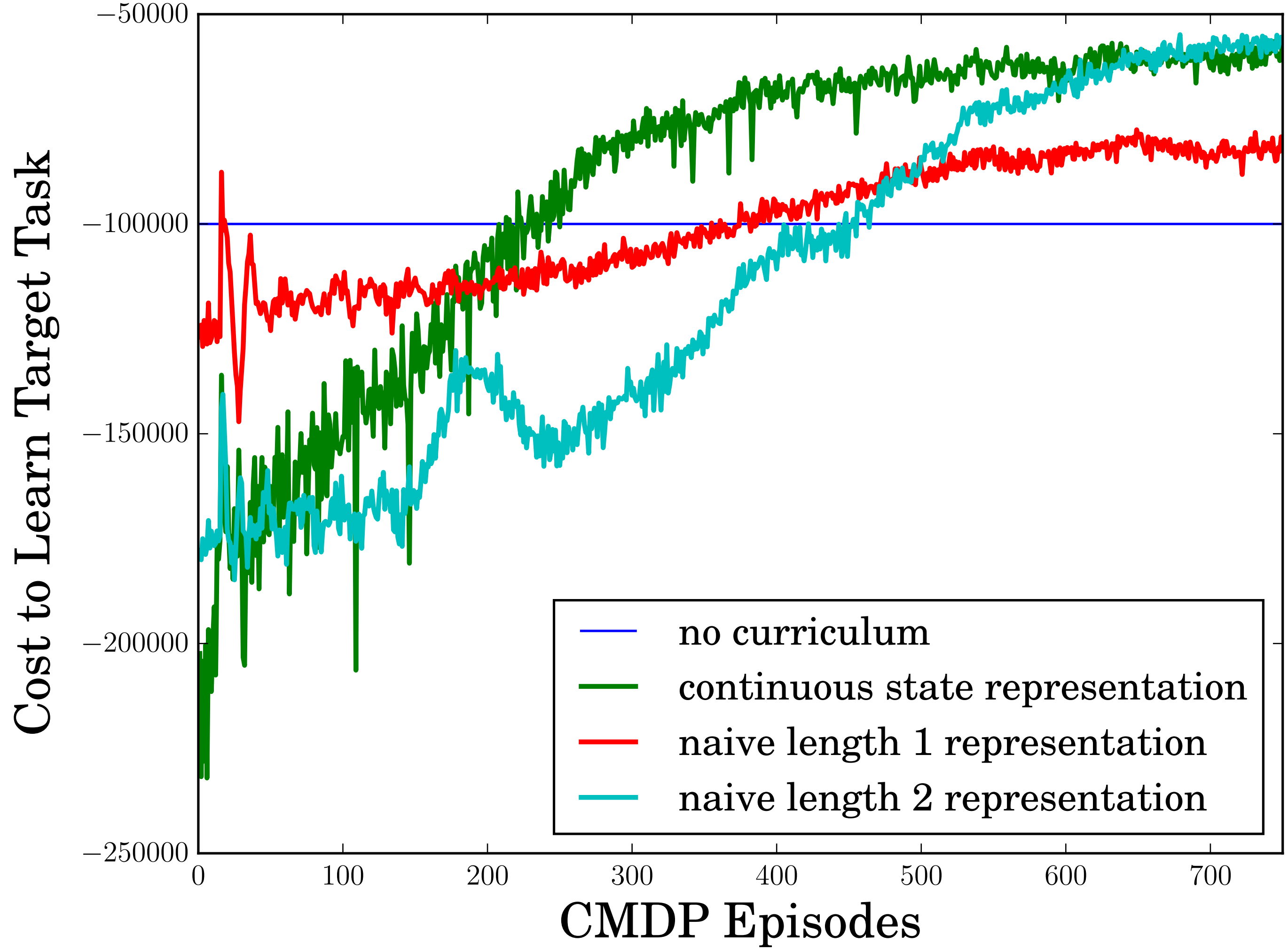
- Continuous State Representation
  - Directly uses raw weights of learning agent as features for CMDP agent

