Representing von Neumann-Morgenstern Games in the Situation Calculus

Oliver Schulte and James Delgrande
School of Computing Science
Simon Fraser University
Burnaby, B.C., Canada
{oschulte,jim}@cs.sfu.ca

November 28, 2002

Abstract

Sequential von Neumann-Morgernstern (VM) games are a very general formalism for representing multi-agent interactions and planning problems in a variety of types of environments. We show that sequential VM games with countably many actions and continuous utility functions have a sound and complete axiomatization in the situation calculus. This axiomatization allows us to represent game-theoretic reasoning and solution concepts such as Nash equilibrium. We discuss the application of various concepts from VM game theory to the theory of planning and multi-agent interactions, such as representing concurrent actions and using the Baire topology to define continuous payoff functions.

Keywords: multiagent systems, game theory, decision theory, reasoning about actions and change, knowledge representation

1 Introduction

Much recent research has shown that systems of interacting software agents are a useful basis for addressing issues of computational intelligence. Our best general theory of interactions between agents is the mathematical theory of games, developed to a high level since the seminal work of von Neumann and Morgenstern [von Neumann & Morgenstern, 1944]. Von Neumann and Morgenstern (VM for short) designed game theory as a very general tool for modelling agent interactions. Experience has confirmed the generality of the formalism: computer scien-

tists, economists, political scientists and others have used game theory to model a wide variety of scenarios that arise in economic, social, and political spheres.

Our aim is to represent the concepts of VM game theory in a logical formalism, so as to enable computational agents to employ game-theoretical models for the interactions that they are engaged in. It turns out that the epistemic extension $\mathcal{L}_e$ of the *situation calculus* [Levesque, Pirri, & Reiter, 1998, Sec.7] is adequate for this task. We consider VM games in which the agents can take at most countably many actions and in which the agents' payoff functions are continuous. We show that every such VM game G has an axiomatization $Axioms(G)$ in the situation calculus that represents the game in the following strong sense: The game G itself is a model of $Axioms(G)$, and all models of $Axioms(G)$ are isomorphic (that is, $Axioms(G)$ is a *categorical* axiomatization of G). It follows that the axiomatization is *correct*, in the sense that $Axioms(G)$ entails *only* true assertions about the game G, and *complete* in the sense that $Axioms(G)$ entails *all* true assertions about the game G.

This result is interesting for a number of reasons. First, the result establishes that the situation calculus is a very general language for representing multi-agent interactions. Second, it shows how to construct a sound and complete set of situation calculus axioms for a given application domain. For example, [Bart, Delgrande, & Schulte, 2001] gives an extended construction of such a set of axioms for a version of the board game "Clue" (see also Section 7). Third, it provides an agent designer with a general recipe for utilizing a game-theoretic model of a given type of interaction (e.g., Prisoner's Dilemma, Battle of the Sexes, Cournot Duopoly [Osborne & Rubinstein, 1994]) and describing the model in a logical formalism. Fourth, it opens up the potential for applying solution and search algorithms from games research to multi-agent interactions [Koller & Pfeffer, 1997, van den Herik & Iida, 2001, Bart, Delgrande, & Schulte, 2001]. Finally, game theory is a major mathematical development of the 20th century, and we expect that many of the general concepts of game theory will prove fruitful for research into intelligent agents. For example, we introduce a standard topological structure associated with games known as the *Baire topology*. The Baire topology determines the large class of continuous payoff function, which can be defined in the situation calculus in a natural way.

An attractive feature of VM game theory is that it provides a single formalism for representing both multi-agent interactions and single-agent planning problems. This is because a single-agent interaction with an environment can be modelled as a 2-player game in which one of the players—the environment—is indifferent about the outcome of the interaction. Thus our representation result includes planning problems as a special case.

After establishing the expressive power of the situation calculus for representing the structure of a multi-agent interaction, we consider agents' *reasoning* about

optimal actions in the multi-agent system. We show how to define the game-theoretic notion of a *strategy*, or policy, and introduce predicates describing which strategies are optimal in a given environment and which strategy combinations form Nash equilibria.

The paper is organized as follows. We begin with the definition of a sequential VM game. The following section introduces the situation calculus. Then we specify the set of axioms $Axioms(G)$ for a given game G, in two stages. First, we axiomatize the structure of the agents' interaction—roughly, what agents can do and what they know when. Second, we show how to define continuous utility functions in the situation calculus. For illustration we outline an extended application from previous work in which we used the situation calculus to represent a variant of the board game "Clue". (Some readers may want to look over the discussion of Clue in Section 7 before the more abstract results in the main sections.) Finally, we describe optimal strategies and Nash equilibria in a game tree.

2 Sequential Games

To represent a sequential multi-agent interaction, we need to capture three main aspects:

1. Time: The fact that agents act at different times.
2. Knowledge: An agent will have varying knowledge at different points in time.
3. Capacity: An agent can act, in different ways at different times.

Game trees combine these three aspects in one mathematical structure. First, a tree models different moments in time, as in the branching-time semantics for temporal logic [van Benthem, 1983]. Second, an epistemic accessibility relation over nodes in the tree captures which situations an agent can distinguish, as in the Kripke semantics for epistemic modal logic [Hintikka, 1962]. Game theorists take epistemic accessibility to be a partition, corresponding to the modal logic S5. Third, for each node in the game tree there is a set of associated actions that represents which actions are possible at a given stage of the game.

The formal details are as follows.

Notation and Definitions Sequential games are also known as *extensive form games* or *game trees*. The following definition is due to von Neumann, Morgenstern, and Kuhn; we adopt the formulation of [Osborne & Rubinstein, 1994, Ch.11.1]. We begin with some notation for sequences. An *infinite sequence* is

a function from the positive natural numbers to some set; we denote infinite sequences by the letter h and variants such as h' or h_i. A *finite sequence of length* n is a function with domain $1, ..., n$, denoted by the letter s and variants. Almost all the sequences we consider in this paper are sequences of actions. We denote actions throughout by the letter a and variants. We write $s = a_1, ..., a_n$ to indicate the finite sequence whose i-th member is a_i, and write $h = a_1, .., a_n, ...$ for infinite sequences. If $s = a_1, ..., a_n$ is a finite sequence of n actions, the concatenation $s * a = a_1, ..., a_n, a$ yields a finite sequence of length $n+1$. We follow the practice of set theory and write $s' \subseteq s$ to indicate that sequence s' is a prefix of s ($s' \subset s$ for proper prefixes); likewise, we write $s \subset h$ to indicate that s is a finite initial segment of the infinite sequence h. The term "sequence" without qualification applies both to finite and infinite sequences, in which case we use the letter σ and variants.

