An abstract multi-agent framework applied to a social interaction game

Hendrik Wietze de Haan Wim H. Hesselink
Gerard R. Renardel de Lavalette

Department of Mathematics and Computing Science, University of Groningen,
P.O. Box 800, 9700 AV Groningen, The Netherlands

Abstract

We develop an abstract framework for modelling multi-agent systems. The framework splits the state of an agent into an internal state and a set of observables. In the execution model, an agent performs either an observation step, a reasoning step, or an external step, i.e. an action that affects other agent's observables. We use this framework to model the game Mafia as a multi-agent system.

1 Introduction

In this paper we report on ongoing research in the design of multi-agent systems (MAS). We intend to contribute to a methodology for design of and reasoning about MAS, based on mathematical semantics. We see MAS as a special kind of concurrent systems, and therefore we base our work on existing theories related to concurrency [1, 8]. We work out a mathematical framework, based on the distinction between internals (the mental states of the agents) and externals (things in the outside world that can be observed by the agents). We focus on the structure of the mental states and the kind of reasoning that takes place. As a test case, we apply our framework to model the social interaction game Mafia.

Current multi-agent frameworks focus on the agent programming language or the architecture. Agent0 [11] emphasizes the formal semantics of the mental state of an agent, but offers no formal language for programming these agents. AgentSpeak(L) [10] and 3APL [6] provide such languages. DESIRE [2] is a compositional architecture framework used to specify a real world multi-agent system. The focus there is on what kind of knowledge is needed for which agents. Agents are typically described in intentional terms such as beliefs, knowledge, desires, intentions etc, generally expressed using modal logics.

In earlier work ([4], [5]), we investigated the semantics of knowledge-based programs, focusing on the resolution of a circularity: the knowledge of an agent is defined in terms of possible behaviours of the program, while the possible behaviours are determined by actions that depend on the knowledge of agents. The link with the work reported here is that we use the same mathematical starting point (trace theory), but the emphasis is now on the structure of the mental state of the agents.

In section 2 we present our abstract framework. The game Mafia is described in section 3. In section 4, the game is formalized with some strategies for the players. We end with discussion and ideas for further work in section 5.

2 Abstract Agent Model

Let $\mathcal{A}$ be the set of all agents in the system. Each agent $a \in \mathcal{A}$ has an internal state I_a.

Agents can observe (part of) the external world, and so we let E be the set of externals. Agents can perform actions that change the external world, and thus influence the externals observed by other agents.

We concentrate on a core of three activities present in most agent frameworks. These three activities of an agent are: making observations, reasoning about the observations and taking actions. This is a common design known as *sense-reason-interact*-cycle.

The essence of these three activities is captured by

$$\begin{aligned} &sense_a : I_a \times E \rightharpoonup I_a, \\ &reason_a : I_a \rightharpoonup I_a, \\ &act_a : I_a \times E \rightharpoonup E. \end{aligned}$$

An agent may change her internal state due to externals, or by reasoning. The external world is changed by an agent when she takes actions. The choice of action is based on her internal state.

The multi-agent system in itself has a global state consisting of the internal states of the agents and the state of the external world. That is, the global state space X is defined as $X = (\Pi_{a \in \mathcal{A}} I_a) \times E$. For simplicity we assume interleaving semantics, i.e. actions are serialized. True parallelism complicates more than that it adds in expressiveness. An execution of the multi-agent system is a sequence of global states i.e. an element from X^*, such that subsequent pairs of elements are related via one of the three functions $sense_a, reason_a$ or act_a of some agent a. The set of all executions is regarded as the semantics of the system.

For the moment we place no restriction on the order in which the agents perform their activities. This is in contrast to [12], where reasoning should precede (inter)action. We prefer a more general framework: if a specific order is needed this can be coded into the activities.

2.1 A more concrete framework

In this section we pin down the formalism more strictly. For the sake of incremental design, it is advantageous to model $sense_a$, $reason_a$, and act_a as binary relations rather than functions. In these relations, we also change the arguments for sensing, reasoning or acting, so that the agent "knows" about its own steps and need not repeat them unwittingly.

