A Simple Approach to Continual Learning By Transferring Skill Parameters

K.R. Zentner, Ryan Julian, Ujjwal Puri,
Yulun Zhang, and Gaurav S. Sukhatme

2021

Bardia Mojra
January 26, 2022
Robotic Vision Lab
University of Texas at Arlington

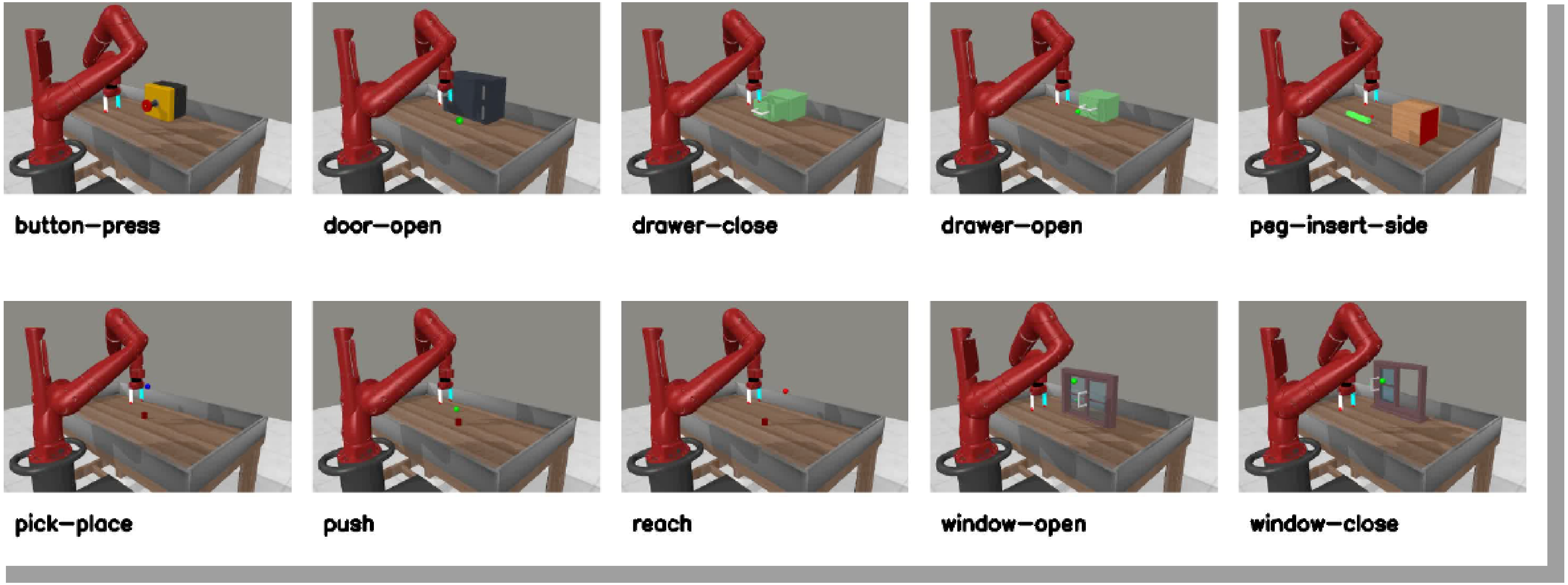
UNIVERSITY OF TEXAS
ROBOTIC
VISION

Introduction

- Problem:
  - existing transfer learning methods are too inefficient for vision-motor tasks (robotic manipulation)
  - Storing past experiences entirely, takes too much space and recall time
- Proposed:
  - Store previous knowledge only in terms of skill policies (parameters)
  - Use general purpose latent state skill representations for decomposed skills
  - Construct more complex skills based on simpler and prior skills
  - Use the propose optimal curriculum learning method for efficient transfer

Meta-World

- Web-based simulation environment for training DRL models for robotic tasks



Transfer Learning

- Defined as learning by transferring knowledge (known skills) from another agent
- Problems:
  - Traceability problem
  - Reward function estimation and tuning problem
  - Bad transfer