Now we are ready to define a sequential game.

Definition 2.1 (von Neumann, Morgenstern, Kuhn) *A* sequential game G *is a tuple* $\langle N, H, player, f_c, \{I_i\}, \{u_i\}\rangle$ *whose components are as follows.*

1. *A finite set* N *(the set of* players*).*
2. *A set* H *of sequences satisfying the following three properties.*
   (a) *The empty sequence* $\emptyset$ *is a member of* H.
   (b) *If* σ *is in* H*, then every initial segment of* σ *is in* H.
   (c) *If* h *is an infinite sequence such that every finite initial segment of* h *is in* H*, then* h *is in* H.

   Each member of H *is a* history*; each element of a history is an* action *taken by a player. A history* σ *is* terminal *if there is no history* $s \in H$ *such that* $\sigma \subset s$*. (Thus all infinite histories are terminal.) The set of terminal histories is denoted by* Z*. The set of actions available at a finite history* s *is denoted by* $A(s) = \{a : s*a \in H\}$.
3. *A function* player *that assigns to each nonterminal history a member of* $N \cup \{c\}$*. The function* player *determines which player takes an action after the history* s*. If* player$(s) = c$*, then it is "nature's" turn to make a chance move.*
4. *A function* f_c *that assigns to every history* s *for which* $player(s) = c$ *a probability measure* $f_c(\cdot|s)$ *on* $A(s)$*. Each probability measure is independent of every other such measure. (Thus* $f_c(a|s)$ *is the probability that "nature" chooses action* a *after the history* s*.)*

5. *For each player $i \in N$ an* information partition $\mathcal{I}_i$ *defined on* $\{s \in H : player(s) = i\}$. *An element I_i of $\mathcal{I}_i$ is called an* information set *of Player i. We require that if s, s' are members of the same information set I_i, then $A(s) = A(s')$.*

6. *For each player $i \in N$ a* payoff function $u_i : Z \rightarrow R$ *that assigns a real number to each terminal history.*

An Example To illustrate how interactions between agents may be represented as game trees, we adopt an abridged version of a scenario from [Bicchieri, Ephrati, & Antonelli, 1996]. We return to this example throughout the paper. Consider servers on the internet. Each server is connected to several sources of information and several users, as well as other servers. There is a cost to receiving and transmitting messages for the servers, which they recover by charging their users. We have two servers Srv_1 and Srv_2, and two users—journalists—U_1 and U_2. Both servers are connected to each other and to user U_1; server Srv_2 also serves user U_2. There are two types of news items that interest the users: politics and showbiz. The various costs and charges for transmissions add up to payoffs for the servers, depending on what message gets sent where. For example, it costs Srv_1 4 cents to send a message to Srv_2, and it costs Srv_2 2 cents to send a message to U_2. If U_2 receives a showbiz message from Srv_2 via Srv_1, he pays Srv_1 and Srv_2 each 6 cents. So in that case the overall payoff to Srv_1 is $-4 + 6 = 2$, and the overall payoff to Srv_2 is $-2 + 6 = 4$. Bicchieri *et al.* describe the charges in detail; we summarize them in Figure 1.

Figure 1 represents the structure of the interaction possibilities between the servers and the users. We begin with the environment delivering a type of item to the first server. If the chance p of each type of item arising is known among the players, the root node r corresponding to the empty history $\emptyset$ would be a chance node (i.e., $player(\emptyset) = c$) with associated probability p. If the probability p is unknown, we would assign the root node to an "environment" player (i.e., $player(\emptyset) = env$) who is indifferent among all outcomes (and whose payoff function hence need not be listed). Thus game theory offers two ways of representing uncertainty about the environment, depending on whether the probabilities governing the environment are common knowledge among the players or not.

Every node in the tree represents a history in the game. The nodes belonging to Srv_1 are $\{a, b\}$, and its information partition is $\mathcal{I}_1 = \{\{a\}, \{b\}\}$. Bicchieri *et al.* assume that the messages sent from Srv_1 to Srv_2 are encoded so that Srv_2 does not know which type of message it receives. Therefore the information partition of Srv_2 is $\mathcal{I}_2 = \{\{c, d\}\}$. If Srv_2 knew what type of news item it receives, its information partition would be $\mathcal{I}'_2 = \{\{c\}, \{d\}\}$. The payoff functions are as illustrated, with server Srv_1's payoffs shown on top. Thus $u_1(\emptyset * \text{Showbiz} * \text{send}$

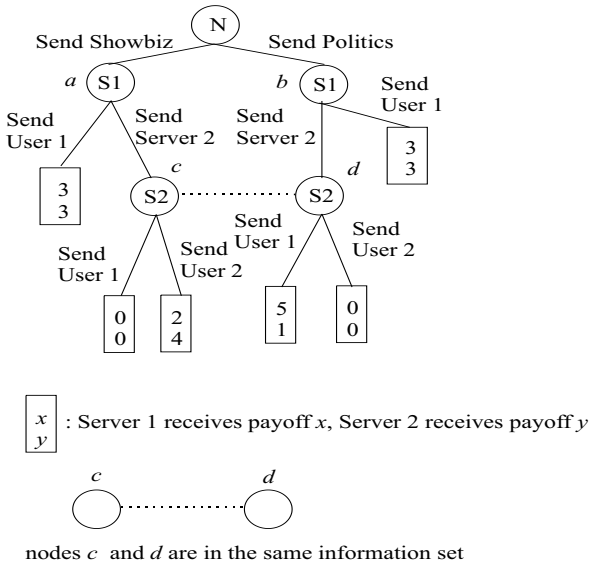

Figure 1: A game-theoretic model of the interaction between two internet servers, Srv_1 and Srv_2, with two users.

Srv_2∗send $U_2) = 2$, and $u_2(\emptyset$∗Showbiz∗send Srv_2∗send $U_2) = 4$.

