# PRIMA: Planner-Reasoner Inside a Multi-task Reasoning Agent

Daoming Lyu Auburn University, Auburn,
Bo Liu Auburn University, Auburn,

13 Feb 2022

Submitted to

Sir Muhammad Ata Ur Rehman

Group Members

- Munim Siddiqui (20K-1730)
- Murtaza Salman (20K-047
- Ali Qureshi (20K-0149)

# Contents

# Introduction to Multi-task Learning (MTL)

- **MTL** is an approach that improves sample complexity by learning to solve multiple tasks simultaneously, as opposed to the traditional "one model per task" strategy.
- Enhances AI's efficiency and predictive accuracy by leveraging the commonalities across tasks
- A significant challenge of MTL is the management of its model size that is larger than the single-task models.
- **Machine Task Learning** struggles with large models due to the conflict between generalization and task-specific efficiency.
  - **Generalization:** refers to a model's ability to perform well on unseen data i.e data not used during training.
  - **Task-Specific Efficiency:** Concerns how well a model can perform a specific task
- **PRIMA architecture**, trained using deep reinforcement learning, achieves a balance between capability and efficiency in MTL, yielding state-of-the-art performance.

# Background

- **MTL Evolution:** Demonstrates improved generalization by leveraging shared knowledge across tasks.
- **Deep Learning Integration:** Incorporates deep neural networks to enhance MTL's success and applicability.
- **Inherent Challenges:** Balances model size and task-specific efficiency while preserving generalization.
- **Logic Reasoning:** Employs First-Order Logic for deducing new information from known facts in AI.
- **PRIMA Framework:** Introduces a approach to efficiently balance reasoning capabilities in MTL.

# Problem Formulation

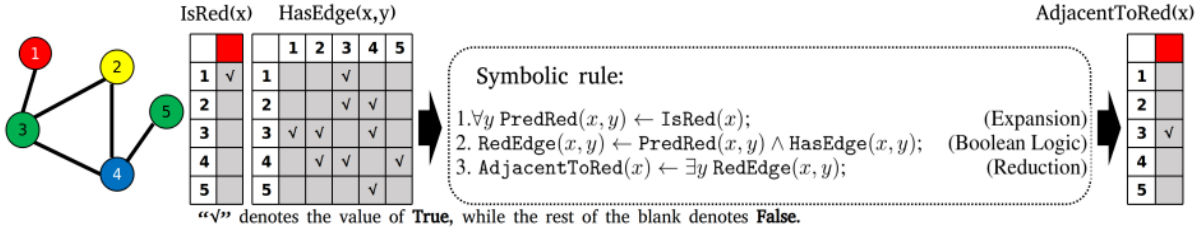



Figure 1: An example from the `AdjacentToRed` task and its formulation as a logical reasoning problem.

- **Simplification of First-Order Logic (FOL):** Focuses on variables, constants, and predicates to define relationships as True or False.
- **Construction of Logical Formulas:** Describes the construction of logical expressions using atoms, connectives, and quantifiers.
- **Definition of Multi-task Reasoning (MTR):** Aims to predict outputs across tasks, emphasizing joint learning of multiple reasoning tasks.
- **Objective of MTR:** Seeks to enhance task performance by applying reasoning skills across various tasks in a unified model.

# Limitations of Current Multi-task Learning Models

- **Generalization vs. Efficiency Conflict**: Current MTL models struggle to balance broad generalization across tasks with the need for efficient, task-specific execution.
- **Model Size Dilemma:** The large size of conventional MTL models poses challenges  which can slow them down and make them less effective at specific tasks..
- **Capability Overreach:** By trying to do too much, MTL models might not do any one task particularly well.
- **Not flexible enough:** These models can't easily adjust to the specific needs of each task, which can make them less efficient overall.