We distinguish a set Obs of observables and a set Act of action symbols. With each action symbol $f \in Act$ an interpretation $[\![f]\!] \subseteq E \times E$ is associated. Each agent a has a mailbox $E_a \subseteq Obs$ of observables that have not yet been *sensed*. The set E of externals is refined to $E = \Pi_{a \in \mathcal{A}} E_a$. For 1-to-1 communication we introduce a primitive action symbol *send*(*Receiver*, *Message*) where *Receiver* $\in \mathcal{A}$ and *Message* $\in Obs$. Similarly,

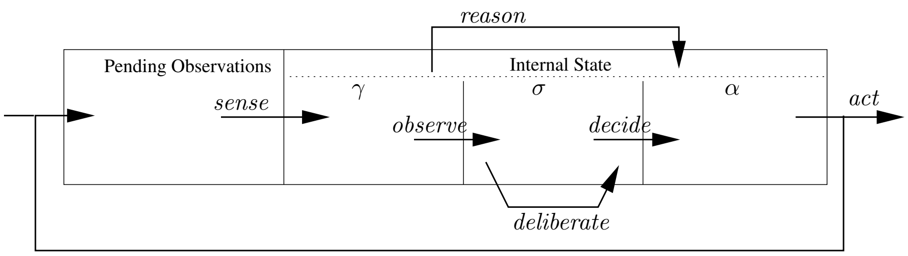

Figure 1: local state of an agent, and relations between components

we introduce a primitive *broadcast*(*Message*). Interpretation of these primitives is quite standard: *send*(r, m) places observable m in mailbox E_r, and *broadcast*(m) places m in all mailboxes.

We model the activities $sense$, $reason$ and act nondeterministically, i.e. as relations rather than functions. Relation $sense_a$ moves an observable from mailbox E_a and places it into the internal state I_a. Relation $reason_a$ updates beliefs based on the new observations, and plans actions to be executed. The relation act_a arbitrarily executes a planned action. The internal state I_a of agent a is a triple (γ, σ, α) of performed observations $\gamma \subseteq Obs$, a mental state component σ and a representation $\alpha \subseteq Act$ of planned actions. All agents have the same $sense$ and act relations, the difference (and complexity) will be in the $reason$ relation.

The relation $reason$ implements the strategy of the player, and may be split up into updating the beliefs based on observations, further deliberation, and selecting an action based on beliefs. That is, $reason$ is a choice between $observe$, $deliberate$ and $decide$:

$$reason = observe \cup deliberate \cup decide.$$

Figure 1 illustrates the state of an agent and relations between components. Figure 1 does not imply any sequentiality, but only a causal relation of the elementary steps. For simple agents the relation $deliberate$ will be the empty relation, since no higher order reasoning is done. This relation will play a role when the mental state represents higher order beliefs, or beliefs on strategies of other agents.

Our framework offers a programming language in which the designers of the multi-agent system can define these relations for its agents. A program is a nondeterministic choice over some guarded commands. Each guarded command is preceded by a $[\!]$ and consists of a guard and a body (actions) separated by an arrow. The guard expresses a condition on the internal state (γ, σ, α) of the agent. The body expresses how the internal state changes. Free variables in a guarded command are implicitly quantified existentially.

3 The game Mafia

To investigate our framework we have chosen to model a simple social game called Mafia; see [7] for some versions of this game. The behaviour of the players within the rules of the game is unrestricted. Strategies of the players can vary wildly, and may be as complex as one wants. As a consequence no exact analysis of (winning) strategies is possible. Simulation on the other hand is quite feasible. The game can be seen as a situation that might be studied in the social sciences. We chose this game since it serves to investigate our framework. It is not our aim to find winning strategies.

In the game Mafia the players are either civilians or mafiosi. At the beginning of the game the games-master secretly assigns some players as mafia. Only mafiosi know who are mafia, i.e. mafia-membership is common knowledge for the mafia. Civilians are ignorant who might be mafia. A civilian knows herself to be civilian.