The game tree of Figure 1 admits different interpretations from the one discussed so far. For example, it also represents a situation in which the messages are not encoded, but in which server Srv_2 has to make a decision independent of what news item Srv_1 sends it. In the latter case it would be more natural to reorder the game tree so that Srv_2 chooses first, then the environment, finally Srv_1. But the insight from game theory here is that as far as rational choice is concerned, we do not need to represent the "absolute" time at which the players move, but what each player *knows* when she makes a move. From this point of view, there is no important difference between a setting in which Player 1 chooses first, and then Player 2 chooses *without knowing* 1's choice, as in Figure 1, and the setting in which both players choose simultaneously (cf. [Osborne & Rubinstein, 1994, p.102]). Thus we can model simultaneous action as equivalent to a situation in which one player moves first, and the other player does not know that player's choice.

3 The Situation Calculus With Infinite Histories

Our presentation of the situation calculus follows [Levesque, Pirri, & Reiter, 1998]. We add elements for referring to infinite sequences of actions, to the foundational

situation calculus axioms. Our axiomatization result in Section 4 employs the epistemic extension of the situation calculus. We use mnemonic symbols for parts of the situation calculus according to their intended meanings. It is important to keep in mind that these are merely symbols in a formal language; it is the task of our axiomatization to ensure that the symbols carry their intended meaning. To emphasize the distinction between syntactic elements and their interpretations, we use boldface type to indicate a symbol in a formal language (although conversely, we do not use boldface for all elements of the situation calculus.) For example, $\mathbf{s}$ denotes a part of the language of the situation calculus, whereas s denotes a mathematical object, typically the intended denotation of $\mathbf{s}$. Capitalization is relatively arbitrary: constants are generally capitalized, while other terms and predicate symbols (with notable exceptions $\mathbf{K}$ and $\mathbf{Know}$) are not.

The *situation calculus with infinite histories* is a multi-sortal language that contains, in addition to the usual connectives and quantifiers of first-order logic, at least the following elements.

1. A sort *action* for actions, with variables $\mathbf{a}, \mathbf{a}'$ and constants $\mathbf{a}_i$.
2. A sort *situation* for situations, with variables $\mathbf{s}, \mathbf{s}'$.
3. A sort $infhist$ for infinite sequences of actions, with variables $\mathbf{h}, \mathbf{h}'$ etc.
4. A sort *objects* for everything else.
5. A function $\mathbf{do} : action \times situation \rightarrow situation$.
6. A distinguished constant $\mathbf{S_0} \in situation$.
7. A binary relation $\sqsubseteq$: $situation \times (situation \cup infhist)$.

   We use $\mathbf{s} \sqsubset \mathbf{s}'$ as a shorthand for $\mathbf{s} \sqsubseteq \mathbf{s}' \wedge \neg(\mathbf{s} = \mathbf{s}')$, and similarly for $\mathbf{s} \sqsubset \mathbf{h}$. The intended interpretation is that $\sqsubseteq$ denotes the relation "extends" between sequences, that is, $\sqsubseteq$ denotes $\subseteq$ as applied to sequences viewed as sets of ordered pairs (cf. Section 2).
8. A binary relation $\mathbf{poss} : action \times situation$, where $\mathbf{poss}(\mathbf{a}, \mathbf{s})$ is intended to indicate that action $\mathbf{a}$ is possible in situation $\mathbf{s}$.
9. A predicate $\mathbf{possible}(\mathbf{s})$ and $\mathbf{possible}(\mathbf{h})$.

We adopt the standard axioms for situations (see [Levesque, Pirri, & Reiter, 1998]). Here and elsewhere in the paper, all free variables are understood to be universally quantified.

$$\neg \mathbf{s} \sqsubset \mathbf{S_0} \tag{1}$$

$$\mathbf{s} \sqsubset \mathbf{do}(\mathbf{a}, \mathbf{s}') \equiv \mathbf{s} \sqsubseteq \mathbf{s}' \tag{2}$$

$$\mathbf{do}(\mathbf{a}, \mathbf{s}) = \mathbf{do}(\mathbf{a}', \mathbf{s}') \rightarrow (\mathbf{a} = \mathbf{a}') \wedge (\mathbf{s} = \mathbf{s}') \tag{3}$$

Axiom 3 ensures that every situation has a unique name. We adopt the second-order induction axiom on situations.

$$\forall \mathbf{P}.[\mathbf{P}(\mathbf{S_0}) \wedge \forall \mathbf{a}, \mathbf{s}.\mathbf{P}(\mathbf{s}) \rightarrow \mathbf{P}(\mathbf{do}(\mathbf{a}, \mathbf{s}))] \rightarrow \forall \mathbf{s}.\mathbf{P}(\mathbf{s}) \tag{4}$$

A consequence of Axiom 4 is that every situation corresponds to a finite sequence of actions (cf. [Ternovskaia, 1997, Sec. 3]).

Next we specify axioms that characterize infinite histories.

$$\mathbf{S_0} \sqsubset \mathbf{h} \tag{5}$$

$$\mathbf{s} \sqsubset \mathbf{h} \rightarrow \exists \mathbf{s}'.\mathbf{s} \sqsubset \mathbf{s}' \wedge \mathbf{s}' \sqsubset \mathbf{h} \tag{6}$$

$$(\mathbf{s}' \sqsubset \mathbf{s} \wedge \mathbf{s} \sqsubset \mathbf{h}) \rightarrow \mathbf{s}' \sqsubset \mathbf{h} \tag{7}$$

$$\mathbf{h} = \mathbf{h}' \equiv (\forall \mathbf{s}.\mathbf{s} \sqsubset \mathbf{h} \equiv \mathbf{s} \sqsubset \mathbf{h}') \tag{8}$$

$$\mathbf{possible}(\mathbf{h}) \equiv \forall \mathbf{s} \sqsubset \mathbf{h}.\mathbf{possible}(\mathbf{s}) \tag{9}$$

The final set of axioms says that the **possible** predicate defines which action sequences are possible: an action sequence is possible if no impossible action is ever taken along it.

$$\mathbf{possible}(\mathbf{S_0}) \tag{10}$$

$$\mathbf{possible}(\mathbf{do}(\mathbf{a}, \mathbf{s})) \equiv \mathbf{possible}(\mathbf{s}) \wedge \mathbf{poss}(\mathbf{a}, \mathbf{s}) \tag{11}$$

4 Representing Game Forms in the Situation Calculus

The situation calculus is a natural language for describing games because its central notion is the same as that of VM sequential games: a sequence of actions. We establish a precise sense in which the situation calculus is appropriate for formalizing VM games: every VM game G with countably many actions and continuous payoff functions has a categorical axiomatization $Axioms(G)$ in the situation calculus, thus an axiomatization that describes all and only features of the game in question.

