SDRL: Interpretable and Data-efficient Deep Reinforcement Learning Leveraging Symbolic Planning

Daoming Lyu\textsuperscript{1}, Fangkai Yang\textsuperscript{2}, Bo Liu\textsuperscript{1}, Steven Gustafson\textsuperscript{3}

\textsuperscript{1}Auburn University, Auburn, AL, USA
\textsuperscript{2}NVIDIA, Redmond, WA, USA
\textsuperscript{3}Maana Inc., Bellevue, WA, USA
Sequential decision-making (SDM) concerns an agent making a sequence of actions based on its behavior in the environment.

Deep reinforcement learning (DRL) achieves tremendous success on sequential decision-making problems using deep neural networks (Mnih et al., 2015).
Challenge: Montezuma’s Revenge

- The avatar: climbs down the ladder, jumps over a rotating skull, **picks up the key (+100)**, goes back and uses the key to **open the right door (+300)**.
- Vanilla DQN achieves 0 score (*Mnih et al.*, 2015).
Challenge: Montezuma’s Revenge

Problem: long horizon sequential actions, sparse and delayed reward.

- poor data efficiency.
- lack of interpretability.
Solution: task decomposition

- Symbolic planning: subtasks scheduling (high-level plan).
- DRL: subtask learning (low-level control).
- Meta-learner: subtask evaluation.

Goal

- Symbolic planning drives learning, improving task-level interpretability.
- DRL learns feasible subtasks, improving data-efficiency.
Background: Symbolic Planning with Action Language

**Action language** *(Gelfond & Lifschitz, 1998)*: a formal, declarative, logic-based language that describes dynamic domains.

- Dynamic domains can be represented as a transition system.
Action Language $BC$ (Lee et al., 2013) is a language that describes the transition system using a set of causal laws.

- **dynamic laws** describe transition of states
  
  $\text{move}(x, y_1, y_2)$ causes $\text{on}(x, y_2)$ if $\text{on}(x, y_1)$.

- **static laws** describe value of fluents inside a state
  
  $\text{intower}(x, y_2)$ if $\text{intower}(x, y_1), \text{on}(y_1, y_2)$. 

\[\begin{align*}
\text{Problem} & \quad \rightarrow \quad \text{Logic Program} \\
\text{Modeling} & \quad \downarrow \quad \text{Solving} \\
\text{Solution} & \quad \rightarrow \quad \text{Stable Models (Answer Set)} \\
\text{Interpreting} & \quad \downarrow \quad \text{Grounder} \\
\text{Solver} & \quad \leftarrow \quad \text{Solver}
\end{align*}\]
Background: Reinforcement Learning

- **Reinforcement learning** is defined on a Markov Decision Process \((S, A, P_{ss'}, r, \gamma)\). To achieve optimal behavior, a policy \(\pi : S \times A \mapsto [0, 1]\) is learned.

- An **option** is defined on the tuple \((I, \pi, \beta)\), which enables the decision-making to have a hierarchical structure:
  - the initiation set \(I \subseteq S\),
  - policy \(\pi : S \times A \mapsto [0, 1]\),
  - probabilistic termination condition \(\beta : S \mapsto [0, 1]\).
Symbolic Planner: orchestrates sequence of subtasks using high-level symbolic plan.

Controller: uses DRL approaches to learn the subpolicy for each subtask with intrinsic rewards.

Meta-Controller: measures learning performance of subtasks, updates intrinsic goal to enable reward-driven plan improvement.
Symbolic Planner
Symbolic Planner: Planning with Intrinsic Goal

- **Intrinsic goal**: a linear constraint on plan quality
  
  \[ quality \geq quality(\Pi_t) \] 
  where \( \Pi_t \) is the plan at episode \( t \).

- Plan quality: a utility function
  
  \[ quality(\Pi_t) = \sum_{\langle s_{i-1}, g_{i-1}, s_i \rangle \in \Pi_t} \rho^{g_i-1}(s_{i-1}) \]

  where \( \rho^{g_i} \) is the gain reward for subtask \( g_i \).

