INFOMLSAI Logics for Safe AI
Coursework 3 Model Answers

Coursework released:	31 May 2021, on Blackboard
Coursework due:	23:59 11 June 2021, on Blackboard
Submission format:	a folder containing a pdf and an .ispl file, one per group

Please do the coursework in groups of 2-3 people. Submit a single zipped folder on Blackboard for your group. The folder should contain the pdf file and the ispl file. Please name the folder group-X, where X is the number of your group, and state in the pdf file and comments in the ispl file the names of the members of the group.

Tasks that can be done in Week 6 (w/c 31 May)

The following task can be done after watching the lecture on Coalition Logic

W6-1 Represent two agents playing Rock, Paper, Scissors, as a Concurrent Game Structure M_{rps}. Assume that atomic propositions are win1 and win2. Look at the encoding of the Prisoner's Dilemma in Slides 5/1 or in the reader. However instead of making the game single shot (so that after the first move, the game stays in the same state whatever the actions) allow for iterated playing. (1 mark)

Answer:

$M_{rps} = \langle\{1,2\}, \{q_0, q_1, q_2, q_3\}, \mathcal{V}, \{ rock, paper, scissors\}, d, o\rangle$, where:

- q_0 is the initial state (not strictly necessary), q_1 is where player 1 wins, q_2 is where player 2 wins, and q_3 is a draw.
- $\mathcal{V}$ assigns sets of states to propositions win_1 for player 1 wins and win_2 for player 2 wins (draw is when neither wins):
  - $\mathcal{V}(win_1) = \{q_1\}$
  - $\mathcal{V}(win_2) = \{q_2\}$
- $d(a, q) = \{rock, paper, scissors\}$ (d makes all actions available to all agents in all states)
- o is as follows, for any state $q_i \in \{0, 1, 2, 3\}$:
  - $o(q_i, rock, rock) = q_3$
  - $o(q_i, rock, paper) = q_2$
  - $o(q_i, rock, scissors) = q_1$

- $o(q_i, paper, rock) = q_1$
- $o(q_i, paper, paper) = q_3$
- $o(q_i, paper, scissors) = q_2$
- $o(q_i, scissors, rock) = q_2$
- $o(q_i, scissors, paper) = q_1$
- $o(q_i, scissors, scissors) = q_3$

W6-2 Express in Coalition Logic: coalition of agents $\{1, 2\}$ can enforce win_1 and the same coalition can enforce $\neg\mathsf{win}_1$. Is this formula is true in q_0 in CGS M_{rps}? Justify your answer. (1 mark)

Answer: The formula is $[\{1, 2\}]\,\mathsf{win}_1 \wedge [\{1, 2\}]\neg\mathsf{win}_1$. It is true in q_0. This is because in q_0, 1 and 2 can select a joint action $\langle rock, scissors\rangle$ and enforce outcome $\{q_1\}$ where win_1 holds, so $[\{1, 2\}]\,\mathsf{win}_1$ is true. They can also select a joint action $\langle scissors, rock\rangle$ and enforce outcome $\{q_2\}$ where $\neg\mathsf{win}_1$ holds, so $[\{1, 2\}]\neg\mathsf{win}_1$ is true.

The following task can be done after watching the lecture on ATL:

W6-3 In a fairly played Rock, Paper, Scissors game, neither of the players has a winning strategy (although this Janken robot (link) always wins against humans). However assume that, the first player is either stupid or does not care, and in iterated games, the first player always plays a fixed strategy of cycling thorough rock, then paper, then scissors, then again rock, etc. Change CGS M_{rps} into M_{rps1f} (for fixed strategy of 1) to represent this fixed strategy of the first player. Player 2 should be unchanged and have a free choice of actions in every state. *Hint: you can encode in the state which action Player 1 has just played or is about to play.*

In this structure, player 2 should have a strategy to always win regardless where the play starts and where player 1 is in its cycle of chosing action rock, paper, scissors. (In a real game situation, player 2 would need to know this to chose the first correct action. But so far we are in a perfect information setting, so it is enough that player 2 *has* a correct action, and we are not worrying about how it knows what this action is.) (1 mark)

Answer: $M_{rps1f} = \langle\{1, 2\}, \{q_0, q_1^r, q_2^r, q_3^r, q_1^p, q_2^p, q_3^p, q_s^r, q_2^s, q_3^s\}, \mathcal{V}, \{rock, paper, scissors\}, d, o\rangle$, where:

