Appendix A Proofs

Proofs of Theorem 1, 2 and Corollary 1 can be found in the supplimentary material of (Lee and Wang 2018).

A.1 Propositions and Lemmas

We write $\langle a_0, a_1 \ldots, a_{m-1}\rangle^t$ (each $a_i \in \mathbf{A}$) to denote the formula $0\!:\!a_0 \wedge 1\!:\!a_1 \cdots \wedge m-1\!:\!a_{m-1}$. The following lemma tells us that any action sequence has the same probability under $Tr(D,m)$.

For any multi-valued probabilistic program Π, let $pf_1, \ldots, pf_m$ be the probabilistic constants in Π, and $v_{i,1}, \ldots, v_{i,k_i}$, each associated with probability $p_{i,1}, \ldots, p_{i,k_i}$ resp. be the values of pf_i ($i \in \{1, \ldots, m\}$). We use TC_Π be the set of all assignments to probabilistic constants in Π.

Lemma 1

For any $p\mathcal{BC}+$ action description D and any action sequence $\mathbf{a} = \langle a_0, a_1, \ldots, a_{m-1}\rangle$, we have

$$P_{Tr(D,m)}(\mathbf{a}^t) = \frac{1}{(|\sigma^{act}|+1)^m}.$$

Proof.

$$\begin{aligned}
&P_{Tr(D,m)}(\mathbf{a}^t)\\
=& \sum_{\substack{I \models \mathbf{a}^t \\ I \text{ is a stable models of } Tr(D,\,m)}} P_{Tr(D,m)}(I)\\
=& \text{ (In } Tr(D,m) \text{ every total choice leads to } (|\sigma^{act}|+1)^m \text{ stable models. By Proposition 2 in (Lee and Wang 2018),)}\\
& \sum_{\substack{I \models \mathbf{a}^t \\ I \text{ is a stable models of } Tr(D,\,m)}} \frac{W_{Tr(D,m)}(I)}{(|\sigma^{act}|+1)^m}\\
=& \frac{\sum_{tc \in TC_{Tr(D,m)}} \prod_{c=v \in tc} M_\Pi(c=v)}{(|\sigma^{act}|+1)^m}\\
=& \text{ (Derivations same as in the proof of Proposition 2 in (Lee and Wang 2018))}\\
& \frac{1}{(|\sigma^{act}|+1)^m}
\end{aligned}$$

□

The following lemma states that given any action sequence, the probabilities of all possible state sequences sum up to 1.

Lemma 2

For any $p\mathcal{BC}+$ action description D and any action sequence $\mathbf{a} = \langle a_0, a_1, \ldots, a_{m-1}\rangle$, we have

$$\sum_{s_0,\ldots,s_m : s_i \in \mathbf{S}} P_{Tr(D,m)}(\langle s_0, \ldots, s_m\rangle^t \mid \mathbf{a}^t) = 1.$$

Proof.

$$
\begin{aligned}
&\sum_{s_0,\ldots,s_m:s_i\in\mathbf{S}} P_{Tr(D,m)}(\langle s_0,\ldots,s_m\rangle^t \mid \mathbf{a}^t)\\
&= \text{(By Corollary 1 in (Lee and Wang 2018))}\\
&\sum_{s_0,\ldots,s_m:s_i\in\mathbf{S} i\in\{0,\ldots,m-1\}} \prod p(s_i,a_i,s_{i+1})\\
&= \sum_{s_0\in\mathbf{S}} (p(s_0)\cdot \sum_{s_1,\ldots,s_m:s_i\in\mathbf{S} i\in\{1,\ldots,m-1\}} \prod p(s_i,a_i,s_{i+1}))\\
&= \sum_{s_0\in\mathbf{S}} (p(s_0)\cdot \sum_{s_1\in\mathbf{S}} (p(s_0,a_0,s_1)\cdot \sum_{s_2,\ldots,s_m:s_i\in\mathbf{S} i\in\{2,\ldots,m-1\}} \prod p(s_i,a_i,s_{i+1})))\\
&= \sum_{s_0\in\mathbf{S}} (p(s_0)\cdot \sum_{s_1\in\mathbf{S}} (p(s_0,a_0,s_1)\cdot \cdots \cdot \sum_{s_m\in\mathbf{S}} p(s_{m-1},a_i,s_m)\ldots))\\
&= 1.
\end{aligned}
$$

□

The following proposition tells us that the probability of any state sequence conditioned on the constraint representation of a policy π coincide with the probability of the state sequence conditioned on the action sequence specified by π w.r.t. the state sequence.