Transfer Methods

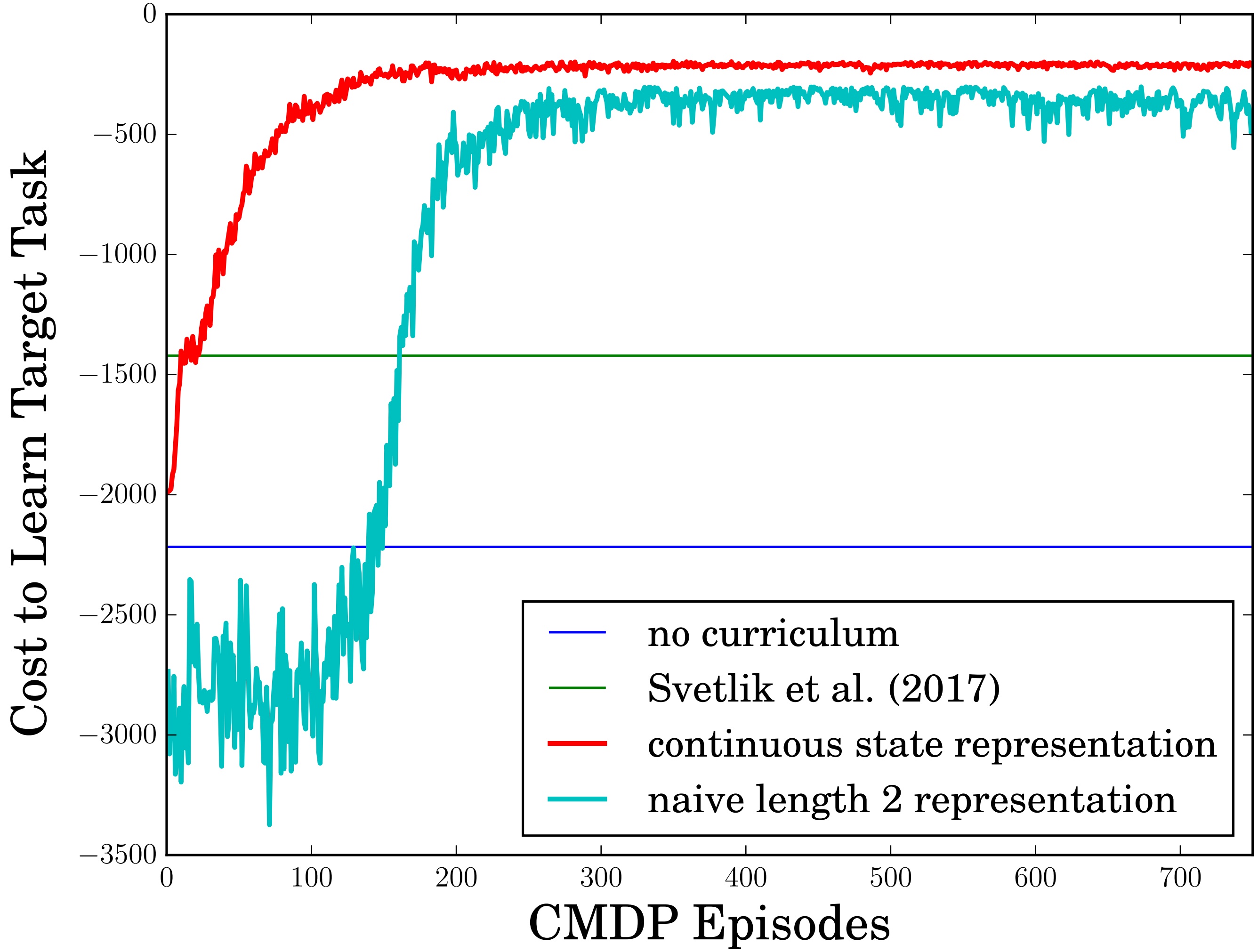
- Value Function Transfer
- Reward Shaping Transfer

How long to train on a source task?