- q_0 is the initial state as before, we assume that player 1 plays $rock$ there. q_1^x is where player 1 wins and player 1 is about to play action $x \in \{r(ock), p(aper), s(cissors)\}$, q_2^x is where player 2 wins and player 1 is about to play action x, and q_3^x is a draw and player 1 is about to play action x.
- $\mathcal{V}$ assigns sets of states to propositions win_1 for player 1 wins and win_2 for player 2 wins:
  - $\mathcal{V}(win_1) = \{q_1^x\}$
  - $\mathcal{V}(win_2) = \{q_2^x\}$

- $d(1, q^r) = \{rock\}$, $d(1, q^p) = \{paper\}$, $d(1, q^s) = \{scissors\}$ and $d(2, q) = \{rock, paper, scissors\}$ (d makes all actions available to agent 2 in all states q)
- o is as follows, for $i \in \{1, 2, 3\}$:
  - $o(q_0, rock, rock) = q_3^p$;
  - $o(q_0, rock, paper) = q_2^p$;
  - $o(q_0, rock, scissors) = q_1^p$;
  - $o(q_i^r, rock, rock) = q_3^p$
  - $o(q_i^r, rock, paper) = q_2^p$
  - $o(q_i^r, rock, scissors) = q_1^p$
  - $o(q_i^p, paper, rock) = q_1^s$
  - $o(q_i^p, paper, paper) = q_3^s$
  - $o(q_i^p, paper, scissors) = q_2^s$
  - $o(q_i^s, scissors, rock) = q_2^r$
  - $o(q_i^s, scissors, paper) = q_1^r$
  - $o(q_i^s, scissors, scissors) = q_3^r$

W6-4 Express in ATL: player 2 has a strategy to always win. Note that in the current state win_2 does not have to hold yet, so 'always' here should mean from the next state onwards. Define a memoryless strategy for player 2 that makes this formula true in any state in M_{rps1f}. (1 mark)

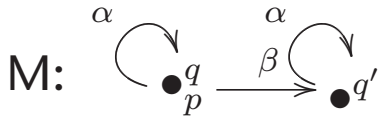
Answer: $\langle\langle\{2\}\rangle\rangle X \langle\langle\{2\}\rangle\rangle G\, \mathsf{win}_2$. The strategy is to chose paper when player 1 choses rock, scissors when player 1 choses paper, and rock when player 1 choses scissors, formally:

- $f_2(q_0) = paper$
- $f_2(q_i^r) = paper$
- $f_2(q_i^p) = scissors$
- $f_2(q_i^s) = rock$

W6-5 In ATL*, it makes a difference whether strategies are memoryless or perfect recall. Perfect recall strategies are unrealistic in many circumstances. Usually, there is a bound on the number of previous states the agent can remember, so its strategies always take a bounded number of preceding states into account, such as at most 10 or 100000 (but not an arbitrary number).[1] In both counterexamples to the equivalence of perfect recall and memoryless strategies in ATL* (in the lecture and in the reader), one only needs a bounded strategy to enforce the property (action choice is made knowing the current and the preceding state, which is a history of length at most 2). Give an example of an ATL* property and a CGS, where the bound 2 is not sufficient for a strategy to enforce the property. (1 mark)

[1] For a proper definition and discussion of bounded strategies, see [1], although note that this paper applies to imperfect information setting.

Answer. The simplest example is making the agent to count to 3 instead of 2, in the modification of the example in the lecture:



$$\varphi = \langle\langle a \rangle\rangle (Xp \wedge XXp \wedge XXX\neg p)$$

(the strategy is $q \mapsto \alpha$, $qq \mapsto \alpha$, $qqq \mapsto \beta, \dots$)

Tasks that can be done during Week 7 (w/c 7 June)

The following tasks can be done after watching the lectures on model checking ATL.

W7-1 Implement M_{rps1f} in ISPL. Check that the formula from W6-4 is true there and provide a witness strategy (look in the formula1.dot file for the actions).

