Among Us: Modelling movement with epistemic logic

Group 19: Barbera de Mol (s3374610)
Jeroen Muller (s2590182)
Max Valk (s3246922)

June 25, 2021

Background

One important kind of knowledge used in reasoning about the Among Us game concerns the way players move from room to room. It is common for players to discuss knowledge like *"I was close to A while B was killed at the other side of the map. This means A cannot be the killer."* Can we capture this kind of reasoning in epistemic logic? A good player does not just consider the information that he observed directly, but also what can be extrapolated from it by considering the rules of the game. For example, A may have been out of sight for a short time, but we know this period was too short for A to move all the way to the place where B's body was found. This means we still know enough to not suspect A of the killing of B

Game rules

We can try to capture this kind of reasoning by construct a Kripke model that describes what players know about the state of the game world at any given time, and the way their actions change the state of the game. We will study a highly simplified version of the game in which we consider agents' *roles*, *actions* and *locations*. Let our agents be denoted by the set $\mathcal{A}$. Agents can have the following roles:

- *Impostor* The impostor can move around in the game world, and can kill crewmates
- *Crewmate* Crewmates can only move around
- *Dead* Once a crewmate is killed by the impostor, they remain inactive in the loction where they died until the end of the game.

The role of the agents is denoted by a set of atomic propositions $\{X_{\mathsf{imp}}; X_{\mathsf{crew}}; X_{\mathsf{dead}} | X \in \mathcal{A}\}$, denoting that agent $X \in \mathcal{A}$ is respectively the impostor, a crewmate of dead.

At any given time step, each agent is in one of a fixed set of rooms $\mathcal{R}$. We describe the location with the set of propositions $\{X_r | X \in \mathcal{A}, r \in \mathcal{R}\}$, where X_r denotes that X is in room r. There is a symmetric and reflexive relation C on the rooms such that $r_1 C r_2$ if room r_2 can be reached from room r_1 in one movement step.

The state of the game can be changed by three types of action, that will be formalized as DEL action models below:

- *Move* An agent moves from one room to a connected room (which may be the same room they are currently in, as the connection relation is reflexive);

- *Kill* The impostor kills a crewmate that is currently in the same room as the impostor;
- *Report* An agent reports the location and identity a dead agent that is currently in the same room as the reporting agent; It may be better to change the rules such that dead bodies are reported immediately.
- *Observe* Each agent observes which agents are in the same room as him, and if they are dead or alive.

The game is divided into a set of time steps labeled $t_1, \ldots, t_N$. In every time step, the following actions take place in sequence:

1. Each living agent makes a *move* actions;
2. Each agent *observe*s the room they are in;
3. The impostor performs zero or one *kill* actions;
4. Each agent *observe*s the room they are in;
5. Dead bodies that are in the same room as a living crewmate are *report*ed, and their identity and location is publicly announced.

The movement and kill actions here depend on the decisions of the players, but the observe and report actions are fixed and depend only on the state of the world.

Kripke model

The following conjunction must hold in any consistent game state, acconting for the fact that agents can only have one role and be in one place at any time:

$$\bigwedge_{X \in \mathcal{A}} \left[(X_{\mathsf{imp}} \vee X_{\mathsf{crew}} \vee X_{\mathsf{dead}}) \wedge \bigvee_{r \in \mathcal{R}} X_r \right]$$

.

Relations

Action models

We formalize the actions by definining an *action model* for each action. By executing the action model, we can transform the Kripke model.

Single agent movement

One state for each room connection in either direction. All states are equivalent for all agents but the moving agent, and have as pre- and postcondition those states.

Observation

This action model has one state for each possible configuration of the world. States are equivalent for an agent if the room in which that agent is is identical.

$$s \sim_a t \quad \text{iff} \quad \forall_{b \in \mathcal{A}} \forall_{r \in \mathcal{R}} \left[(s \in V(a_r) \wedge s \in V(b_r)) \Leftrightarrow (t \in V(a_r) \wedge t \in V(b_r)) \right]$$

We can construct this action model as follows:

- Enumerate all possible configurations
- For each configuration, divide the propositions into subsets based on the room they refer to
- For each agent, store the room
- Enumerate all pairs of configurations
- Divide the proposition into subsets, one for each room
-

Removing faraway states

We should be able to remove states that are more than 1 step away from the point without affecting first-order knowledge.

We only need to generate the action models that satisfy the preconditions in at least one state.

In the reference, action models are defined such that a postcondition defines the truth of *all* atomic prepositions. However in the figures, they only show a limited set of postconditions. Of course this means that the domain of discourse of the action model is smaller than that of the Kripke model.