The task of the civilians is to eliminate the mafia, while the mafiosi try to eliminate all civilians. The mafiosi are allowed to settle on a strategy, before the game proceeds. Civilians have no common strategy. The game proceeds in rounds of day and night.

During the day the players, civilians and mafiosi, publicly discuss and accuse each other of being mafia. After some time the games-master calls for a voting round. Each player publicly nominates one other player as a mafiosa. The games-master keeps track of the votes. A player with the most votes is eliminated from the game.

After a player has been voted out of the game, the games-master announces that it has become night. During the night mafiosi shoot at players: each mafiosa secretely communicates to the games-master at whom she shoots. Each mafiosa must shoot precisely once during the night. When all mafiosi have fired, the games-master announces who were killed during the night. A player is killed if she is shot at least twice during the same night. If there is only a single mafiosa remaining, a single shot is fatal. The game ends when all civilians or all mafiosi have been eliminated.

4 Modelling the game Mafia

In this section we fill in the framework from 2.1, such that we can model the game Mafia and show how simple strategies can be defined. We start by defining the external world in terms of observables and action symbols. For the internal state the mental state component σ must be defined.

We assume a set $\mathcal{A}$ of agents, that is partitioned into a set of players $\mathcal{A}_p$ and a games-master denoted by *gm*.

Actions in the game are accusing, voting for or shooting at players. Other actions are the announcements by the games-master. We define a set of observables by

$$\begin{aligned} \mathit{Obs} = \quad & \{\mathit{accusing}, \mathit{voting}, \mathit{shooting}, \mathit{eliminated}(a), \\ & \mathit{accuse}(a,b), \mathit{vote}(a,b), \mathit{shoot}(a,b) \mid a, b \in \mathcal{A}_p\} \end{aligned}$$

The intended meaning of $\mathit{accuse}(a,b)$ is that player a accuses player b of being a mafia-member. All communications are broadcast, except for the shots of the mafia, so the set of action symbols is

$$\begin{aligned} \mathit{Act} = \quad & \{\mathit{broadcast}(\mathit{accusing}), \mathit{broadcast}(\mathit{voting}), \mathit{broadcast}(\mathit{shooting}), \\ & \mathit{broadcast}(\mathit{eliminated}(a)), \mathit{broadcast}(\mathit{accuse}(a,b)), \\ & \mathit{broadcast}(\mathit{vote(a,b)}), \mathit{send}(gm, \mathit{shoot}(a,b)) \mid a, b \in \mathcal{A}_p\} \end{aligned}$$

We must choose how to represent the beliefs. Only civilians have beliefs on who might be mafia while mafia *know* who are mafia. Civilians may also believe players not to be mafia. So the mental component σ of civilian a may contain propositional atoms *ismafia*(b) or *isnotmafia*(b) ($b \in \mathcal{A}_p$) expressing that a believes b to be mafia or believes that b is not mafia. For a mafiosa, the set σ contains propositions *ismafia*(b) for all players b that are mafia.

The component σ should also track which players are still in the game, since accusing, voting or shooting at players that are already out of the game is not useful. Therefore, *alive*$(b) \in \sigma$ means that player b is known to be in the game. The atoms *accusing*, *voting*, *shooting* indicate the phase of the game.

4.1 Initial state

Initially nothing has been communicated, and the γ component and the mailbox for all agents is the empty set. The players have not yet planned any action, so the α component for all players is the empty set. Only the games-master has an action planned, viz. announcing that the players may accuse each other.

The mental state component σ is a bit different. For a mafiosa (and also the games-master) the component σ contains propositions *ismafia*(b) for all players b that belong to the mafia. Player a is a civilian if and only if *isnotmafia*$(a) \in \sigma(a)$. For a civilian a, the mental state component may contain *ismafia*(b) or *isnotmafia*(b) for some arbitrary players b, e.g. initially civilians have unspecified beliefs. The mental state component σ for all agents also contains propositions *alive*(b) for all players b.