Continual Learning

- Defined as learning new skills either through real-world experimentation or transferring and re-tuning in simulation
- Problems:
  - Multi-task learning: could destroy prior skill (pessimal example)
  - Catastrophic forgetting
  - Current methods are too inefficient

Reusable Skill Libraries (DRL-Based Models)

- Associative Skill Memories
- Probabilistic Movement Primitive
- Latent space parameter decomposition

Continual Learning with Skill Libraries and Curricula

- Learn skills in form of factorized policy model classes
- Train an online model-based planner for reusing skills with high level action space (domain) for hierarchical RL
- Use on-policy RL to directly update skills

Setting

- T: a possibly-unbounded discrete space of tasks.
- S: a single continuous state space shared among all tasks T.
- A: a single continuous action space shared among all tasks T.
- The MTRL problem: (T , S, A), and each task τ ∈ T is an infinite-horizon Markov decision process (MDP)

$$\tau = (S, A, M_i, p_{\tau_i}(s, a, s'), r_{\tau_i}(s, a, s')) \,,$$

- M_i: represents the set is manipulation skills the robot is initialized or pre-trained with
- i: ith the epoch number

Simple Continual Learning with Skill Transfer

Algorithm 1 Proposed Continual Learning Framework

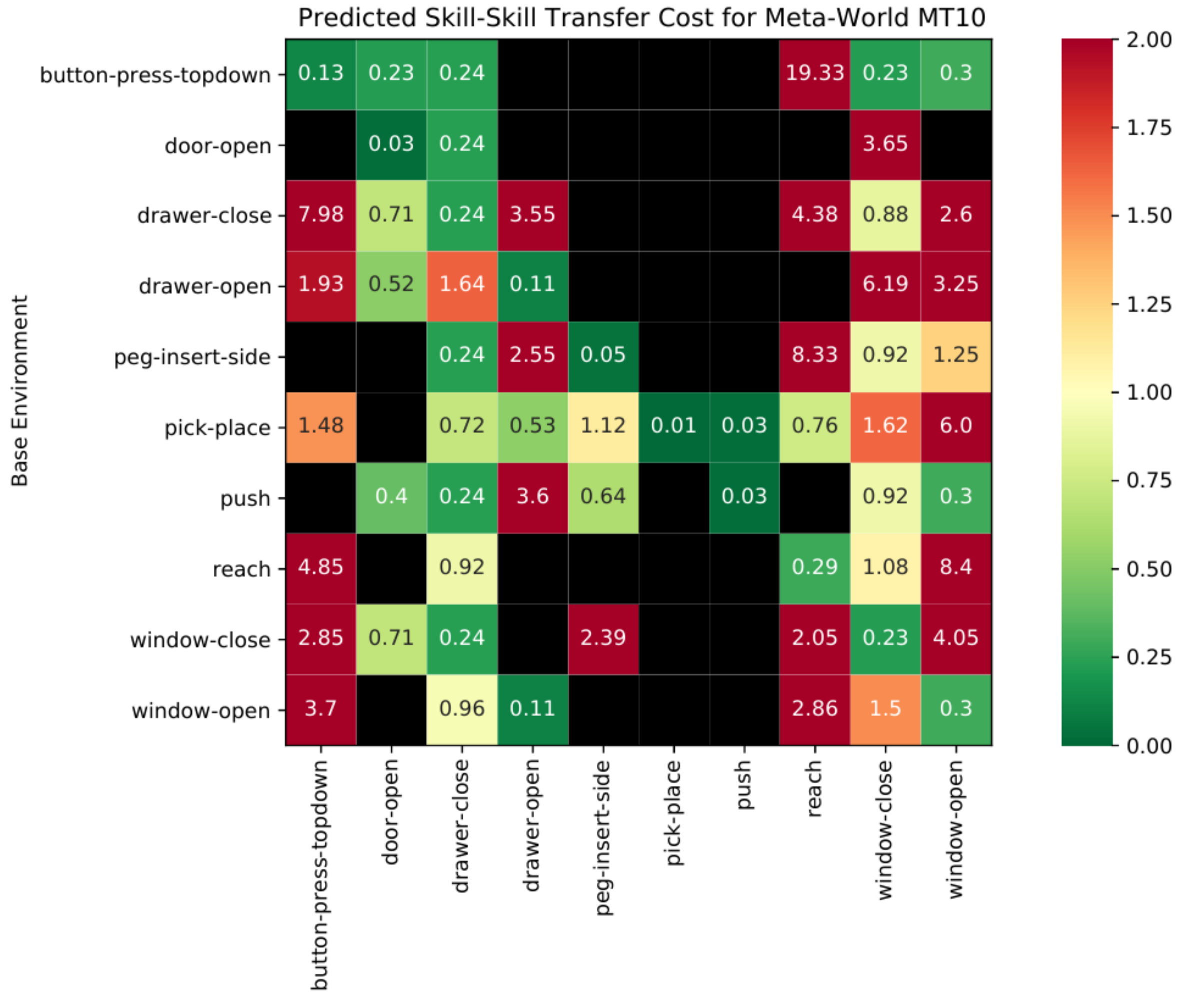
Input: Initial skill library $\mathcal{M}_0$, target task space $\mathcal{T}$, RL algorithm $\mathcal{F} \rightarrow (\pi, \rho)$, target task rule `ChooseTargetTask`, base skill rule `ChooseBaseSkill`
$i \leftarrow 1$
while not done **do**
  $\tau \leftarrow \texttt{ChooseTargetTask}(\mathcal{T}, \mathcal{M}_{i-1})$
  while π_{target} not solved **do**
    $\pi_{base} \leftarrow \texttt{ChooseBaseSkill}(\mathcal{T}, \mathcal{M}_{i-1})$
    $\pi_{target}, \cdot \leftarrow \mathcal{F}(\tau, \texttt{clone}(\pi_{base}))$
  end while
  $\mathcal{M}_i \leftarrow \{\pi_{target}\} \cup \mathcal{M}_{i-1}$
  $i \leftarrow i + 1$
end while
Output: Skill library $\mathcal{M}_i$