Proposition 4

For any $p\mathcal{BC}+$ action description D, state sequence $\mathbf{s} = \langle s_0, s_1, \ldots, s_m\rangle$, and a non-stationary policy π, we have

$$
\begin{aligned}
&P_{Tr(D,m)}(\mathbf{s}^t \mid C_{\pi,m}) =\\
&P_{Tr(D,m)}(\mathbf{s}^t \mid \langle \pi(s_0,0), \pi(s_1,1), \ldots, \pi(s_{m-1},m-1)\rangle^t)
\end{aligned}
$$

Proof.

$$
\begin{aligned}
&P_{Tr(D,m)}(\mathbf{s}^t \mid C_{\pi,m})\\
&= \frac{P_{Tr(D,m)}(\langle s_0,\ldots,s_m\rangle^t \wedge C_{\pi,m})}{P_{Tr(D,m)}(C_{\pi,m})}\\
&= \frac{P_{Tr(D,m)}(\langle s_0, \pi(s_0,0)\ldots,\pi(s_{m-1},m-1), s_m\rangle^t}{P_{Tr(D,m)}(C_{\pi,m})}\\
&= \frac{P_{Tr(D,m)}(\langle \pi(s_0,0)\ldots,\pi(s_{m-1},m-1), s_m\rangle^t \mid 0{:}s_0)\cdot P_{Tr(D,m)}(0{:}s_0)}{\sum_{s'_0,\ldots,s'_m:s'_i\in\mathbf{S}} P_{Tr(D,m)}(\langle s'_0, \pi(s'_0,0)\ldots,\pi(s'_{m-1},m-1), s'_m\rangle^t)}.
\end{aligned}
$$

We use $k(s_0,\ldots,s_m)$ as an abbreviation of

$$P_{Tr(D,m)}(\langle \pi(s_0,0),\ldots,\pi(s_{m-1},m-1)\rangle^t).$$

We have

$$
\begin{aligned}
& P_{Tr(D,m)}(\mathbf{s}^t \mid C_{\pi,m}) \\
= & \frac{P_{Tr(D,m)}(\langle s_1,\ldots,s_m\rangle^t \mid \langle s_0,\pi(s_0,0),\ldots,\pi(s_{m-1},m-1)\rangle^t)\cdot P_{Tr(D,m)}(0{:}s_0)\cdot k(s_0,\ldots,s_m)}{\sum_{s'_0,\ldots,s'_m:s'_i\in\mathbf{S}} P_{Tr(D,m)}(\langle s'_1,\ldots,s'_m\rangle^t \mid \langle s'_0,\pi(s'_0,0),\ldots,\pi(s'_{m-1},m-1)\rangle^t)\cdot P_{Tr(D,m)}(0{:}s'_0)\cdot k(s'_0,\ldots,s'_m)} \\
= & \text{ (By Lemma 1, for any } s_0,\ldots,s_m(s_i\in\mathbf{S}), \text{ we have } k(s_0,\ldots,s_m)=\frac{1}{(\sigma^{act}|+1)^m}) \\
& \frac{P_{Tr(D,m)}(\langle s_1,\ldots,s_m\rangle^t \mid \langle s_0,\pi(s_0,0),\ldots,\pi(s_{m-1},m-1)\rangle^t)\cdot P_{Tr(D,m)}(0:s_0)\cdot \frac{1}{(|\sigma^{act}|+1)^m}}{\sum_{s'_0,\ldots,s'_m:s'_i\in\mathbf{S}} P_{Tr(D,m)}(\langle s'_1,\ldots,s'_m\rangle^t \mid \langle s'_0,\pi(s'_0,0),\ldots,\pi(s'_{m-1},m-1)\rangle^t)\cdot P_{Tr(D,m)}(0{:}s'_0)\cdot \frac{1}{(|\sigma^{act}|+1)^m}} \\
= & \frac{P_{Tr(D,m)}(\langle s_1,\ldots,s_m\rangle^t \mid \langle s_0,\pi(s_0,0),\ldots,\pi(s_{m-1},m-1)\rangle^t)\cdot P_{Tr(D,m)}(0:s_0)}{\sum_{s'_0,\ldots,s'_m:s'_i\in\mathbf{S}} P_{Tr(D,m)}(\langle s'_1,\ldots,s'_m\rangle^t \mid \langle s'_0,\pi(s'_0,0),\ldots,\pi(s'_{m-1},m-1)\rangle^t)\cdot P_{Tr(D,m)}(0{:}s'_0)} \\
= & \text{ (By Lemma 2, the denominator equals 1)} \\
& P_{Tr(D,m)}(\langle s_1,\ldots,s_m\rangle^t \mid \langle s_0,\pi(s_0,0),\ldots,\pi(s_{m-1},m-1)\rangle^t)\cdot P_{Tr(D,m)}(0:s_0) \\
= & P_{Tr(D,m)}(\langle s_0,s_1,\ldots,s_m\rangle^t \mid \langle \pi(s_0,0),\ldots,\pi(s_m-1,m-1)\rangle^t)
\end{aligned}
$$