## 3. Model Architecture of PRIMA

- PRIMA helps us reason about multiple things at once.
- We use a strategy called forward-chaining to figure out answers step by step.
- But it's too slow to consider every possibility, so we need to be smarter about how we do it.
- PRIMA has two main parts: the Reasoner and the Planner.
- The Reasoner figures out logical rules using neural networks.
- The Planner decides which rules to use and how to put them together.
- We train PRIMA to be good at reasoning efficiently using deep reinforcement learning.

## 3.1 Reasoner: transforming logic rules into neural operators

- The Reasoner is like a smart assistant that understands logic and helps us find answers.
- It uses neural networks to translate logical rules into something it can understand.
- It focuses on three main rules: Boolean Logic, Expansion, and Reduction, which help it make sense of logical statements.

# 3.2 Planner: activating and forward-chaining learnable Horn clauses

- The Planner is like a strategist that plans how to solve problems efficiently.
- It decides which logical rules to use and in what order, sort of like picking the best path through a maze.
- Instead of trying every possible combination, it's smart and picks the most promising routes.
- This way, it finds solutions faster without getting stuck in unnecessary details.

# 4. Learning-to-Reason via Reinforcement Learning

- We're using reinforcement learning (RL) to teach a computer to reason logically.
- The Planner makes decisions about which neural operators to use based on binary indicators.
- This process is similar to making sequential decisions in Markov decision processes (MDPs)

# 4.1 An MDP formulation of the learning-to-reason problem

- **MDP Definition**: Think of an MDP as a framework with states, actions, transition probabilities, rewards, and a discount factor.
- **In Our Context**: The states represent sets of predicates, actions are binary indicators for activating neural operators, and rewards are based on reasoning accuracy.
- **Objective:** We want to find the best policy (sequence of actions) that maximizes the total expected reward over a series of reasoning steps.

# 4.2 Policy network: modeling of planner

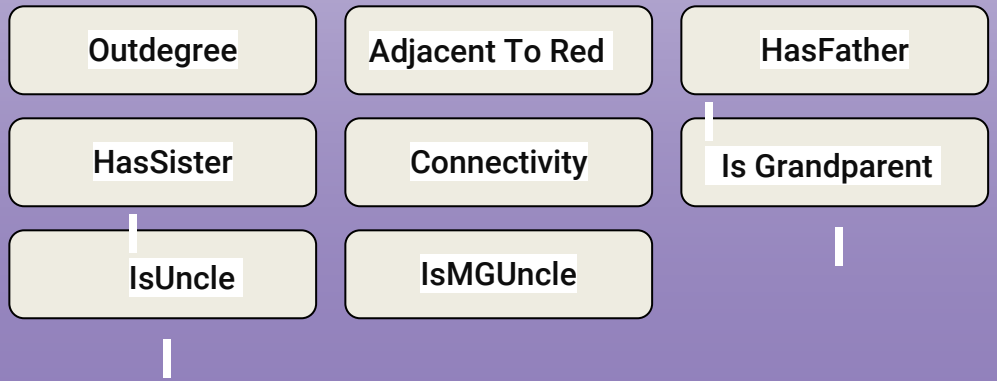
- **Planner Module**: It's like the brain deciding what to do. It's represented by a policy network.
- Base Planner: It's a simple version of the Planner, deciding which operators to activate based on input predicates.
- **Enhanced Planner**: This version uses Monte Carlo Tree Search (MCTS) to improve decision-making by exploring possible reasoning paths.

## 4.3 Overall learning framework

- We're using RL methods to teach the system to reason. We'll try REINFORCE and MuZero.
- **REINFORCE:** It's a model-free RL method, meaning it learns directly from experiences without a model of the environment.
- **MuZero:** It's a model-based RL method that combines planning and learning, and it's good for complex tasks.
- We use MuZero because it handles partial structural knowledge well, reducing testing complexity.
- **Training Process:** We load the model weights, execute reasoning path rollouts according to the policy network, and then train the Planner-Reasoner using the data collected.