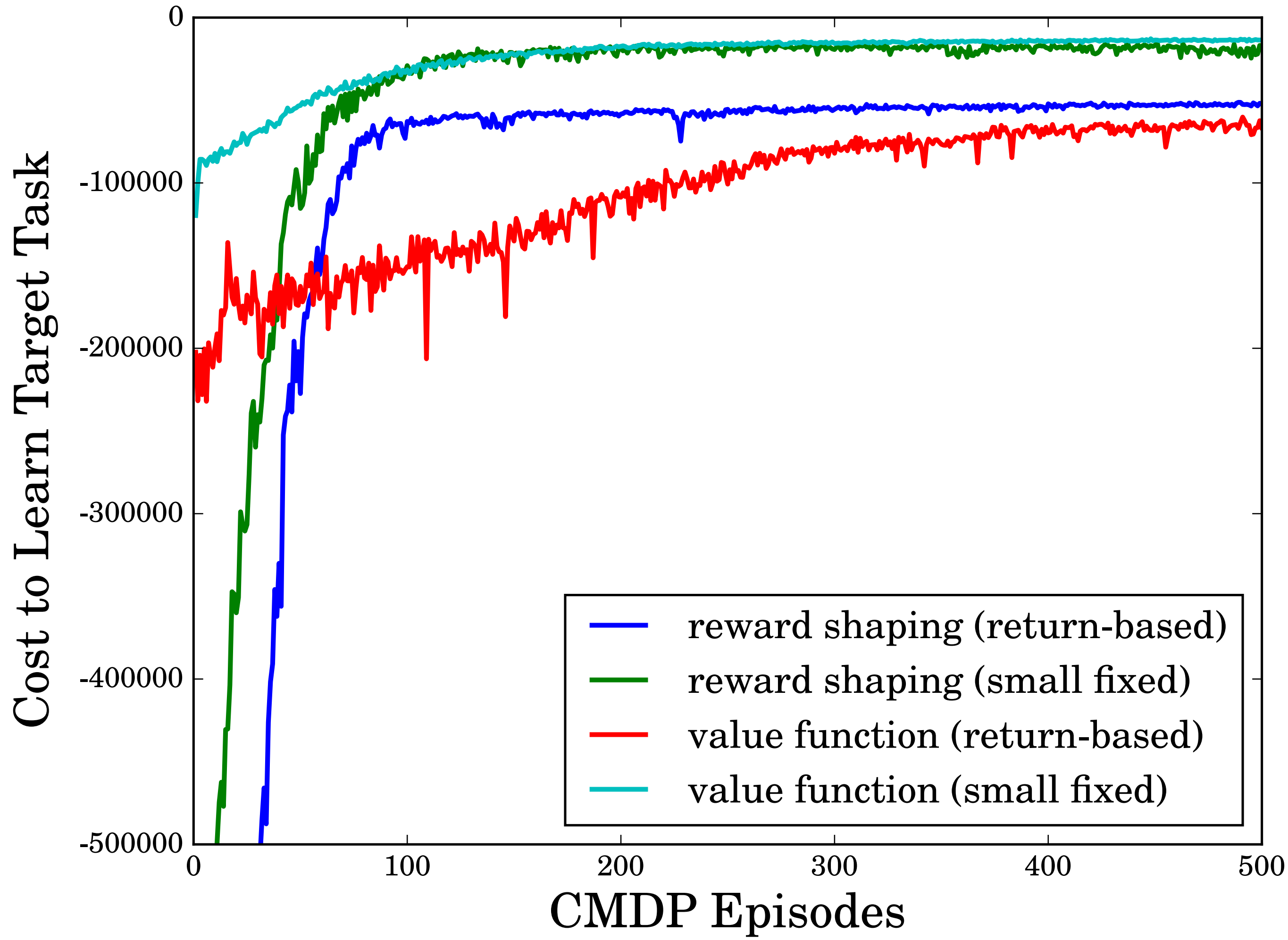
Pacman Value Function Transfer



Pacman Reward Shaping Transfer



How long to train?



Related Work

Restrictions on source tasks

- Florensa et al. 2018, Riedmiller et al. 2018, Sukhbaatar et al. 2017

Heuristic based sequencing

- Da Silva et al. 2018, Svetlik et al. 2017

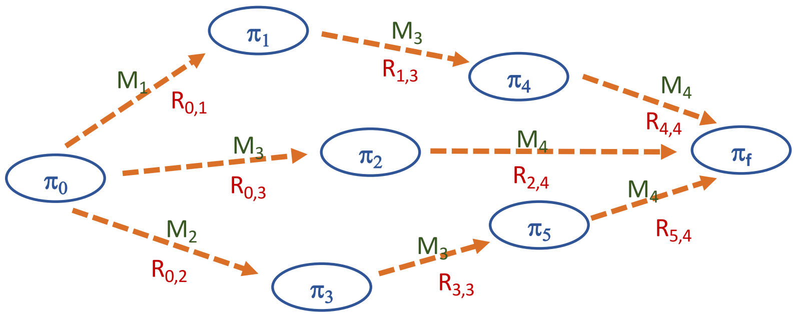
MDP/POMDP based sequencing

- Matiisen et al. 2017, Narvekar et al. 2017

CL for supervised learning

- Bengio et al. 2009, Fan et al. 2018, Graves et al. 2017

Summary



- Generalize/Formulate curriculum generation as an MDP

- Demonstrate curriculum policies can be learned, and is robust to:
  - Learning agent state/action representation
  - CMDP representations
  - Transfer algorithm used