Hint: when you are encoding a pre-defined CGS in ISPL, it is often the easiest to define a variable `state: {q0, q1, q2}` *(where the values are all the states in the CGS) in the Environment agent, and copy the transition function into the Environment's Evolution function, as in*
`state = q1 if state = q0 and Agent1.Action = alpha`
Agents should have state in their Lobsvars because their Protocol depends on states. (1 mark)

Suggestion for testing: if the groups are defined as follows:

```
Groups
       g1 = {Player1};
       g2 = {Player2};
       g_all = {Player1,Player2};
end Groups
```

then some formulas that should be true are:

```
<g2> X win2;
<g2> F win1;
!<g1> X win1;
<g_all> F win1;
```

Answer: See `w-7-1-rps1f.ispl`

W7-2 Implement the following scenario in ISPL. The museum consists of three rooms along a corridor: *room0*, *room1*, *room2*. The entrance is *room0*, the exit is *room2*. A valuable painting is in *room2*. A guard is patrolling the three rooms in the following fixed pattern: *room0, room1, room2, room1, room0, room1, room2*, ... The guard has an action to move *left* and *right* to the next room; the encoding

should make sure that the guard only moves all the way to the right and all the way to the left along the corridor. The thief has the same actions *left, right* but also *wait, steal* and *exit*. The thief can steal the painting if he is alone in *room2* (the guard is in another room). The thief is caught if he is in the same room with the guard after the painting is stolen. The goal of the thief is to have the painting stolen and exit without being caught (to exit, the thief needs to be in *room2*). Express in ATL and check in your encoding that the thief can steal the painting and escape while not being caught. Hint: encode in the `Protocol` which actions are available in which state; the guard's direction of patrolling can be encoded as part of the guard's state. Another hint: the states are quite large, so it is handy to specify evolution of each variable separately. To make sure they update in sync, precede the file with a statement `Semantics=SingleAssignment`. This makes sure that all applicable clauses are applied simultaneously. This requires that only one clause specifying how a variable is updated is enabled in each state.

Suggestion for testing: if `gT` is the thief, and *escaped* becomes true after executing an exit action, then some formulas that should be true are:

```
<gT> G !caught;
<gT> F (stolen and escaped);
!<gT> X stolen;
!<gT> G escaped;
!<gT> X caught;
```

Note that a conjunction of the first two formulas is *not* the correct answer. It is possible to have a strategy to never be caught (e.g. by not stealing) and a different strategy to steal and escape, but the requirement is to have one single strategy to achieve both. Hint: use Until to express that the thief can steal the painting and escape while not being caught.

(1 mark)

Answer: See `w7-2-thief-guard.ispl`. The formula is

```
<gT> ((!caught) U (stolen and escaped)).
```

W7-3 Modify the guard's patrol behaviour to allow the guard to *wait* in each room, and to perform both the *left* and *right* actions in *room1*, i.e., the guard's patrol behaviour is unrestricted. Express in ATL and check in your encoding that the guard can always prevent the picture from being stolen. (1 mark)

Answer: See `w7-3+4-thief-guard.ispl`. The formula is

```
<gG> G (!stolen)
```

W7-4 Using your encoding for **W7-3**, express in ATL and check in your encoding that a coalition consisting of the thief and the guard can ensure that the painting is stolen and the thief exits without being caught, i.e., if the guard's patrol behaviour is unrestricted, the thief needs the guard's help to steal the picture. (1 mark)

Answer: See `w7-3+4-thief-guard.ispl`. The formula is

```
<gTG> ((!caught) U (stolen and escaped)).
```

W7-5 Concurrent game structures are represented in ISPL in a compact form (see reader sections 5.3.1 and 5.3.2). Give an example of an ISPL encoding where the transition relation in the corresponding CGS has the size exponential in the size of the ISPL representation of the transition relation (the number of Evolution clauses). Do not write your own encoding if you can point to one of the encodings from this or previous coursework or from the examples that come with MCMAS. (1 mark)

Answer: In the museum thief example, the transition relation is encoded in the `Evolution` sections of the environment, and the thief and guard agents. For each property of a state (such as the position of the thief, the position of the guard, etc.), the `Evolution` specifies how it changes in terms of agent's actions. The total size of all `Evolution` statements is bounded by $O(p \times (k + d))$ where p is the number of properties, k the number of agents, and d is the number of actions. So even if we need to list several actions per agent in each clause, it is still linear in the numbers of agents and actions.

A corresponding CGS will have $O(2^p)$ states. From each state there is at least one transition (in fact in the worst case it is $O(d^k)$ transitions), so the CGS representation is exponentially larger.

References

[1] Steen Vester. Alternating-time temporal logic with finite-memory strategies. In Gabriele Puppis and Tiziano Villa, editors, *Proceedings Fourth International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2013*, volume 119 of *EPTCS*, pages 194–207, 2013.