Let $\langle N, H, player, f_c, \{I_i\}, \{u_i\}\rangle$ be a sequential game G. We use $F(G)$ to denote the tuple $\langle N, H, player, f_c, \{I_i\}\rangle$, called the *game form* of G [Osborne & Rubinstein, 1994, p.201]. The game form specifies what actions are possible at various stages of a

game, but it does not tell us how the players evaluate the outcomes. In this section, we construct a categorical axiomatization of a game form F with countably many actions $A(F)$. Section 6 considers axiomatizations of payoff functions. Our construction proceeds as follows. We begin with a game-independent axiom that captures the closure requirement of Clause 2c of Definition 2.1. Then we introduce constants for players, actions, and situations, and assign each action constant $\mathbf{a}_i$ a denotation $\lceil \mathbf{a}_i \rceil$, which defines a denotation for situation constants. Then we specify a set of axioms for the other aspects of the game in terms of the denotations of the action and situation constants.

Completeness Clause 2c of Definition 2.1 requires that in a game tree, if every finite initial segment s of an infinite history h is one of the finite histories in the game tree, then h is an infinite history in the game tree. This clause rules out, for example, a situation in which for some action a and every n, the sequence a^n is part of the game tree, but the infinite sequence a^ω is not. In such a situation we might think of the infinite sequence a^ω as "missing" from the game tree, and view clause 2c as ruling out this kind of incompleteness. (In topological terms, Clause 2c requires that the Baire topology renders a game tree a complete metric space; see Section 6 and references there). This notion of completeness is topological and different from the concept of completeness of a logical system. From a logical point of view, topological completeness is complex because it requires quantification over sequences of situations, which we represent as certain kinds of properties of situations as follows.

Let $basis(\mathbf{P})$ stand for the formula $\mathbf{P}(\mathbf{S_0})$; let $inf(\mathbf{P})$ stand for $\forall \mathbf{s}.\mathbf{P}(\mathbf{s}) \rightarrow \exists \mathbf{s}'.\mathbf{s} \sqsubset \mathbf{s}' \wedge \mathbf{P}(\mathbf{s}')$; let $closed(\mathbf{P})$ stand for $(\mathbf{s}' \sqsubset \mathbf{s} \wedge \mathbf{P}(\mathbf{s})) \rightarrow \mathbf{P}(\mathbf{s}')$, and finally let $Seq(\mathbf{P})$ stand for $basis(\mathbf{P}) \wedge inf(\mathbf{P}) \wedge closed(\mathbf{P})$. The completeness axiom corresponding to Clause 2c is then the following:

$$\forall \mathbf{P}.Seq(\mathbf{P}) \rightarrow (\exists \mathbf{h} \forall \mathbf{s}.[\mathbf{P}(\mathbf{s}) \equiv \mathbf{s} \sqsubset \mathbf{h}]) \tag{12}$$

Axiom 12 is independent of any particular game structure. Nonetheless we list it as an axiom for game trees rather than as a general axiom characterizing infinite sequences because the completeness requirement is part of the notion of a game tree, rather than part of the notion of an infinite sequence. As we saw in Section 3, the properties of infinite sequences can be defined in first-order logic, whereas the completeness requirement in the definition of a game tree requires second-order quantification.

Players We introduce a set of constants to denote players, such as $\{\mathbf{c}, \mathbf{1}, \mathbf{2}, ...\mathbf{n}\}$, and one variable $\mathbf{p}$ ranging over a new sort *players*. In the server game with

chance moves, we add constants $\mathbf{c}, \mathbf{Srv_1}, \mathbf{Srv_2}$. We introduce *unique names axioms* of the general form $\mathbf{i} \neq \mathbf{j}, \mathbf{i} \neq \mathbf{c}$ for $i \neq j$. For the server game, we add $\mathbf{c} \neq \mathbf{Srv_1}, \mathbf{c} \neq \mathbf{Srv_2}, \mathbf{S}_1 \neq \mathbf{Srv_2}$. We also add a *domain closure axiom* $\forall \mathbf{p}.\mathbf{p} = \mathbf{c} \vee \mathbf{p} = \mathbf{1} \vee ... \vee \mathbf{p} = \mathbf{n}$. In the server game, the domain closure axiom is $\forall \mathbf{p}.\mathbf{p} = \mathbf{c} \vee \mathbf{p} = \mathbf{Srv_1} \vee \mathbf{p} = \mathbf{Srv_2}$.

Actions We add a set of constants for actions. If there are finitely many actions $a_1, .., a_n$, we proceed as with the players, adding finitely many constants, unique names axioms and a domain closure axiom. For example, in the server game, we may introduce names $\mathbf{ToU_1}, \mathbf{ToU_2}, \mathbf{ToSrv_2}, \mathbf{SendShow}, \mathbf{SendPoli}$ for each action and then include axioms such as $\mathbf{ToU_1} \neq \mathbf{ToU_2}, \mathbf{SendShow} \neq \mathbf{SendPoli}$, etc. However, with infinitely many actions we cannot formulate a finite domain closure axiom. We propose a different solution for this case. Just as the induction axiom (4) for situations serves as a domain closure axiom for situations, we use a second-order induction axiom for actions to ensure that there are at most countably many actions in any model of our axioms. The formal axioms are as follows.

We introduce a successor operation $+ : action \rightarrow action$ that takes an action $\mathbf{a}$ to the "next" action $\mathbf{a}^+$. We write $\mathbf{a}^{(n)}$ as a shorthand for $\mathbf{a_0}^{+...+}$ where the successor operation is applied n times to a distinguished constant $\mathbf{a_0}$ (thus $\mathbf{a}^{(0)} = \mathbf{a}_0$). The constants $\mathbf{a}^{(n)}$ may serve as names for actions. As in the case of finitely many actions, we introduce unique names axioms of the form $\mathbf{a}^{(i)} \neq \mathbf{a}^{(j)}$ where $i \neq j$. We adopt the induction axiom

$$\forall \mathbf{P}.[\mathbf{P}(\mathbf{a_0}) \wedge \forall \mathbf{a}.\mathbf{P}(\mathbf{a}) \rightarrow \mathbf{P}(\mathbf{a}^+)] \rightarrow \forall \mathbf{a}.\mathbf{P}(\mathbf{a}). \tag{13}$$

We assume familiarity with the notion of an interpretation or model (see for example [Boolos & Jeffrey, 1974, Ch. 9]).