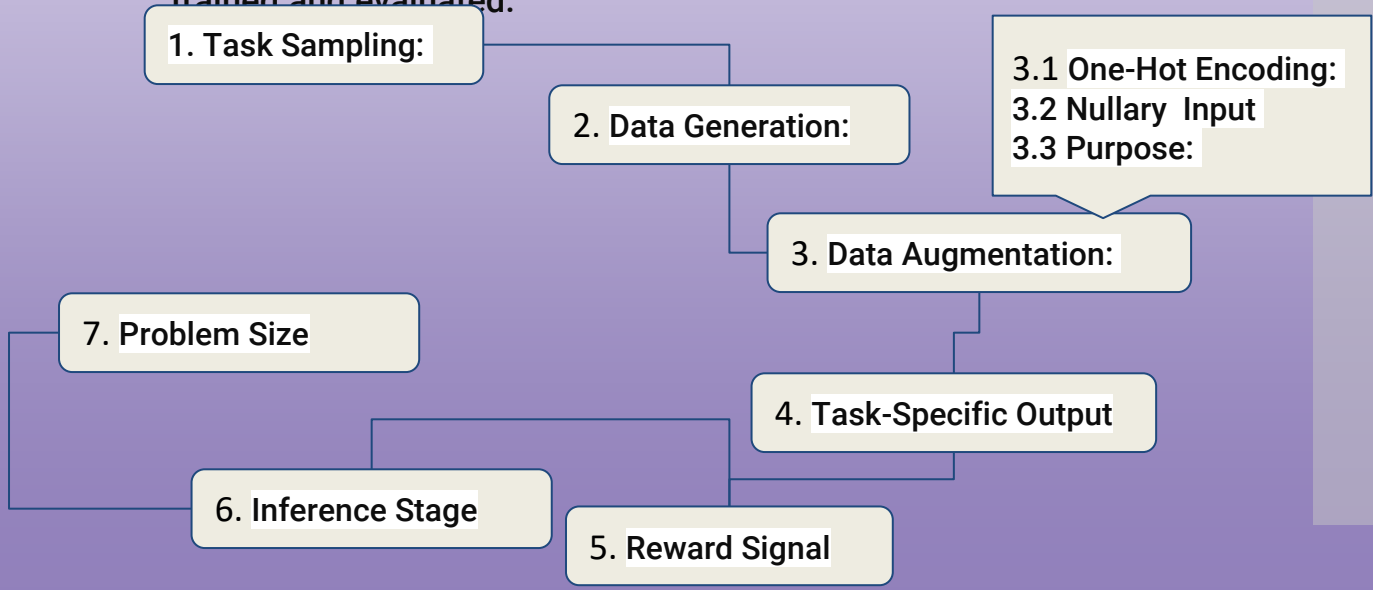
# 5 Experimental Results and Analysis

- Evaluate the performance of different variants of PRIMA on eight tasks from the family tree and graph benchmarks.
- Tasks are widely used benchmarks for inductive logic programming.
- Evaluate their testing accuracy and reasoning cost



# 5.1 Experimental Setups

- Methodology used for training and testing in the multi-task setting, particularly focusing on how tasks are sampled, data is generated, and the model is trained and evaluated.



## 5.2 Overall performance

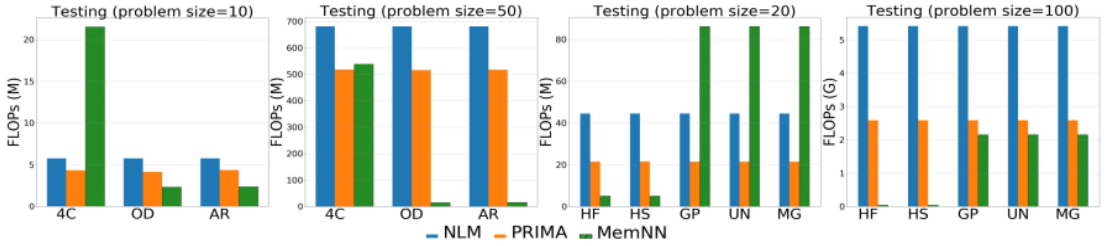
- **Single-task reasoning capability**
- Before we proceed to evaluate the MTR capabilities of PRIMA, We adapt to the single-task setting and train a separate model for each task. We name this