4.2 Games-master

The games-master coordinates the game. The games-master is reactive, i.e. she performs no real reasoning to decide what actions to take. We present the program as a nondeterministic choice between alternatives.

$$
\begin{array}{lll}
\multicolumn{3}{l}{\textit{reason}_{gm}:} \\
[\!]\ \textit{eliminated}(b) \in \gamma & \longrightarrow & \gamma := \gamma \setminus \{\textit{eliminated}(b)\},\ \sigma := \sigma \setminus \{\textit{alive}(b)\} \\
[\!]\ \textit{accusing} \in \gamma & \longrightarrow & \gamma := \gamma \setminus \{\textit{accusing}\},\ \sigma := \sigma \cup \{\textit{accusing}\} \\
[\!]\ \textit{voting} \in \gamma & \longrightarrow & \gamma := \gamma \setminus \{\textit{voting}\},\ \sigma := \sigma \cup \{\textit{voting}\} \\
[\!]\ \textit{shooting} \in \gamma & \longrightarrow & \gamma := \gamma \setminus \{\textit{shooting}\},\ \sigma := \sigma \cup \{\textit{shooting}\} \\
[\!]\ \textit{AccusingDone} & \longrightarrow & \alpha := \alpha \cup \{\textit{broadcast}(\textit{voting})\}, \\
 & & \gamma := \gamma \setminus \{\textit{accuse}(b,c) \mid b,c \in \mathcal{A}_p\},\ \sigma := \sigma \setminus \{\textit{accusing}\} \\
[\!]\ \textit{VotingDone} & \longrightarrow & \alpha := \alpha \cup \{\textit{broadcast}(\textit{shooting})\} \cup \textit{VoteOffAPlayer}, \\
 & & \gamma := \gamma \setminus \{\textit{vote}(b,c) \mid b,c \in \mathcal{A}_p\},\ \sigma := \sigma \setminus \{\textit{voting}\} \\
[\!]\ \textit{ShootingDone} & \longrightarrow & \alpha := \alpha \cup \{\textit{broadcast}(\textit{accusing})\} \cup \textit{KillPlayers}, \\
 & & \gamma := \gamma \setminus \{\textit{shoot}(b,c) \mid b,c \in \mathcal{A}_p\},\ \sigma := \sigma \setminus \{\textit{shooting}\}
\end{array}
$$

where *VotingDone*, *ShootingDone* and *AccusingDone* are conditions on γ and σ of the games-master. For brevity we only define *ShootingDone*. This holds when shooting holds and all living mafiosi have fired:

$$
\begin{array}{l}
\textit{shooting} \in \sigma\ \wedge \\
(\forall\, b \in \mathcal{A}_p\,.\,\textit{ismafia}(b) \in \sigma\ \wedge\ \textit{alive}(b) \in \sigma\ \Rightarrow\ (\exists c \in \mathcal{A}_p\,.\,\textit{shoot}(b,c) \in \gamma))
\end{array}
$$

The abbreviation *VoteOffAPlayer* is a singleton set of the form $\{broadcast(eliminate(p))\}$ where p is a living player with the maximal number of votes. Similarly, *KillPlayers* is a set of actions to eliminate players that have been killed by the mafia.

4.3 Mafia

For reasons of simplicity, we assume that the mafia is not organized. A mafia-member accuses, votes and shoots at civilians as she pleases. Just like the games-master, the mental state component σ keeps track of what phase the game is in, which players have been eliminated, and who are mafiosi. The program for a mafia-member does not differ much from the games-master, but a mafiosa may only act when she is still in the game.