Lemma 1 *Let $\mathcal{M} = \langle actions, +, \lceil\rceil \rangle$ be a model of Axiom 13 and the unique names axioms. Then for every $a \in actions$, there is one and only one constant $\mathbf{a}^{(n)}$ such that $\lceil \mathbf{a}^{(n)} \rceil = a$.*

Proof (Outline). The unique names axioms ensure that for every action a, there are no two constants $\mathbf{a}^{(n)}, \mathbf{a}^{(n')}$ such that $\lceil \mathbf{a}^{(n)} \rceil = \lceil \mathbf{a}^{(n')} \rceil = a$. To establish that there is one such constant, use an inductive proof on the property of being named by a constant; more precisely, use the induction axiom to show that for every action $a \in actions$, there is some constant $\mathbf{a}^{(n)}$ such that $\lceil \mathbf{a}^{(n)} \rceil = a$.□

Now choose a function $\lceil\rceil$ that is a 1-1 and total assignment of actions in the game form $A(F)$ to action constants. For example, we may choose $\lceil \mathbf{SendShow} \rceil$ =Showbiz, $\lceil \mathbf{ToSrv_2} \rceil$ = send Srv_2, $\lceil \mathbf{ToU_1} \rceil$ = send U_1, etc.

Situations Let $\widehat{\mathbf{do}}(\mathbf{a_1}, ..., \mathbf{a}_n)$ denote the result of repeatedly applying the do constructor to the actions $\mathbf{a_1}, ..., \mathbf{a}_n$ from the initial situation. Thus $\widehat{\mathbf{do}}(\mathbf{a_1}, ..., \mathbf{a}_n)$ is shorthand for $\mathbf{do}(\mathbf{a_1}, \mathbf{do}(\mathbf{a_2}, ..., \mathbf{do}(\mathbf{a}_n))))...)$. We stipulate that $\widehat{\mathbf{do}}(\emptyset) = \mathbf{S_0}$, so that if $n = 0$ then $\widehat{\mathbf{do}}(\mathbf{a_1}, ..., \mathbf{a}_n) = \mathbf{S_0}$. The constant expressions of the form $\widehat{\mathbf{do}}(\mathbf{a_1}, ..., \mathbf{a}_n)$ serve as our names for situations. We extend the denotation $\lceil \mathbf{a}_i \rceil$ for actions to situations in the obvious way: $\lceil \widehat{\mathbf{do}}(\mathbf{a_1}, ..., \mathbf{a}_n) \rceil = \lceil \mathbf{a_1} \rceil * \cdots * \lceil \mathbf{a}_n \rceil$. For example, $\lceil \widehat{\mathbf{do}}(\mathbf{SendShow}, \mathbf{ToSrv_2}) \rceil$ =Showbiz$*$ send Srv_2. Note that $\lceil \mathbf{S_0} \rceil = \emptyset$.

Knowledge To represent knowledge, we follow [Levesque, Pirri, & Reiter, 1998, Sec.7] and employ a binary relation $\mathbf{K}$, where $\mathbf{K}(\mathbf{s}, \mathbf{s}')$ is read as "situation s is an epistemic alternative to situation s'". The relation $\mathbf{K}$ is intended to hold between two situation constants $\mathbf{s}, \mathbf{s}'$ just in case $\mathbf{s}$ and $\mathbf{s}'$ denote situations in the same information set. Thus, in the server game, $\mathbf{K}(\widehat{\mathbf{do}}(\mathbf{SendShow}, \mathbf{ToS2}), \widehat{\mathbf{do}}(\mathbf{SendPoli}, \mathbf{ToS2}))$ holds. As in [Levesque, Pirri, & Reiter, 1998, Sec.7], we use $\mathbf{Knows}(\phi, \sigma)$ (informally, the agent knows that ϕ is true in situation σ) to abbreviate $\forall \mathbf{s}, \mathbf{K}(\mathbf{s}, \sigma) \rightarrow \phi[\mathbf{s}]$, where ϕ has no situation arguments, and $\phi[\mathbf{s}]$ extends each predicate and term symbol with a situation argument.

Triviality Axioms We add axioms to ensure that all functions of situations take on a "don't care" value $\perp$ for impossible situations. For example, we would stipulate that $\mathbf{player}(\widehat{\mathbf{do}}(\mathbf{SendShow}, \mathbf{ToU1}, \mathbf{ToU2})) = \perp$ holds.

Table 1 specifies the set of axioms $Axioms(F)$ for a VM game form F. For simplicity, we assume that there are no chance moves in F (see below). We write $I(s)$ to denote the information set to which a finite game history s belongs (in F). To reduce notational clutter, the table denotes situation constants by symbols such as $\mathbf{s}_i$, with the understanding that $\mathbf{s}_i$ stands for some constant term of the form $\widehat{\mathbf{do}}(\mathbf{a_1}, ..., \mathbf{a_n})$. Let $F = \langle N, H, player, \{I_i\} \rangle$ be a game form whose set of actions is A. In set-theoretic notation, the set of all infinite sequences of actions is A^ω, and we write $A^{<\omega}$ for the set of all finite action sequences. We say that the *tree-like structure for* F is the tuple $I(F) = \langle N, A, A^{<\omega}, A^\omega, \perp, poss, possible, \subseteq, *, player', K \rangle$, where each object interprets the obvious symbol (e.g., A^ω is the sort for $\mathbf{h}$, and $*$ interprets $\mathbf{do}$), and

1. $possible(s) \iff s \in H$,
2. $K(s, s') \iff I(s) = I(s')$,
3. $player'(s) = \perp$ if $\neg possible(s)$ and $player'(s) = player(s)$ otherwise.