- Symbolic planner: generates a new plan that
  
  - **explores** new subtasks,
  - **exploits** more rewarding subtasks.
From Symbolic Transition to Subtask

- Assumption: given the set $S$ of symbolic states and $\tilde{S}$ of sensory input, we assumed there is an *Oracle* for symbol grounding: $F : S \times \tilde{S} \mapsto \{t, f\}$.

- Given $F$ and a pair of symbolic states $s, s' \in S$:
  - initiation set $I = \{\tilde{s} \in \tilde{S} : F(s, \tilde{s}) = t\}$,
  - $\pi : \tilde{S} \mapsto \tilde{A}$ is the subpolicy for the corresponding subtask,
  - $\beta$ is the termination condition such that

$$
\beta(\tilde{s}') = \begin{cases} 
1 & F(s', \tilde{s}') = t, \text{ for } \tilde{s}' \in \tilde{S}, \\
0 & \text{otherwise.}
\end{cases}
$$
Controllers: DRL with Intrinsic Reward

- **Intrinsic reward**: pseudo-reward crafted by the human.
- Given a subtask defined on \((I, \pi, \beta)\), intrinsic reward

\[
 r_i(\tilde{s}') = \begin{cases} 
 \phi & \beta(\tilde{s}') = 1 \\
 r & \text{otherwise} 
\end{cases}
\]

where \(\phi\) is a positive constant encouraging achieving subtasks and \(r\) is the reward from the environment at state \(\tilde{s}'\).
Meta-Controller: Evaluation with Extrinsic Reward

- **Extrinsic rewards**: \( r_e(s, g) = f(\epsilon) \) where \( \epsilon \) can measure the competence of the learned subpolicy for each subtask.

  - For example, let \( \epsilon \) be the **success ratio**, \( f \) can be defined as

  \[
  f(\epsilon) = \begin{cases} 
  -\psi & \epsilon < \text{threshold} \\
  r(s, g) & \epsilon \geq \text{threshold}
  \end{cases}
  \]

  - \( \psi \) is a positive constant to punish selecting unlearnable subtasks,
  - \( r(s, g) \) is the cumulative environmental reward by following the subtask \( g \).
Experimental Results I.

% object declaration
location(mp;rd;ls;lll;lrl;key).

% dynamic causal law declaration
move(L) causes loc=L if location(L).
move(L) causes cost=1+2 if rho((at(Ll)),move(L))=Z,
loc=Ll,picked(key)=false.
move(L) causes cost=1+2 if rho((at(Ll),picked(key)),
move(L))=Z,loc=Ll,picked(key)=true.

inertial loc, inertial quality.
% static causal law declaration
picked(key)=true if loc=key.
noneexecutable move(key) if picked(key).
default rho((at(Ll)),move(L))=10.
default rho((at(Ll),picked(key)),move(L))=10.

<table>
<thead>
<tr>
<th>No.</th>
<th>subtask</th>
<th>policy learned</th>
<th>in optimal plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MP to LRL, no key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>LRL to LLL, no key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3</td>
<td>LLL to key, no key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>LRL to LLL, with key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>5</td>
<td>LLL to LRL, with or without key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>6</td>
<td>LRL to MP, with or without key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>7</td>
<td>MP to RD, with key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>8</td>
<td>LRL to LS, with or without key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>9</td>
<td>LS to key, with or without key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>10</td>
<td>MP to RD, no key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>11</td>
<td>LRL to key, with or without</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>12</td>
<td>key to LRL, with key</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>13</td>
<td>LRL to RD, with key</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

MP: middle platform
LRL: lower right ladder
LLL: lower left ladder
KEY: key
LS: left of rotating skull
RD: right door
Experimental Results II.

We present a **SDRL** framework features:

- High-level symbolic planning based on intrinsic goal
- Low-level policy control with DRL.
- Subtask learning evaluation by a meta-learner.

This is the first work on integrating symbolic planning with DRL that achieves both **task-level interpretability** and **data-efficiency** for decision-making.

Future work.