| testing accuracy | | | Family Tree | HasFather | HasSister | IsGrandparent | IsUncle | IsMGUncle | Graph | AdjacentToRed | 4-Connectivity | 1-OutDegree |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Single Task | PRISA-REINFORCE | | m=20 | 62.6 | 50.7 | 96.5 | 97.3 | 99.8 | m=10 | 47.7 | 33.5 | 48.7 |
| | | | m=100 | 87.8 | 69.8 | 2.3 | 97.7 | 98.4 | m=50 | 71.6 | 92.8 | 97.4 |
| | | | PSS | 0 | 0 | 0 | 0 | 0 | PSS | 0 | 0 | 0 |
| | PRISA-PPO | | m=20 | 71.5 | 64.3 | 97.5 | 98.1 | 99.6 | m=10 | 62.3 | 57.8 | 61.6 |
| | | | m=100 | 93.2 | 78.7 | 98.2 | 97.3 | 99.1 | m=50 | 85.5 | 95.2 | 96.3 |
| | | | PSS | 0 | 0 | 0 | 0 | 0 | PSS | 0 | 0 | 0 |
| | PRISA-MuZero | | m=20 | **100** | **100** | **100** | **100** | **100** | m=10 | **100** | **100** | **100** |
| | | | m=100 | **100** | **100** | **100** | **100** | **100** | m=50 | **100** | **100** | **100** |
| | | | PSS | **100** | **100** | **100** | **100** | **100** | PSS | 90 | **100** | **100** |

# 5.2 Overall performance

- **Multi-task reasoning capability**

| testing accuracy | | | Family Tree | HasFather | HasSister | IsGrandparent | IsUncle | IsMGUncle | Graph | AdjacentToRed | 4-Connectivity | 1-OutDegree |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Single Task | MemNN | m=20 | 99.9 | 86.3 | 96.5 | 96.3 | 99.7 | m=10 | 95.2 | 92.3 | 99.8 |
| | | | m=100 | 59.8 | 59.8 | 97.7 | 96 | 98.4 | m=50 | 93.1 | 81.3 | 78.6 |
| | | | PSS | **0** | **0** | **0** | **0** | **0** | PSS | **0** | **0** | **0** |
| | | ∂ILP/ NLM/ DLM | m=20 | 100 | 100 | 100 | 100 | 100 | m=10 | 100 | 100 | 100 |
| | | | m=100 | 100 | 100 | 100 | 100 | 100 | m=50 | 100 | 100 | 100 |
| | | | PSS | 100 | 100 | 100 | 100/ 90/ 100 | 100/ 20/ 70 | PSS | 100/ 90/ 90 | 100 | 100 |
| | Multi-Task | DLM-MTR | m=20 | 100 | 100 | 100 | 100 | 100 | m=10 | 96.7 | 100 | 100 |
| | | | m=100 | 100 | 100 | 100 | 100 | 100 | m=50 | 97.2 | 100 | 100 |
| | | | PSS | 100 | 100 | 100 | 100 | 100 | PSS | **0** | 100 | 100 |
| | | PRIMA | m=20 | 100 | 100 | 100 | 100 | 100 | m=10 | 100 | 100 | 100 |
| | | | m=100 | 100 | 100 | 100 | 100 | 100 | m=50 | 100 | 100 | 100 |
| | | | PSS | 100 | 100 | 100 | 100 | 90 | PSS | 90 | 100 | 100 |



## 6. Conclusion

- long-standing challenge in multi-task reasoning (MTR) is the intrinsic conflict between capability and efficiency
- development of a novel neural-logic architecture termed PRIMA
- PRIMA improves inference efficiency and achieves a state-of-the-art balance between MTR capability and inference efficiency.
- The model's training follows a complete data-driven end-to-end approach via deep reinforcement learning, and the performance is validated across a variety of benchmarks.

# ANY QUESTIONS?