Other Techniques

- Warm-up procedure for value function transfer
- Rejecting bad transfers
- Skill-skill transfer cost
  - Number of samples needed to acquire target skill
- Predicted skill-skill transfer cost

$$A_{base \to target} = \frac{C_{base \to target}}{C_{scratch \to target}}$$

Predicted Skill-Skill Transfer Cost

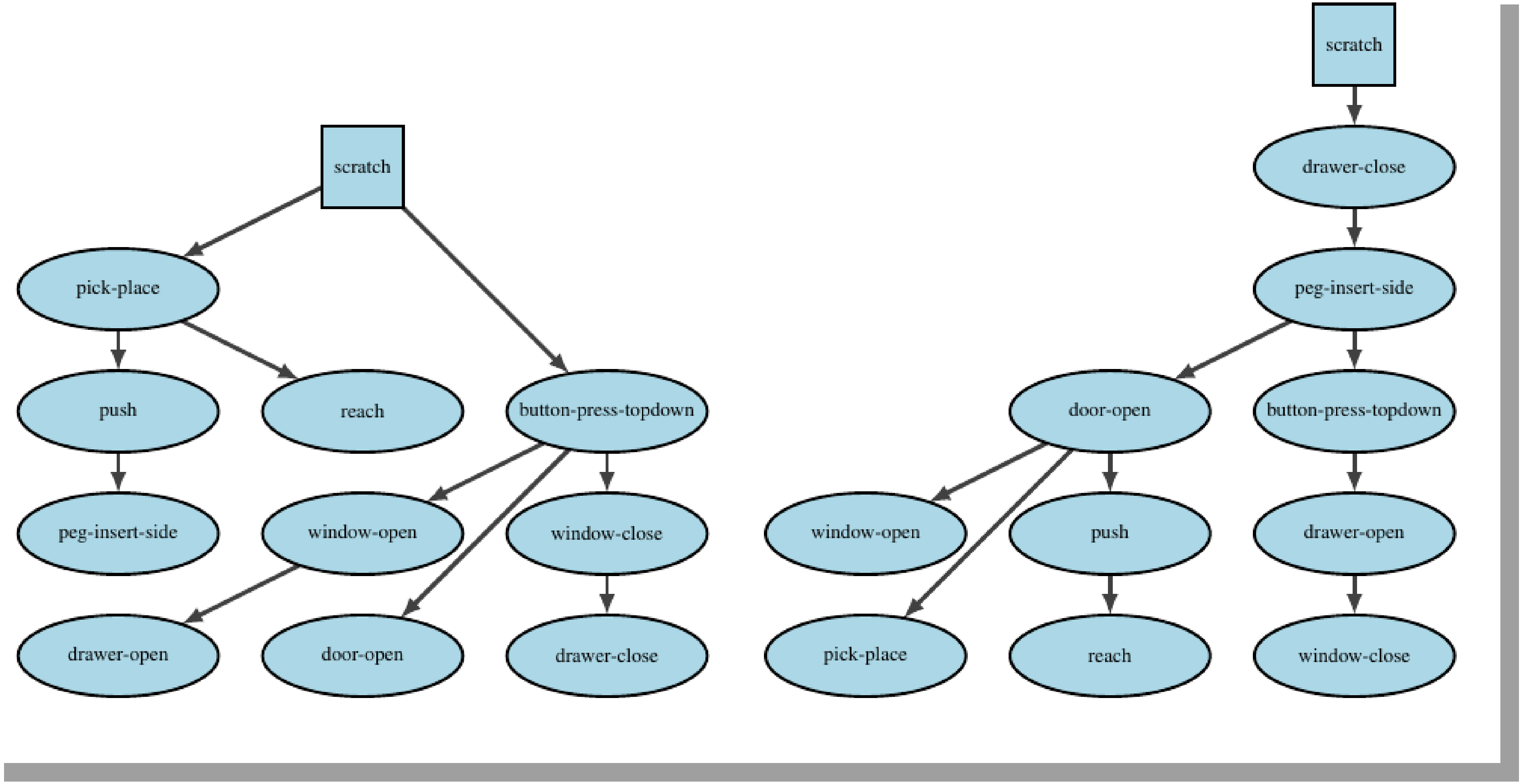


Curriculum Selection Algorithm

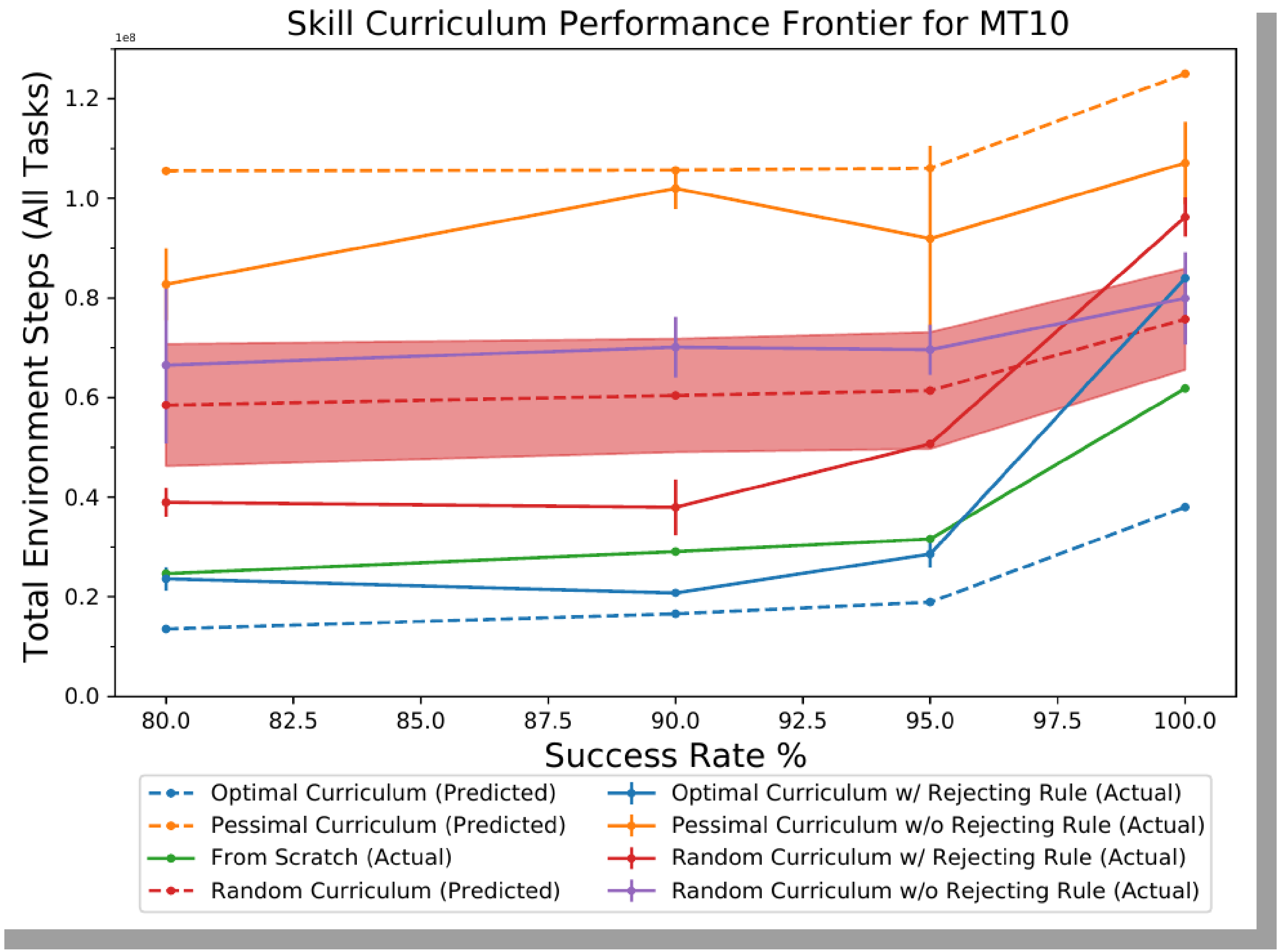
Algorithm 2 DMST-Based Curriculum Transfer

Input: Initial skill library $\mathcal{M}_0$, target task space $\mathcal{T}$, RL algorithm $\mathcal{F} \rightarrow (\pi, C)$
$V \leftarrow \mathcal{T} \cup scratch$
$E \leftarrow \{\}$
for $\tau_{base} \in \mathcal{T}$ **do**
  $E \leftarrow (scratch, \tau_{base}, -1.0)$
  $\pi_{base}, C_{scratch \rightarrow target} \leftarrow \mathcal{F}(\tau_{base}, \pi_{random})$
  for $\tau_{target} \in \mathcal{T}$ **do**
    $\cdot, C_{base \rightarrow target} = \mathcal{F}(\tau_{target}, \pi_{base})$
    $E \leftarrow E(\tau_{base}, \tau_{target}, C_{base \rightarrow target}) \cup E$
  end for
end for
$T_{optimal} \leftarrow \texttt{kruskal}((V, E))$
$i \leftarrow 1$
$\pi_{base} \leftarrow \pi_{random}$
for $\tau_{target} \in \texttt{traverse}(T_{optimal})$ **do**
  while τ_{target} not solved **do**
    $\pi_{target}, \cdot \leftarrow \mathcal{F}(\tau_{target}, \texttt{clone}(\pi_{base}))$
    if τ_{target} not solved **then**
      $E \leftarrow E \setminus (\tau_{base}, \tau_{target})$
      $T_{optimal} \leftarrow \texttt{kruskal}((V, E))$
    end if
  end while
  $\mathcal{M}_i \leftarrow \{\pi_{target}\} \cup \mathcal{M}_{i-1}$
  $i \leftarrow i + 1$
end for
Output: Skill library $\mathcal{M}$

Optimal and Pessimal Curricula



Experiments



- Thank you!