INFOMLSAI Logics for Safe AI

Coursework 1 Model Solutions

Tasks that can be done in Week 1 (w/c 26 April)

W1-1 The *office world* domain is shown in Figure 1 and defined in [1]. Provide a reward function specification in LTL for the following office world task: the agent is required to get to a state where coffee is true, and then back to the office, and after that maintain stop forever, all the while without stepping on decorations. Use coffee, office, stop and decs for propositions.

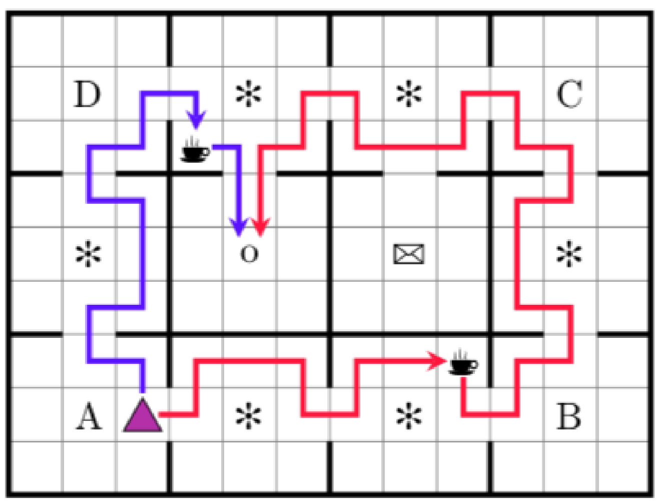

Figure 1: Office World Domain

W1-1 Solution Here is a possible solution (close to what is given in the original paper):

$$G\neg\mathsf{decs} \wedge F(\mathsf{coffee} \wedge XF\mathsf{office}) \wedge G(\mathsf{coffee} \rightarrow XG(\mathsf{office} \rightarrow G\mathsf{stop}))$$

W1-2 Describe a path that satisfies $G\, p \rightarrow F q$ and does not satisfy $G\,(p \rightarrow Fq)$.

W1-2 Solution One possible path (clearly, multiple examples are possible) is: $q_0, q_1, q_1, q_1, \ldots$ (the same state q_1 repeated after the initial one) where q_0 satisfies p, q_0 and q_1 do not satisfy q, and q_1 does not satisfy p. On this path, $G\,p$ is false because p is not true in every state on the path (only in the initial one), $G\,p \to Fq$ is true because implication is true when the antecedent is false, and $G\,(p \to Fq)$ is false because in q_0 p is true but q never becomes true, so Fq is false.

W1-3 In the paper [2], a logic LTL_f is defined, that interprets LTL formulas on finite traces. Let the length of a trace λ be $length(\lambda)$ and $last(\lambda) = length(\lambda) - 1$ (counting from 0). The truth definition for LTL_f modalities is as follows:

- $\lambda \models X\varphi$ iff $0 < last(\lambda)$ and $\lambda, [1, \ldots, last(\lambda)] \models \varphi$;
- $\lambda \models \varphi\, U\, \psi$ iff for some j such that $0 \leq j \leq last(\lambda)$, $\lambda[j, \ldots, last(\lambda)] \models \psi$ and $\lambda[k, \ldots, last(\lambda)] \models \varphi$ for all k with $0 \leq k < j$.

Give an example formula that is true on some finite trace and false on all infinite traces.

W1-3 Solution $F\neg X\top$ (other formulas saying that there is a state with no next state are also fine).

Tasks that can be done during Week 2 (w/c 3 May)

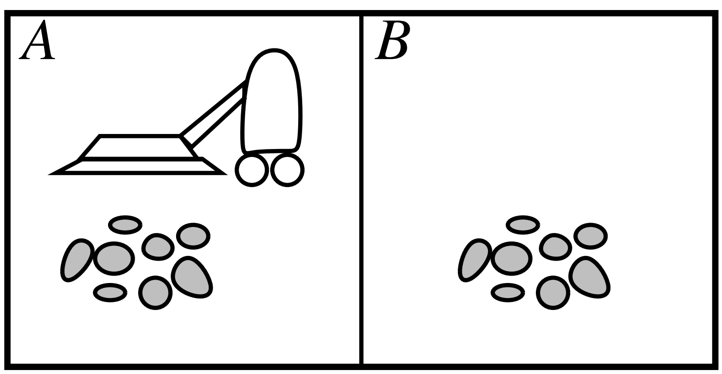

Figure 2: Vacuum World Domain

W2-1 The *vacuum world* domain is shown in Figure 2 and introduced in [3]. Create a description in ISPL of the following vacuum cleaner agent: the variables are inA, inB, cleanA, cleanB, and actions are: $\{nil, move, suck\}$, where nil is doing nothing, $move$ moves to another room, and $suck$ vacuums the room where the vacuum cleaner is (making the room clean). The initial state is when the vacuum cleaner is in room A and both rooms are dirty.

The following formulas should be true there:

- $\mathsf{inA} \wedge \neg\mathsf{inB} \wedge \neg\mathsf{cleanA} \wedge \neg\mathsf{cleanB}$
- $\mathsf{inA} \rightarrow EF\mathsf{inB}$
- $\mathsf{inA} \rightarrow EF\mathsf{cleanA}$
- $EF(\mathsf{cleanA} \wedge \mathsf{cleanB})$

Extract a witness for $EF(\mathsf{cleanA} \wedge \mathsf{cleanB})$.