4.4 Civilian

We present a program for a civilian in parts. The mental state component σ represents what phase of the game an civilian is in, and which players are still in the game. The first part of the program is similar to that of a mafiosa. A civilian that is out of the game can not act.

$$
\begin{array}{lll}
[\!] \; \mathit{eliminated}(b) \in \gamma & \longrightarrow & \gamma := \gamma \setminus \{\mathit{eliminated}(b)\},\ \sigma := \sigma \setminus \{\mathit{alive}(b)\} \\
[\!] \; \mathit{accusing} \in \gamma & \longrightarrow & \gamma := \gamma \setminus \{\mathit{accusing}\},\ \sigma := \sigma \cup \{\mathit{accusing}\} \\
[\!] \; \mathit{voting} \in \gamma & \longrightarrow & \gamma := \gamma \setminus \{\mathit{voting}\},\ \sigma := \sigma \cup \{\mathit{voting}\} \\
[\!] \; \mathit{shooting} \in \gamma & \longrightarrow & \gamma := \gamma \setminus \{\mathit{shooting}\},\ \sigma := \sigma \cup \{\mathit{shooting}\}
\end{array}
$$

Next, the actions that a civilian takes are specified. These actions are prescribed by her strategy. A strategy can be seen as a complete description of an agent, that specifies how the agent reasons about new observations and chooses her actions given her mental state. For simplicity, we only consider simple strategies, that do not include higher order beliefs or beliefs on other agents' strategies.

We can decompose a strategy of a civilian into strategies for accusation, voting and updating of beliefs. For example, a civilian accuses any player that she believes to be mafia.

$$
\begin{array}{l}
[\!] \; \mathit{accusing} \in \sigma \ \wedge\ \mathit{ismafia}(b) \in \sigma \ \wedge\ \mathit{alive}(b) \in \sigma \ \longrightarrow \\
\quad \sigma := \sigma \setminus \{\mathit{accusing}\},\ \alpha := \alpha \cup \{\mathit{broadcast}(\mathit{accuse}(\mathit{self}, b))\}
\end{array}
$$

She may vote for a player that she believes to be mafia and that has been accused by some other trustworthy player.

$$
\begin{array}{l}
[\!] \; \mathit{voting} \in \sigma \ \wedge\ \mathit{ismafia}(b) \in \sigma \ \wedge\ \mathit{alive}(b) \in \sigma \ \wedge\ \mathit{accuse}(c, b) \in \gamma \ \wedge\ \mathit{isnotmafia}(c) \in \sigma \\
\quad \longrightarrow \ \sigma := \sigma \setminus \{\mathit{voting}\},\ \alpha := \alpha \cup \{\mathit{broadcast}(\mathit{vote}(\mathit{self}, b))\}
\end{array}
$$

Civilians may not shoot, so they pass over this stage of the game and may clear the past accusations and votes. One might have more complex strategies where all communication is remembered.

$$
[\!] \; \mathit{shooting} \in \sigma \ \longrightarrow \ \gamma := \gamma \setminus \{\mathit{accuse}(b, c), \mathit{vote}(b, c) \mid b, c \in \mathcal{A}_p\},\ \sigma := \sigma \setminus \{\mathit{shooting}\}
$$

Updating of beliefs can be done in various ways. Believing the accusations made by players believed not to be mafia is expressed by

$$[\!]\ \mathit{accuse}(b,c) \in \gamma \ \wedge\ \mathit{isnotmafia}(b) \in \sigma \ \wedge\ c \neq \mathit{self} \ \longrightarrow$$
$$\sigma := \sigma \cup \{\mathit{ismafia}(c)\} \setminus \{\mathit{isnotmafia}(c)\}$$

Dropping a belief that player c is mafia when c is accused by a player believed to be mafia:

$$[\!]\ \mathit{accuse}(b,c) \in \gamma \ \wedge\ \mathit{ismafia}(b) \in \sigma \ \wedge\ \mathit{ismafia}(c) \in \sigma \ \longrightarrow\ \sigma := \sigma \setminus \{\mathit{ismafia}(c)\}$$

This is by no means a complete specification of a civilian. A civilian may have multiple strategies for accusing, voting and updating of beliefs.

5 Discussion and Further Work

We have presented a simple framework for designing MAS. We do not a priori choose the mental state of an agent to be a sentence in some formal language. This gives us the flexibility e.g. to assume that the mental state contains a priority queue of aims, without the need to encode such priority queues. Also, the mental state may contain a belief base, where every belief is tagged with a fidelity factor. Of course, everything can be expressed in a suitable formal language, but we prefer to specify in the universal language of mathematics and to delay the choice of specific encodings as long as possible.