□

A.2 Proofs of Proposition 2, Proposition 3, Theorem 3 and Theorem 4

The following proposition tells us that, for any states and actions sequence, any stable model of $Tr(D,m)$ that satisfies the sequence has the same utility. Consequently, the expected utility of the sequence can be computed by looking at any single stable model that satisfies the sequence.

Proposition 2 *For any two stable models X_1, X_2 of $Tr(D,m)$ that satisfy a history $\mathbf{h}=\langle s_0,a_0,s_1,a_1,\ldots,a_{m-1},s_m\rangle$, we have*

$$U_{Tr(D,m)}(X_1) \;=\; U_{Tr(D,m)}(X_2) \;=\; E[U_{Tr(D,m)}(\mathbf{h}^t)].$$

Proof. Since both X_1 and X_2 both satisfy $\mathbf{h}^t$, X_1 and X_2 agree on truth assignment on $\sigma_m^{act}\cup\sigma_m^{fl}$. Notice that atom of the form $\texttt{utility}(v,\mathbf{t})$ in $Tr(D,m)$ occurs only of the form (21), and only atom in $\sigma_m^{act}\cup\sigma_m^{fl}$ occurs in the body of rules of the form (21).

- Suppose an atom $\texttt{utility}(v,\mathbf{t})$ is in X_1. Then the body B of at least one rule of the form (21) with $\texttt{utility}(v,\mathbf{t})$ in its head in $Tr(D,m)$ is satisfied by X_1. B must be satisfied by X_2 as well, and thus $\texttt{utility}(v,\mathbf{t})$ is in X_2 as well.
- Suppose an atom $\texttt{utility}(v,\mathbf{t})$, is not in X_1. Then, assume, to the contrary, that $\texttt{utility}(v,\mathbf{t})$ is in X_2, then by the same reasoning process above in the first bullet, $\texttt{utility}(v,\mathbf{t})$ should be in X_1 as well, which is a contradiction. So $\texttt{utility}(v,\mathbf{t})$ is also not in X_2.

So X_1 and X_2 agree on truth assignment on all atoms of the form $\texttt{utility}(v,\mathbf{t})$, and conse-

quently we have $U_{Tr(D,m)}(X_1) = U_{Tr(D,m)}(X_2)$, as well as

$$\begin{aligned}
& E[U_{Tr(D,m)}(\mathbf{h}^t)] \\
&= \sum_{I \models \mathbf{h}^t} P_{Tr(D,m)}(I \mid \mathbf{h}^t) \cdot U_{Tr(D,m)}(I) \\
&= U_{Tr(D,m)}(X_1) \cdot \sum_{I \models \mathbf{h}^t} P_{Tr(D,m)}(I \mid \mathbf{h}^t) \\
&= \text{(The second term equals 1)} \\
& U_{Tr(D,m)}(X_1).
\end{aligned}$$

□

The following proposition tells us that the expected utility of an action and state sequence can be computed by summing up the expected utility from each transition.