Composing action models

The notation $\langle \mathsf{M}, \mathsf{j} \rangle \, \mathsf{pre}'(\mathsf{j}')$ becomes more clear when considering DEL pp. 150. We should read this as an abbreviation of $\neg \, [\mathsf{M}, \mathsf{j}] \, \neg \mathsf{pre}'(\mathsf{j}')$. So a model satisfies $\mathsf{pre}((\mathsf{j}, \mathsf{j}'))$ if

Move

First the move action for a single agent.

States $\{\mathsf{move} : (A, r, s) | A \in \mathcal{A}; rCs\} \cup \{A_{\mathsf{dead}}\}$

Preconditions A_r, $\neg A_{\mathsf{dead}}$

Postconditions $\neg A_r$, A_s

Relations States are related for all agents that are not A.

The full move action is the composition of one such action model for each agent.

Kill

There is one state in the *kill* action model for each combination of impostor, crewmate and room, plus another for inaction.

States

Preconditions A_{imp}, B_{crew}, A_R, B_R

Postconditions $\neg A_{\mathsf{crew}}$, A_{dead}

Relations States are related for all agents not partaking in the action.

Observe

States One for each possible state of the game world

Preconditions All propositions that are true in that world state

Postconditions None

Relations States are related for an agent if all propositions involving the room where that agent is are the same.

Report

States One for each

Preconditions All propositions that are true in that world state

Postconditions None

Relations States are related for an agent if all propositions involving the room where that agent is are the same.

We will formalize the three actions as DEL action models with assignment acting on the Kripke model.

Basic actions can be written as an action model with assignment. A set $\mathcal{A}$ of agents, a set $\mathcal{R}$ of rooms, and a relation C Our world consists of a set of rooms $R_1, \ldots, R_n$ and a set of agents $A, \ldots, X$. The game starts at time step t_0, and in each time step all agents can take a movement action, possibly followed by another action (kill or report).

Postconditions: How can we use postconditions? We should use them to track the implications of the history. I think we should also track locations of bodies in the Kripke model. The Kripke model consists of disconnected sets of states, one for each assignment of roles (in the simple case, one for each possible impostor).

Propositions: A_1 (A is in room 1), A_{imp} (A is the impostor), en A_{alive} (A is alive). The action model for a movement step consists of a single movement action for all living agents, and our action model for the action step consists of zero or one action. The possible actions can depend on preconditions such as the role of the impostor.

0.1 Updating the Kripke model with postconditions

The Kripke model is updated by executing an action model with preconditios and postconditions. This is the Dynamic Epistemic Logic with Attributions.

An action model with postconditions has the following structure: $\langle \mathsf{S}, \sim_a, \mathsf{pre}, \mathsf{pos} \rangle$. Intuitively, we simply apply the action model (constructing a Cartesian product of actions with states) and then apply the assignments in the postcondition to each state. The postconditions are given as a set of assignments (proposition, true/false). This means every application of the action model increases the number of states, such that there is one state for each possible history.

States can be removed from the model if an action model is applied where they satisfy none of the preconditions.

By applying such an action model in each step we can keep track of such a Kripke model throughout the game. When an announcement occurs, we can simply throw away some of the states.

0.2 Observations

Observations can be modeled as actions without postconditions. It is possible (and maybe convenient) to incorporate the observations in the action model by specifying that some agents can distinguish some of the actions.

0.3 Identical states

Due to the assignment, it is possible that multiple states with the same valuation are created at some point. We can replace all such states with their bisimulation class.

0.4 Example case

We work out a very simple case on paper. What is the minimum size where such a model becomes useful? I think we need at least four agents: one agent that is killed, one impostor, three agents that remain (for two agents, the roles become trivial after someone is killed because the remaining crewmate can identify the impostor).

Agents A and B walk together, and C and D walk together. Then, C kills D. Because A and B know each other's position, they both know C must be the killer, and vote to remove him.

Time	Room 1	Room 2	Room 3
$t = 0$		ABCD	
$t = 1$	AB		CD
$t = 2$	A	BC	~~D~~
$t = 3$	C	A	B~~D~~

Enumerating agent knowledge

Questions

What do the models look like? I guess they will be $S5_{(m)}$-models, because both actions and the initial model satisfy these conditions.

References

- *Benevides and Lima 2017* Action Models with Postconditions
- *Jan van Eijck* Bisimulation
- Python-Bisimulation
- *Hans van Ditmarsch, Wiebe van der Hoek and Barteld Kooi* Dynamic Epistemic Logic
- https://stackoverflow.com/questions/38924421/is-there-a-standard-way-to-partition-an-interable-into-equivalence-classes-given