As yet, our design has no higher-order reasoning in the agents, but we plan to incorporate beliefs, desires, and strategies of an agent in her mental state, in such a form that the agent can reason about them.

The amount of parallelism of our framework is good for general MAS, but somewhat inconvenient in this specific game, since the game imposes a strict sequential ordering of the activities: all agents are in the same phase, be it accusing, voting, or shooting (being shot at). This however is an aspect of the game to be modelled and not an intrinsic property of multi-agent systems. It has been treated therefore in the game-specific code. Most of it is done in the code for the games-master. In more general, or more flexible multi-agent systems, we expect that the great potential for parallelism is an asset.

The relational, i.e. nondeterministic, flavour of our framework is very important for a successful design process. Even if one aims at deterministic agents, it is useful to program the agents first in a nondeterministic way such that they operate correctly in the given environment. In a later stage of the design, one can then reduce the nondeterminism so that the agent still operates correctly but now more according to her own strategy. The relational style of the programs is easily translated to an implementation in a declarative programming language such as Prolog.

It is important that every agent has her own set of externals to observe. This gives us the possibility of peer-to-peer communication. On a more technical level, it allows us to discard information that has been processed by the agent, so that we need no time stamps to distinguish new facts from older ones.

Even the present low-level representation in our modelling of the Mafia game will enable us to simulate the game using Prolog with several randomized strategies for the players. It will be interesting to see whether we can predict the outcomes based on the amount of Mafia members and the strategies of the civilians.

We think the Mafia game is a good test case to test frameworks and methodologies for the design of multi-agent systems. It also is an interesting case since the possibility for variations allow for further investigations in different areas of multi-agent systems.

References

[1] K.R. Apt, E.-R. Olderog: *Verification of Sequential and Concurrent Programs*. Springer-Verlag, 1991.

[2] F.M.T. Brazier, B. Dunin-Kęplicz, N.R. Jennings and J. Treur, *Formal Specification of Multi-Agent Systems: a Real-World Case*. In: V. Lesser (ed.), Proc. of the First International Conference on Multi-Agent Systems, ICMAS-95, MIT Press, Cambridge, MA, 1995, pp. 25-32.

[3] S. Franklin, A. Graesser, *Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agent*, Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, published as Intelligent Agents III, Springer-Verlag, 1997, 21-35.

[4] H.W. de Haan, W.H. Hesselink, G.R. Renardel de Lavalette, *Knowledge-based programming inspired by an asynchronous hardware leader election problem* In: B. Dunin-Kęplicz, R. Verbrugge (eds.): FAMAS'03, ETAPS 2003, Warsaw, Poland. pp. 117-132.

[5] H.W. de Haan, W.H. Hesselink, G.R. Renardel de Lavalette, *Knowledge-based asynchronous programming*, Fundamenta Informaticae, 2004 (to appear)

[6] K. Hindriks, *Agent Programming Languages: Programming with Mental Models*, Utrecht University, SIKS Dissertation Series No. 2001-2, ISBN 90-393-2590-1.

[7] `http://www.princeton.edu/˜mafia/`

[8] Z. Manna, A. Pnueli: *Temporal verification of reactive systems: safety*. Springer-Verlag, 1995.

[9] J.P. Müller, 'Control Architectures for Autonomous & Interacting Agents: A Survey', in *Intelligent Agent Systems: Theoretical and Practical Issues*, L.Cavedon, A.Rao, W.Wobcke (eds.), LNCS 1209, Springer-Verlag, Berlin, 1996, pp. 1-26.

[10] A.S. Rao, *AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language*, in W. van der Velde & J. Perram (eds.), "Agents Breaking Away", LNAI 1038, Springer-Verlag, 1996, pp. 42-55.

[11] Y. Shoham, *Agent-oriented programming*, Artificial Intelligence **60**, 1993, pp. 51-92.

[12] W. de Vries, *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems*, Utrecht University. SIKS Dissertation Series No. 2002-14, ISBN 90-393-3197-9.