Proposition 5

For any $p\mathcal{BC}+$ action description D and a history $\mathbf{h} = \langle s_0, a_0, s_1, \ldots, a_{m-1}, s_m \rangle$, such that there exists at least one stable model of $Tr(D, m)$ that satisfies $\mathbf{h}$, we have

$$E[U_{Tr(D,m)}(\mathbf{h}^t)] = \sum_{i \in \{0,\ldots,m-1\}} u(s_i, a_i, s_{i+1}).$$

Proof. Let X be any stable model of $Tr(D, m)$ that satisfies $\mathbf{h}^t$. By Proposition 2, we have

$$\begin{aligned}
& E[U_{Tr(D,m)}(\mathbf{h}^t)] \\
&= U_{Tr(D,m)}(X) \\
&= \sum_{i \in \{0,\ldots,m-1\}} \Big(\sum_{\substack{utility(v,i,\mathbf{x}) \leftarrow (i+1:F) \wedge (i:G) \in Tr(D,m) \\ X \text{ satisfies } (i+1:F) \wedge (i:G)}} v \Big) \\
&= \sum_{i \in \{0,\ldots,m-1\}} \Big(\sum_{\substack{utility(v,0,\mathbf{x}) \leftarrow (1:F) \wedge (0:G) \in Tr(D,m) \\ 0:X^i \text{ satisfies } (1:F) \wedge (0:G)}} v \Big) \\
&= \sum_{i \in \{0,\ldots,m-1\}} U_{Tr(D,1)}(0:X^i) \\
&= \text{(By Proposition 2)} \\
& \sum_{i \in \{0,\ldots,m-1\}} E[U_{Tr(D,1)}(0:s_i, 0:a_i, 1:s_{i+1})] \\
&= \sum_{i \in \{0,\ldots,m-1\}} u(s_i, a_i, s_{i+1}).
\end{aligned}$$

□

Proposition 3 *Given any initial state s_0 that is consistent with D_{init}, for any policy π, we have*

$$\begin{aligned}
& E[U_{Tr(D,m)}(C_{\pi,m} \wedge \langle s_0 \rangle^t)] = \\
& \sum_{\mathbf{s} = \langle s_1,\ldots,s_m \rangle : s_i \in \mathbf{S}} R_D(\mathbf{h}_\pi(\mathbf{s})^t) \times P_{Tr(D,m)}(\mathbf{s}^t \wedge C_{\pi,m}).
\end{aligned}$$

Proof. We have

$$
\begin{aligned}
& E[U_{Tr(D,m)}(C_{\pi,m} \wedge \langle s_0 \rangle^t)] \\
= & \sum_{I \models 0{:}s_0 \wedge C_{\pi,m}} P_{Tr(D,m)}(I \mid 0{:}s_0 \wedge C_{\pi,m}) \cdot U_{Tr(D,m)}(I) \\
= & \sum_{\substack{I \models 0{:}s_0 \wedge C_{\pi,m} \\ I \text{ is a stable model of } Tr(D,m)}} P_{Tr(D,m)}(I \mid 0{:}s_0 \wedge C_{\pi,m}) \cdot U_{Tr(D,m)}(I) \\
= & \text{ (We partition stable models } I \text{ according to their truth assignment on } \sigma_m^{fl}) \\
& \sum_{\mathbf{s}=\langle s_1,\ldots,s_m \rangle : s_i \in \mathbf{S}} \; \sum_{\substack{I \models \mathbf{s}^t \wedge C_{\pi,m} \\ I \text{ is a stable model of } Tr(D,m)}} P_{Tr(D,m)}(I \mid 0{:}s_0 \wedge C_{\pi,m}) \cdot U_{Tr(D,m)}(I) \\
= & \text{ (Since } I \models \mathbf{s}^t \wedge C_{\pi,m} \text{ implies } I \models \mathbf{h}_\pi(\mathbf{s})^t \text{, by Proposition 2 we have)} \\
& \sum_{\mathbf{s}=\langle s_1,\ldots,s_m \rangle : s_i \in \mathbf{S}} \; \sum_{\substack{I \models \mathbf{s}^t \wedge C_{\pi,m} \\ I \text{ is a stable model of } Tr(D,m)}} P_{Tr(D,m)}(I \mid 0{:}s_0 \wedge C_{\pi,m}) \cdot E[U_{Tr(D,m)}(\mathbf{h}_\pi(s)^t)] \\
= & \sum_{\mathbf{s}=\langle s_1,\ldots,s_m \rangle : s_i \in \mathbf{S}} Pr_{Tr(D,m)}(\mathbf{s}^t \mid 0{:}s_0 \wedge C_{\pi,m}) \cdot E[U_{Tr(D,m)}(\mathbf{h}_\pi(s)^t)] \\
= & \sum_{\mathbf{s}=\langle s_1,\ldots,s_m \rangle : s_i \in \mathbf{S}} Pr_{Tr(D,m)}(\mathbf{s}^t \mid 0{:}s_0 \wedge C_{\pi,m}) \cdot E[U_{Tr(D,m)}(\mathbf{s}^t \wedge C_{\pi,m})] \\
= & \sum_{\mathbf{s}=\langle s_1,\ldots,s_m \rangle : s_i \in \mathbf{S}} R_D(\mathbf{h}_\pi(\mathbf{s})^t) \times P_{Tr(D,m)}(\mathbf{s}^t \wedge C_{\pi,m}).
\end{aligned}
$$