W2-1 Witness for $EF(\mathsf{cleanA} \wedge \mathsf{cleanB})$

```
-- State 0 --
  Agent Environment
  Agent Vacuum
    cleanA=false
    cleanB=false
    room=a

-- State 1 --
  Agent Environment
  Agent Vacuum
    cleanA=true
    cleanB=false
    room=a

-- State 2 --
  Agent Environment
  Agent Vacuum
    cleanA=true
    cleanB=false
    room=b

-- State 3 --
  Agent Environment
  Agent Vacuum
    cleanA=true
    cleanB=true
    room=b
```

The following formulas should be false in the initial state:

- cleanB
- $AG\mathsf{inA}$
- $AF\mathsf{inB}$

Extract a counterexample for $AG\mathsf{inA}$.

W2-1 Counterexample for $AG\mathsf{inA}$

```
-- State 0 --
Agent Environment
Agent Vacuum
  cleanA = false
  cleanB = false
  room = a

-- State 1 --
Agent Environment
Agent Vacuum
  cleanA = false
  cleanB = false
  room = b
```

W2-2 Given the `one-robot+carriage.ispl` file, translate the following English properties in CTL (you can use ascii encoding as in MCMAS) and check using MCMAS whether they hold. Submit the translations together with the MCMAS outcomes (and witnesses/counterexamples if appropriate).

W2-2 Solutions

- it is possible to avoid pos2 forever: $EG\,(\neg \mathsf{pos2})$ is *TRUE* with witness

```
-- State 0 --
  Agent Environment
  Agent Robot1
    pos=pos0
```

- it is possible to be in position pos2 in the next step: $EX\mathsf{pos2}$ is *FALSE*. Note that MCMAS incorrectly generates a counterexample for this property—as explained in the lectures, model checkers can only produce useful counterexamples for false universal properties. No points were lost for submitting the counterexample produced by MCMAS.
- it is possible in the future to be in position pos1 and in the next step after that in pos2: $EF(\mathsf{pos1} \wedge EX\,\mathsf{pos2})$ is *TRUE* with witness

```
-- State 0 --
  Agent Environment
  Agent Robot1
    pos=pos0

-- State 1 --
  Agent Environment
  Agent Robot1
    pos=pos0

-- State 2 --
  Agent Environment
  Agent Robot1
    pos=pos1
```

```
-- State 3 --
  Agent Environment
  Agent Robot1
    pos=pos2
```

- it is possible to be in pos0 until reaching pos2: $E\mathsf{pos0}\,U\,\mathsf{pos2}$ is *FALSE*. As above, MCMAS incorrectly generates a counterexample for this property.
- it is possible to be in pos0 or pos1 until reaching pos2: $E(\mathsf{pos0}\vee\mathsf{pos1})\,U\,\mathsf{pos2}$ is *TRUE* with witness

```
-- State 0 --
  Agent Environment
  Agent Robot1
    pos=pos0

-- State 1 --
  Agent Environment
  Agent Robot1
    pos=pos1

-- State 2 --
  Agent Environment
  Agent Robot1
    pos=pos2
```

- it is possible to always have pos1 reachable in at most 2 steps: $EG\,EX(\mathsf{pos1}\vee EX\,\mathsf{pos1})$ is *TRUE* with witness

```
-- State 0 --
  Agent Environment
  Agent Robot1
    pos=pos0

-- State 1 --
  Agent Environment
  Agent Robot1
    pos=pos1

-- State 2 --
  Agent Environment
  Agent Robot1
    pos=pos1
```

W2-3 The CTL truth definition counts the present state as 'future': for example, $M, q \models E\varphi\,U\,\psi$ if $M, q \models \psi$. Consider a definition that instead does not count the present as the future:
$M, q \models E\varphi\,U'\psi$ iff there exists a path λ from q such that for some $i > 0$, $M, \lambda[i] \models \psi$, and for all j with $0 \leq j < i$, $M, \lambda[j] \models \varphi$.

Write the case for the model checking algorithm for CTL with this EU' modality.

W2-3 Solution One possible solution is to define $E\varphi\, U'\psi$ as $\varphi \wedge EX\, E\varphi\, U\, \psi$ and use the existing model checking algorithm for this formula. Alternatively, we can add a case for $E\varphi\, U'\psi$ to the algorithm:

$$
\begin{array}{l}
\textbf{case } \varphi' = E\psi_1 U'\psi_2 \\
\quad Q_1 \leftarrow \emptyset; \quad Q_2 \leftarrow pre_\exists([\psi_2]_M) \cap [\psi_1]_M \\
\quad \textbf{while } Q_2 \not\subseteq Q_1 \textbf{ do} \\
\quad\quad Q_1 \leftarrow Q_1 \cup Q_2; \quad Q_2 \leftarrow pre_\exists(Q_1) \cap [\psi_1]_M \\
\quad [\varphi']_M \leftarrow Q_1
\end{array}
$$

The case for $A\psi_1\, U\, \psi_2$ is similar, with $pre_\exists$ replaced with $pre_\forall$.

References

[1] Alberto Camacho, Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Anthony Valenzano, and Sheila A. McIlraith. LTL and beyond: Formal languages for reward function specification in reinforcement learning. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 6065–6073. ijcai.org, 2019. `https://doi.org/10.24963/ijcai.2019/840`.

[2] Giuseppe De Giacomo and Moshe Y. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In Francesca Rossi, editor, *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, pages 854–860. IJCAI/AAAI, 2013. `http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6997`.

[3] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 1995. `http://aima.cs.berkeley.edu`.