Axiom Set	Constraints
$\mathbf{i} \neq \mathbf{j}$	$i \neq j$
$\forall \mathbf{p}.\mathbf{p} = \mathbf{1} \vee ... \vee \mathbf{p} = \mathbf{n}$	
Actions: Unique Names Domain Closure	see text
$\mathbf{poss}(\mathbf{a_i}, \mathbf{s_j})$	$\lceil \mathbf{a}_i \rceil \in A(\lceil \mathbf{s}_j \rceil)$
$\neg\mathbf{poss}(\mathbf{a_i}, \mathbf{s_j})$	$\lceil \mathbf{a}_i \rceil \notin A(\lceil \mathbf{s}_j \rceil)$
$\mathbf{player}(\mathbf{s}_j) = \mathbf{i}$	$player(\lceil \mathbf{s}_j \rceil) = \lceil \mathbf{i} \rceil$
$\mathbf{player}(\mathbf{s}_j) = \perp$	$\lceil \mathbf{s}_j \rceil \notin H$;
$\mathbf{K}(\mathbf{s}_i, \mathbf{s}_j)$	$I(\lceil \mathbf{s}_i \rceil) = I(\lceil \mathbf{s}_j \rceil)$
$\neg\mathbf{K}(\mathbf{s}_i, \mathbf{s}_j)$	$I(\lceil \mathbf{s}_i \rceil) \neq I(\lceil \mathbf{s}_j \rceil)$

Table 1: The set $Axioms(F)$ associated with a game form $F = \langle N, H, player, \{I_i\}\rangle$ and a denotation function $\lceil\rceil$. Terms such as a_i and s_j stand for action and situation constants, respectively.

The intended interpretation for the set of axioms $Axioms(F)$, defined via the denotation function $\lceil\rceil$ as described above, is the pair $\mathcal{I} = \langle I(F), \lceil\rceil\rangle$. We note that if two interpretations are isomorphic, then the same sentences are true in both interpretations [Boolos & Jeffrey, 1974, p.191].

Theorem 4.1 *Let $F = \langle N, H, player, \{I_i\}\rangle$ be a game form. Suppose that $\lceil\rceil$ gives a 1-1 and onto mapping between action constants and $Actions(F)$.*

1. *The intended interpretation of $Axioms(F)$, defined via the denotation function $\lceil\rceil$, is a model of $Axioms(F)$ and Axioms 1 –13.*

2. *All models of $Axioms(F)$ and Axioms 1 –13 are isomorphic.*

Proof. (Outline) Part 1: It is fairly straightforward to verify that the intended interpretation for F—basically, the game tree—satisfies the general axioms and $Axioms(F)$.

For Part 2, the argument follows our construction as follows. Let $\mathcal{M} = \langle M, \lceil\rceil_{\mathcal{M}}\rangle$ be a model of $Axioms(F)$ and Axioms 1 –13. We can show that $\mathcal{M}$ is isomorphic to $\mathcal{I} = \langle I(F), \lceil\rceil\rangle$:

1. The domain closure and unique names axioms for the sorts $player$ and $action$ guarantee that there is a 1-1 mapping between A and N and the respective sorts $player$ and $action$ in $\mathcal{M}$. More precisely, let $name_{\mathcal{M}}(p)$ be the constant $\mathbf{i}$ of sort $player$ such that $\lceil \mathbf{i} \rceil_{\mathcal{M}} = p$, and define similarly $name_{\mathcal{M}}(a)$ for action a. Then the function f defined by $f(p) =$

$\lceil name_{\mathcal{M}}(p) \rceil$ takes an object of sort $player$ in $\mathcal{M}$ to an object of sort player in $\mathcal{I}$, and f is a 1-1 onto mapping. Similarly for action objects.

2. Given that every action in $\mathcal{M}$ has a unique action constant naming it, it follows from Axioms 1–4 that every situation in $\mathcal{M}$ has a unique situation constant naming it (as before, we may write $name_{\mathcal{M}}(s)$), which implies that there is a 1-1 onto mapping between the situations in $\mathcal{M}$ and $A^{<\omega}$.

3. Axioms 5–8 ensure that every object in the sort $infhist$ in $\mathcal{M}$ corresponds to an infinite sequence of (nested) situations; Axiom 12 ensures that for every such nested infinite sequence of situations, there is some object h in $infhist$ such that h extends each situation in the sequence. Given the result of step (2), it follows that there is a 1-1 onto mapping between the sort $infhist$ in $\mathcal{M}$ and A^{ω}.

4. Since $poss_{\mathcal{M}}(a, s)$ holds in $\mathcal{M}$ iff $\mathbf{poss}(name_{\mathcal{M}}(a), name_{\mathcal{M}}(s))$ is in $Axioms(F)$, which is true just in case $poss_{\mathcal{I}}(\lceil name_{\mathcal{M}}(a) \rceil, \lceil name_{\mathcal{M}}(s) \rceil)$ holds in $\mathcal{I}$, this map constitutes a 1-1 onto mapping between the extensions of $poss_{\mathcal{M}}$ and $poss_{\mathcal{I}}$. Together with Axioms 9–11, this ensures a 1-1 mapping between the extensions of the $possible$ predicate in each model.

5. We also have that $player_{\mathcal{M}}(s) = i$ iff $\mathbf{player}(name_{\mathcal{M}}(s)) = name_{\mathcal{M}}(i)$ is in $Axioms(F)$, which holds just in case $player_{\mathcal{I}}(name_{\mathcal{I}}(s)) = name_{\mathcal{I}}(i)$, so the graphs of the player function are isomorphic in each model.

6. By the same argument, the extensions of the K predicate are isomorphic in each model.

So any model $\mathcal{M}$ of the axioms $Axioms(F)$ and Axioms 1 –13 is isomorphic to the intended model $\mathcal{I}$, which shows that all models of these axioms are isomorphic. □

It follows from Theorem 4.1 that for each player i, the relation $\mathbf{K}$ from our axiomatization is an equivalence relation on the situations at which i moves, because $\mathbf{K}$ represents the information partition of Player i.

In order to represent chance moves, we need axioms characterizing real numbers; we do not want to go into axiomatic real number theory here, since that topic is well-known and the issues are similar to those that we take up in Section 6. A brief outline: start with (second-order) axioms characterizing the real numbers. There are at most countably many pairs (a, s) such that "nature" chooses action a in situation s with some probability $f_c(a|s)$. Introduce a constant for each of these probabilities, and axioms to ensure that the constant denotes the correct real numbers. Add axioms of the form $\mathbf{f}_c(\mathbf{a}, \mathbf{s}) = \mathbf{p}$ whenever $f_c(\lceil \mathbf{a} \rceil | \lceil \mathbf{s} \rceil) = \lceil \mathbf{p} \rceil$, as well as appropriate triviality axioms for $\mathbf{f}_c$.