□

Theorem 3 *Given an initial state $s_0 \in \mathbf{S}$ that is consistent with D_{init}, for any non-stationary policy π and any finite state sequence $\mathbf{s} = \langle s_0, s_1, \ldots, s_{m-1}, s_m \rangle$ such that each s_i in $\mathbf{S}$ ($i \in \{0, \ldots, m\}$), we have*

- $R_D(\mathbf{h}_\pi(\mathbf{s})) = R_{M(D)}(\mathbf{h}_\pi(\mathbf{s}))$
- $P_{Tr(D,m)}(\mathbf{s}^t \mid \langle s_0 \rangle^t \wedge C_{\pi,m}) = P_{M(D)}(\mathbf{h}_\pi(\mathbf{s}))$.

Proof. We have

$$
\begin{aligned}
& R_D(\mathbf{h}_\pi(\mathbf{s})) \\
= & E[U_{Tr(D,m)}(\mathbf{s}^t \wedge C_{\pi,m})] \\
= & \text{ (By Proposition 5)} \\
& \sum_{i \in \{0,\ldots,m-1\}} u(s_i, \pi(s_i, i), s_{i+1}) \\
= & \sum_{i \in \{0,\ldots,m-1\}} R(s_i, \pi(s_i, i), s_{i+1}) \\
= & R_{M(D)}(\mathbf{h}_\pi(\mathbf{s}))
\end{aligned}
$$

and

$$\begin{aligned}
&P_{Tr(D,m)}(\mathbf{s}^t \mid \langle s_0\rangle^t \wedge C_{\pi,m})\\
=\ &\text{(By Proposition 4)}\\
&Pr_{Tr(D,m)}(\mathbf{s}^t \mid \mathbf{h}_\pi(\mathbf{s})^t)\\
=\ &\text{(By Corollary 1 in (Lee and Wang 2018))}\\
&\prod_{i\in\{0,\ldots,m-1\}} p(\langle s_i, \pi(s_i,i), s_{i+1}\rangle)\\
=\ &P_{M(D)}(\mathbf{h}_\pi(\mathbf{s}))
\end{aligned}$$

□

Theorem 4 *For any nonnegative integer m and an initial state $s_0 \in \mathbf{S}$ that is consistent with D_{init}, we have*

$$\operatorname*{argmax}_{\pi \text{ is a policy}} E[U_{Tr(D,m)}(C_{\pi,m} \wedge \langle s_0\rangle^t)] = \operatorname*{argmax}_{\pi} ER_{M(D)}(\pi, s_0).$$

Proof. We show that for any non-stationary policy π,

$$E[U_{Tr(D,m)}(C_{\pi,m} \wedge \langle s_0\rangle^t)] = ER_{M(D)}(\pi, s_0).$$

We have

$$\begin{aligned}
&E[U_{Tr(D,m)}(C_{\pi,m} \wedge \langle s_0\rangle^t)]\\
=\ &\text{(By Proposition 3)}\\
&\sum_{\mathbf{s}=\langle s_1,\ldots,s_m\rangle : s_i\in\mathbf{S}} R_D(\mathbf{h}_\pi(\mathbf{s})) \times P_{Tr(D,m)}(\mathbf{s}^t \mid \langle s_0\rangle^t \wedge C_{\pi,m}).\\
=\ &\text{(By Theorem 3)}\\
&\sum_{\mathbf{s}=\langle s_1,\ldots,s_m\rangle : s_i\in\mathbf{S}} R_{M(D)}(\mathbf{h}_\pi(\mathbf{s})) \cdot P_{M(D)}(\mathbf{h}_\pi(\mathbf{s}))\\
=\ &ER_{M(D)}(\pi, s_0).
\end{aligned}$$

