1. Background and Problem Definition

**Classical Domain Independent Planning**

- **Given:**
  - A domain specifying the action model of every actions
  - An action model specifies actions' precond. and effects
  - A problem specifying the starting state and goal conditions

- **Output:** a plan, i.e., sequence of actions that achieve the goal

**Example:** The logistics domain
- Domain includes action model for move, pickup, & unload
- Problem can be "Package at location C" (At(Package, C))

Challenge: how to plan without a given action model?

---

2. SAM Learning for Grounded Domains

**Def:** an action model M is safe if an action A is applicable in state S iff:

- (1) A is applicable at S in the real action model (M*)
- (2) The effects of applying A in state S are

**Inference rules for grounded domains**

<table>
<thead>
<tr>
<th>Given:</th>
<th>Rule 1</th>
<th>Rule 2</th>
<th>Rule 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>saa's'</td>
<td><img src="image" alt="Inference rules for grounded domains" /></td>
<td><img src="image" alt="Inference rules for grounded domains" /></td>
<td><img src="image" alt="Inference rules for grounded domains" /></td>
</tr>
</tbody>
</table>

**SAM Learning:**
- For each action a
  - 1.1. Initialize pre(a) to be all literals
  - 1.2. Initialize eff(a) to be an empty set
  - 1.3. For each state-action-triplet (s,a,s')
    - 1.3.1. Apply Rule 1 to remove literals from pre(a)
    - 1.3.2. Apply Rule 3 to add literals to eff(a)

---

3. Lifted Domains in Classical Planning

A lifted domain defines
- A set of types (e.g., truck, package) and objects (truck, package_A)
- Lifted literals (e.g., At(x?, y?), On(x?, y?))

**Prior work:**
- Safe Action Model (SAM) Learning
  - Learns a safe action model Msa from trajectories
- Sound: plans generated by Msa are sound wrt M*

**Incompleteness:** Msa may be too weak to find a plan
- Does not generalize between instances of the same action

**Output:** a plan P for that is sound wrt M*

---

4. SAM Learning for Lifted Domains

**Inference rules for lifted domains**

<table>
<thead>
<tr>
<th>Given:</th>
<th>Rule 1</th>
<th>Rule 2</th>
<th>Rule 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>saa's'</td>
<td><img src="image" alt="Inference rules for lifted domains" /></td>
<td><img src="image" alt="Inference rules for lifted domains" /></td>
<td><img src="image" alt="Inference rules for lifted domains" /></td>
</tr>
</tbody>
</table>

**If b is injective,** this means [bindings(b, b)] = \{0,1\}

**Rule 3** becomes simply add (L, b, a) to list of effects

---

5. Theoretical Properties

Under the injective binding assumption, it holds that
- 1. SAM learning returns the strongest safe action model
- 2. Planning with the returned safe action model is sound but complete
- 3. Approximate completeness: the number of trajectories required to learn the action model is linear in the size of the domain model

**Thm 4:** Under the injective binding assumption, given \( m \geq \frac{1}{2}(2 \ln |\mathcal{P}| |\mathcal{L}|^k + |\mathcal{L}|) \) trajectories sampled from \( T_D \), then with probability at least \( 1 - \delta \) SAM learning for lifted domains (Algorithm 1) returns a safe action model \( M_{SAM} \) such that the probability of drawing from \( P_D \) a problem that is not solvable with \( M_{SAM} \) is at most \( \epsilon \).

---

6. Multiple Action Bindings

**What if A is not injective?**

- **bindings(b, b)** is a non-trivial set
- **Example:** consider a lifted action A(x, y, z)

- T1: \( \{A(o, a), A(o, a')\} \)
- T2: \( \{A(o, a), A(o, a')\} \)
- T3: \( \{A(o, a), A(o, a')\} \)

- \( \text{Either} \{L\} \text{is an effect or } \{L\} \) or both (Rule 3)
- \( \text{L} \) is not an effect (Rule 3)
- \( \text{L} \) is an effect!

---

7. Extended SAM Learning

- Generate a CNF for every lifted action describing knowledge of effects
- Create proxy actions to maintain the predictability of the effects.

**Example:** assume we learn for action A the following CNF:

\[ \text{(IsEff}(L)(1)(x) \text{ Or IsEff}(L)(1)(y)) \] and \[ \text{(IsEff}(L)(2)(x) \text{ Or IsEff}(L)(2)(y)) \]

The generated proxy actions are:

- \( A_{p-x-y}: \text{eff} = \{L(1)(x), \text{pre} = \{L(2)(2), L(2)(2)\}, \text{merge} x=y \}
- \( A_{p-x+y}: \text{eff} = \{L(1)(x), \text{pre} = \{L(1)(1), L(1)(1)\}, \text{merge} x=+y, z=w \)

---

8. Preliminary Experimental Results

- **Domains:** N-Puzzle (3x3 tiles, 3 predicates, 1 action), Blocks-world (8 blocks, 5 predicates, 4 actions)
- **Baseline:** FAMA (Almeida et al. 2019), outcome may be a unsafe

- **Results:**
  - N-Puzzle: both algorithms find the action model with a single (s,a,s') triplet.
  - Blocks-world: SAM learning able to find the action model with fewer trajectories

---

9. Summary and Future Work

**SAM learning** can learn lifted action models!
- Number of trajectories is query-linear in number of lifted actions
- Does not depend on the number of objects in the world!

**Direction for future work:**
- safe model-free planning in domains with continuous state variables, non-determinism, and partial observability.