5 Discussion and Related Work

We may evaluate a representation formalism such as the situation calculus with respect to two dimensions: expressibility—what sort of domains can the formalism model?—and tractability—how difficult is it to carry out reasoning in the formal language? Typically, there is a trade-off between these two desiderata. Our results so far indicate the expressive power of the situation calculus. Because situation calculus axioms can represent so many and such varied models of agent interactions, we cannot guarantee *in general* that the axiomatization for a given game tree is decidable, let alone tractable. An important research topic is what classes of games have tractable representations in the situation calculus (see Section 9 below). To quote David Poole, "by having a rich representation we can discuss the complexity of various restrictions and build approximation algorithms" [Poole, 1997, Section 1.2].

To illustrate this point, we note that in extant applications of the situation calculus, the $\mathbf{poss}$ predicate typically has an inductive definition in terms of what actions are possible in a situation given what fluents hold in the situation, and then what fluents obtain as a consequence of the actions taken. (We will give examples in Section 7.) By contrast, game theory deals with a set of histories, without any assumptions about how that set may be defined. The absence of state successor axioms for fluents in our axiomatization reflects the generality of VM games for representing many different types of environment. For example, we may distinguish between *static* and *dynamic* environments [Russell & Norvig, 1994, Ch 2.4]. An environment is dynamic for an agent "if the environment can change while an agent is deliberating". In a dynamic environment, a *frame axiom* that says that a fluent will remain the same unless an agent acts to change it may well be inappropriate. For example, if an agent senses that a traffic light is green at time t, it should not assume that the light will be green at time $t+1$, even if it takes no actions affecting the light. Hence a VM model of this and other dynamic environments cannot employ a simple frame axiom (cf. [Poole, 1998]).

We have followed [Levesque, Pirri, & Reiter, 1998] in introducing a binary relation $\mathbf{K}(\mathbf{s}, \mathbf{s}')$ between situations. One difference between our treatment and that of Levesque *et al*. is that they allow uncertainty about the initial situation; they introduce a predicate $\mathbf{K_0}(\mathbf{s})$ whose intended interpretation is that the situation $\mathbf{s}$ may be the initial one, for all the agent knows. It is possible to extend game-theoretic models to contain a set of game trees—often called a "game forest"—rather than a single tree. A game forest contains several different possible initial situations, one for each game tree, just as the approach of [Levesque, Pirri, & Reiter, 1998] allows different possible initial situations. Game theorists use game forests to model games of *incomplete information* in which there is some uncertainty among the

players as to the structure of their interaction [Osborne & Rubinstein, 1994].

Another difference is that Levesque *et al*. use the binary relation $\mathbf{K}$ to represent the knowledge of a *single* agent. Mathematically the single relation $\mathbf{K}$ suffices for our axiomatization, even in the multi-agent setting, because the situations are partitioned according to which player moves at them. In effect, the relation $\mathbf{K}$ summarizes a number of relations $\mathbf{K}_i$ defined on the situations at which Player i moves. To see this, define $\mathbf{K_i}(\mathbf{s}, \mathbf{s}') \iff (\mathbf{player}(\mathbf{s}) = \mathbf{player}(\mathbf{s}') = \mathbf{i}) \wedge \mathbf{K}(\mathbf{s}, \mathbf{s}')$. Then $\mathbf{K_i}$ represents a partition of Player i's nodes. From the point of view of the epistemic situation calculus, the difficulty with this is that Player i's knowledge is defined by the $\mathbf{K_i}$ relation (and hence by the $\mathbf{K}$ relation) only at situations at which it is Player i's turn to move. Thus if $\mathbf{player}(\mathbf{s}) = \mathbf{i}$, what another Player j knows at $\mathbf{s}$ is undefined and so we cannot directly represent, for example, what i knows about j's knowledge at $\mathbf{s}$. To represent at each situation what each player knows at that situation we would require a relation $\mathbf{K_i}$ for all players that is a partition of *all* situations. In terms of the game tree, we would require that the information partition I_i of Player i constitutes a partition of *all* nodes in the game tree, rather than just those at which Player i moves. The general point is that VM game trees leave implicit the representation of what a Player j knows when it is not her turn to move. This may not pose a difficulty for humans using the apparatus of VM game theory, but if we want to explicitly represent the players' knowledge at any situation, we will require additional structure, such as information partitions comprising all nodes in the game tree (see Section 7 for an illustration).

Other logical formalisms have been developed to represent classes of games. For example, Parikh's game logic uses dynamic logic to represent zero-sum games of perfect information [Parikh, 1983] (see also [Harrenstein *et al.*, 2002]). Poole's independent choice logic is a formalism for simultaneous move, or matrix, games [Poole, 1997, Sec.3.3], and the dynamic independent choice logic has resources to describe the components of a game tree [Poole, 1997, Sec.5.5]. The construction in this paper is different because it aims for almost complete generality with respect to the class of representable game trees, and because it focuses on the notion of a sequence of actions central to both game theory and the situation calculus.

6 Defining Continuous Payoff Functions in the Situation Calculus

It remains to define the payoff functions $u_i : Z \rightarrow R$ that assign a payoff for Player i to each terminal history. To simplify, we assume in this section that all terminal histories are infinite. This is no loss of generality because we can define the payoff from a finite history s as the payoff assigned to all infinite histories extending s,

where all histories extending s share the same payoff (cf. Section 8.2). There are in general uncountably many infinite histories, so we cannot introduce a name for each of them in a countable language. Moreover, payoff functions in infinite games can be of arbitrary complexity [Moschovakis, 1980], so we cannot expect all such functions to have a definition in the situation calculus. In economic applications, *continuous* payoff functions typically suffice to represent agents' preferences. Our next step is to show that all continuous payoff functions have a definition in the situation calculus provided that we extend the situation calculus with constants for rational numbers. Before we formally define continuous functions, let us consider two examples to motivate the definition.