□

Appendix B PBCPLUS2MDP System Description

We describe the exact procedure performed by PBCPLUS2MDP in Algorithm 2. PBCPLUS2MDP uses LPMLN2ASP, which is component of LP$^{\text{MLN}}$ 1.0 system (Lee et al. 2017), for exact inference to find states, actions, transition probabilities and transition rewards. PBCPLUS2MDP uses MDPTOOLBOX for solving the MDP generated from the input action description.

The input is the LP$^{\text{MLN}}$ translation $Tr(D,m)$ of a $p\mathcal{BC}+$ action description D, a time horizon T, and a discount factor γ. In the input LP$^{\text{MLN}}$ program, we use atoms of the form $\texttt{fl\_}x(v_1,\ldots,v_m,\texttt{t},i)$, (and $\texttt{fl\_}x(v_1,\ldots,v_m,\texttt{f},i)$) to encode fluent constant $x(v_1,\ldots,v_m)$ is true, (and false, resp.) at time step i. Similarly, action constants and pf constants are encoded with atoms with prefix `act_` and `pf_`, resp. $Tr(D,m)$ is parametrized with maximum step m, for executing with different settings of maximum time step. As an example, The LP$^{\text{MLN}}$ translation of the $p\mathcal{BC}+$ action description in Section 5 (robot and blocks) is listed in Appendix C.

Algorithm 2 PBCPLUS2MDP system

Input:

1. $Tr(D, m)$: A $p\mathcal{BC}+$ action description translated into LP^{MLN} program, parameterized with maxstep m, with states set $\mathbf{S}$ and action sets $\mathbf{A}$
2. T: time horizon
3. γ: discount factor

Output: Optimal policy
Procedure:

1. Execute LPMLN2ASP on $Tr(D, m)$ with $m = 0$ to obtain all stable models of $Tr(D, 0)$; project each stable model of $Tr(D, 0)$ to only atoms corresponding to fluent constant (marked by `fl_` prefix); assign a unique number $idx(s) \in \{0, \ldots, |\mathbf{S}| - 1\}$ to each of the projected stable model s of $Tr(D, 0)$;
2. Execute LPMLN2ASP on $Tr(D, m)$ with $m = 1$ and the clingo option `--project` to project stable models to only atoms corresponding to action constant (marked by `act_` prefix); assign a unique number $idx(a) \in \{0, \ldots, |\mathbf{A}| - 1\}$ to each of the projected stable model a of $Tr(D, 1)$;
3. Initialize 3-dimensional matrix P of shape $(|\mathbf{A}|, |\mathbf{S}|, |\mathbf{S}|)$;
4. Initialize 3-dimensional matrix R of shape $(|\mathbf{A}|, |\mathbf{S}|, |\mathbf{S}|)$;
5. For each state $s \in \mathbf{S}$ and action $a \in \mathbf{A}$:
   (a) execute LPMLN2ASP on $Tr(D, m) \cup \{0 : s\} \cup \{0 : a\} \cup ST\_DEF$ with $m = 1$ and the option `-q "end_state"`, where ST_DEF contains the rule
   $$\{\texttt{end\_state}(idx(s)) \leftarrow 1 : s \mid s \in \mathbf{S}\}.$$
   (b) Obtain $P_{Tr(D,1)}(1 : s' \mid 0 : s, 0 : a)$ by extracting the probability of $P_{Tr(D,1)}(\texttt{end\_state}(idx(s')) \mid 0 : s, 0 : a)$ from the output;
   (c) $P(idx(a), idx(s), idx(s')) \leftarrow P_{Tr(D,1)}(1 : s' \mid 0 : s, 0 : a)$;
   (d) Obtain $E[U_{Tr(D,1)}(1 : s', 0 : s, 0 : a)]$ from the output by selecting an arbitrary answer set returned that satisfies $1 : s' \wedge 0 : s \wedge 0 : a$ and sum up the first arguments of all atoms with predicate name `utility` (By Proposition 2, this is equivalent to $E[U_{Tr(D,1)}(1 : s', 0 : s, 0 : a)]$).
   (e) $R(idx(a), idx(s), idx(s')) \leftarrow E[U_{Tr(D,1)}(1 : s', 0 : s, 0 : a)]$;
6. Call finite horizon policy optimization algorithm of PYMDPTOOLBOX with transition matrix P, reward matrix R, time horizon T and discount factor γ; return the output.

To construct the MDP instance $M(D) = \langle S, A, T, R \rangle$ corresponding to D, PBCPLUS2MDP constructs the set S of states, the set A of actions, transition probability function T and reward function R one by one.

By definition, states of D are interpretations I^{fl} of σ^{fl} such that $0 : I^{fl}$ are residual stable models of D_0. Thus, PBCPLUS2MDP finds the states of D by projecting the stable models of $Tr(D, 0)$ to atoms with prefix `fl_`. LPMLN2ASP is executed to find the stable models of $Tr(D, 0)$. The CLINGO option `--option` is used to project stable models to only atoms with `fl_` prefix. Similarly, PBCPLUS2MDP finds the actions of D by projecting the stable models of $Tr(D, 1)$ to atoms with prefix `act_`.

The transition probability function T and the reward function R are represented by three dimensional matrices, specifying the transition probability and transition reward for each transition $\langle s, a, s' \rangle$. Transition probabilities are obtained by computing conditional probabilities $P_{Tr(D,1)}(1 : s' \mid 0 : s, 0 : a)$ for every transition $\langle s, a, s' \rangle$, using LPMLN2ASP. Transition reward of each transition $\langle s, a, s' \rangle$ are obtained by computing the utility of any stable model of $Tr(D, 1)$ that satisfies $0 : s \wedge 0 : a \wedge 1 : s'$. This is justified by Proposition 2.

Finally, the constructed MDP instance $M(D)$, along with time horizon T and discount factor γ, is used as input to MDPTOOLBOX to find the optimal policy.

The system has the following dependencies:

- PYTHON 2.7

- CLINGO python library: `https://github.com/potassco/clingo/blob/master/INSTALL.md`
- LPMLN2ASP system: `http://reasoning.eas.asu.edu/lpmln/index.html`
- MDPTOOLBOX: `https://pymdptoolbox.readthedocs.io/en/latest/`

The system PBCPLUS2MDP, source code, example instances and outputs can all be found at `https://github.com/ywang485/pbcplus2mdp`.

Appendix C PBCPLUS2MDP Input Encoding of the Robot and Block Example

```
astep(0..m-1).
step(0..m).
boolean(t; f).

block(b1; b2; b3).
location(l1; l2).

%% UEC
:- fl_Above(X1, X2, t, I), fl_Above(X1, X2, f, I).
:- not fl_Above(X1, X2, t, I), not fl_Above(X1, X2, f, I), block(X1), block(X2), step(I).
:- fl_TopClear(X, t, I), fl_TopClear(X, f, I).
:- not fl_TopClear(X, t, I), not fl_TopClear(X, f, I), block(X), step(I).
:- fl_GoalNotAchieved(t, I), fl_GoalNotAchieved(f, I).
:- not fl_GoalNotAchieved(t, I), not fl_GoalNotAchieved(f, I), step(I).

:- fl_At(X, L1, I), fl_At(X, L2, I), L1 != L2.
:- not fl_At(X, l1, I), not fl_At(X, l2, I), block(X), step(I).
:- fl_OnTopOf(X1, X2, t, I), fl_OnTopOf(X1, X2, f, I).
:- not fl_OnTopOf(X1, X2, t, I), not fl_OnTopOf(X1, X2, f, I), block(X1), block(X2), step(
    I).

:- act_StackOn(X1, X2, t, I), act_StackOn(X1, X2, f, I).
:- not act_StackOn(X1, X2, t, I), not act_StackOn(X1, X2, f, I), block(X1), block(X2),
    astep(I).
:- act_MoveTo(X, L, t, I), act_MoveTo(X, L, f, I).
:- not act_MoveTo(X, L, t, I), not act_MoveTo(X, L, f, I), block(X), location(L),astep(I).

:- pf_Move(t, I), pf_Move(f, I).
:- not pf_Move(t, I), not pf_Move(f, I), astep(I).

% ---------- PF(D) ----------
%% Probability Distribution
@log(0.8) pf_Move(t, I) :- astep(I).
@log(0.2) pf_Move(f, I) :- astep(I).

%% Initial State and Actions are Random
{fl_OnTopOf(X1, X2, B, 0)} :- block(X1), block(X2), boolean(B).
{fl_At(X, L, 0)} :- block(X), location(L), boolean(B).
{act_StackOn(X1, X2, B, I)} :- block(X1), block(X2), boolean(B), astep(I).
{act_MoveTo(X, L, B, I)} :- block(X), location(L), boolean(B), astep(I).

%% No Concurrency
:- act_StackOn(X1, X2, t, I), act_StackOn(X3, X4, t, I), astep(I), X1 != X3.
```

```
:- act_StackOn(X1, X2, t, I), act_StackOn(X3, X4, t, I), astep(I), X2 != X4.
:- act_MoveTo(X1, L1, t, I), act_MoveTo(X2, L2, t, I), astep(I), X1 != X2.
:- act_MoveTo(X1, L1, t, I), act_MoveTo(X2, L2, t, I), astep(I), L1 != L2.
:- act_StackOn(X1, X2, t, I), act_MoveTo(X3, L, t, I), astep(I).

%% Static Laws
fl_GoalNotAchieved(t, I) :- fl_At(X, L, I), L != l2.
fl_GoalNotAchieved(f, I) :- not fl_GoalNotAchieved(t, I), step(I).
:- fl_OnTopOf(X1, X, t, I), fl_OnTopOf(X2, X, t, I), X1 != X2.
:- fl_OnTopOf(X, X1, t, I), fl_OnTopOf(X, X2, t, I), X1 != X2.
fl_Above(X1, X2, t, I) :- fl_OnTopOf(X1, X2, t, I).
fl_Above(X1, X2, t, I) :- fl_Above(X1, X, t, I), fl_Above(X, X2, t, I).
:- fl_Above(X1, X2, t, I), fl_Above(X2, X1, t, I).
fl_At(X1, L, I) :- fl_Above(X1, X2, t, I), fl_At(X2, L, I).
fl_Above(X1, X2, f, I) :- not fl_Above(X1, X2, t, I), block(X1), block(X2), step(I).
fl_TopClear(X, f, I) :- fl_OnTopOf(X1, X, t, I).
fl_TopClear(X, t, I) :- not fl_TopClear(X, f, I), block(X), step(I).

%% Fluent Dynamic Laws
fl_At(X, L, I+1) :- act_MoveTo(X, L, t, I), pf_Move(t, I), fl_GoalNotAchieved(t, I).
fl_OnTopOf(X1, X2, t, I+1) :- act_StackOn(X1, X2, t, I), X1 != X2, fl_TopClear(X2, t, I),
    not fl_Above(X2, X1, t, I), fl_At(X1, L, I), fl_At(X2, L, I), fl_GoalNotAchieved(t, I)
    .
fl_OnTopOf(X1, X2, f, I+1) :- act_MoveTo(X1, L2, t, I), pf_Move(t, I), fl_At(X1, L1, I),
    fl_OnTopOf(X1, X2, t, I), L1 != L2, fl_GoalNotAchieved(t, I).
fl_OnTopOf(X1, X, f, I+1) :- act_StackOn(X1, X2, t, I), X1 != X2, fl_TopClear(X2, t, I),
    not fl_Above(X2, X1, t, I), fl_At(X1, L, I), fl_At(X2, L, I), fl_OnTopOf(X1, X, t, I),
     X != X2, fl_GoalNotAchieved(t, I).
{fl_OnTopOf(X1, X2, B, I+1)} :- fl_OnTopOf(X1, X2, B, I), astep(I), boolean(B).
{fl_At(X, L, I+1)} :- fl_At(X, L, I), astep(I), boolean(B).

%% Utility Laws
utility(-1, X, L, I) :- act_MoveTo(X, L, t, I).
utility(10) :- fl_GoalNotAchieved(f, I+1), fl_GoalNotAchieved(t, I).
```