Example 1 In a common setting, familiar from Markov decision processes, an agent receives a "reward" at each situation [Sutton & Barto, 1998, Ch.3]. Assuming that the value of the reward can be measured as a real number, an ongoing interaction between an agent and its environment gives rise to an infinite sequence of rewards $r_1, ..., r_n, ...$ A payoff function u may now measure the value of the sequence of rewards as a real number. A common measure is the *discounted sum*: We use a discount factor δ with $0 < \delta < 1$, and define the payoff to the agent from an infinite sequence of rewards by $u(r_1, ..., r_n, ...) = \sum_{i=1}^{\infty} \delta^i r_i$. For example, suppose that the agent's task is to ensure the truth of a certain fluent, for example that $\mathbf{ison}(\mathbf{light}, \mathbf{s})$ holds. We may then define the reward fluent function $\mathbf{reward}(\mathbf{s}) = \mathbf{1} \equiv \mathbf{ison}(\mathbf{light}, \mathbf{s})$ and $\mathbf{reward}(\mathbf{0}, \mathbf{s}) \equiv \mathbf{ison}(\mathbf{light}, \mathbf{s})$ so that the agent receives a reward of 1 if the light is on, and a reward of 0 otherwise. Then for an infinite history h we have an infinite sequence of rewards $\mathbf{r_1}, ..., \mathbf{r_n}, ...$ consisting of 0's and 1's. If the agent manages to keep the light on all the time, $\mathbf{reward}(\mathbf{s}) = \mathbf{1}$ will be true in all situations, and its total discounted payoff is $\sum_{i=1}^{\infty} \delta^i \times 1 = \delta/1 - \delta$.

Example 2 Suppose that the agent's job is to ensure that a certain condition never occurs. Imagine that there is an action $\mathbf{dropContainer}(\mathbf{s})$ and the agent's goal is to ensure that it never takes this action. We may represent this task specification with the following payoff function: $u(h) = 0$ if there is a situation $s \subset h$ such that $\mathbf{do}(\mathbf{dropContainer}, \mathbf{s})$ holds, and $u(h) = 1$ otherwise. It is impossible to define this payoff function as a discounted sum of bounded rewards.

The special character of the 0-1 payoff function u stems from the fact that if $\mathbf{dropContainer}$ is true at situation s, the payoff function "jumps" from a possible value of 1 to a determinate value of 0, no matter how many times beforehand the agent managed to avoid the action. Intuitively, the payoff to the agent is not built up incrementally from situation to situation. Using standard concepts from

topology, we can formalize this intuitive difference with the notion of a continuous payoff function. To that end, we shall introduce a number of standard topological notions. Our use of topology will be self-contained but terse; a text that covers the definitions we use is [Royden, 1988].

Let $\mathcal{A}, \mathcal{B}$ be two topological spaces. A mapping $f : \mathcal{A} \rightarrow \mathcal{B}$ is *continuous* if for every open set $Y \subseteq \mathcal{B}$, its preimage $f^{-1}(Y)$ is an open subset of $\mathcal{A}$. Thus to define the set of continuous payoff functions, we need to introduce a system of open sets on Z, the set of infinite histories, and R, the set of real numbers. We shall employ the standard topology on R, denoted by $\mathcal{R}$.

The *Baire topology* is a standard topology for a space that consists of infinite sequences, such as the set of infinite game histories. It is important in analysis, game theory [Moschovakis, 1980], other areas of mathematics, and increasingly in computer science [Finkel, 2001], [Duparc, Finkel, & Ressayre, 2001]. Let A be a set of actions. Let $[s] = \{h : s \subset h\}$ be the set of infinite action sequences that extend s. The basic open sets are the sets of the form $[s]$ for each finite sequence (situation) s. An open set is a union of basic open sets, including again the empty set $\emptyset$. We denote the resulting topological space by $\mathcal{B}(A)$.

For each rational q and natural number $n > 0$, define an open interval $O(q, n)$ centered at q by $O(q, n) = \{x : |x - q| < 1/n\}$. Let $E_u(q, n) = u^{-1}(O(q, n))$. Since u is continuous, each set $E_u(q, n)$ is an open set in the Baire space and hence a union of situations. So the function u induces a characteristic relation $interval_u(s, q, n)$ that holds just in case $[s] \subseteq E_u(q, n)$. In other words, $interval_u(s, q, n)$ holds iff for all histories h extending s the utility $u(h)$ is within distance $1/n$ of the rational q. In the example with the discontinuous payoff function above, $interval_u(s, 1, n)$ does *not* hold for any situation s for $n > 1$. For given any situation s at which the disaster has not yet occurred, there is a history $h \supset s$ with the disaster occurring, such that $u(h) = 0 < |1 - 1/n|$. Intuitively, with a continuous payoff function u an initial situation s determines the value $u(h)$ for any history h extending s up to a certain "confidence interval" that bounds the possible values of histories extending s. As we move further into the history h, with longer initial segments s of h, the "confidence intervals" associated with s become centered around the value $u(h)$.[1]

The following theorem exploits the fact that the collection of these intervals

[1] Another example of a discontinuous payoff function is average reward, sometimes used in place of the discounted sum [Sutton & Barto, 1998, Ch.6.7]. Given an infinite sequence of rewards $r_1, ..., r_n, ...$, the average reward at time n is given by $a_n = \sum_{i=1}^{n} r_i/n$. Then define $u(r_1, ..., r_n, ...) = \lim_{n\rightarrow\infty} a_n$ provided that this limit exists. For any finite time n, if all the rewards $r_{n+1}, ...$ received after n differ from a_n by more than ε, it will be the case that $u(r_1, ..., r_n)$ differs from a_n by more than ε. So average reward is not a continuous payoff function in the Baire utility. (We owe this example to David Poole.)

associated with situations uniquely determines a payoff function.

Assume that constants for situations, natural numbers and rationals have been defined along with a relation $\mathbf{interval_u}$ such that $\mathbf{interval_u}(\mathbf{s}, \mathbf{q}, \mathbf{n})$ is true just in case $interval_u(\lceil \mathbf{s} \rceil, \lceil \mathbf{q} \rceil, \lceil \mathbf{n} \rceil)$ holds.

Theorem 6.1 *Let A be a set of actions, and let $u : \mathcal{B}(A) \rightarrow \mathcal{R}$ be a continuous function. Let* payoff *be the axiom* $\forall \mathbf{h}, \mathbf{n}. \exists \mathbf{s} \sqsubset \mathbf{h}. \mathbf{interval}_u(\mathbf{s}, \mathbf{u}(\mathbf{h}), \mathbf{n})$. *Then*

1. *u satisfies* payoff, *and*
2. *if $u' : \mathcal{B}(A) \rightarrow \mathcal{R}$ satisfies* payoff, *then $u' = u$.*

Proof. For part 1, let a history h and a number n be given. Clearly $u(h) \in O(u(h), n)$, so $h \in E_u(u(h), n) = u^{-1}(O(q, n))$. Since $E_u(u(h), n)$ is an open set, there is a situation $s \subset h$ such that $[s] \subseteq E_u(u(h), n)$. Hence $interval_u(s, u(h), n)$ holds and s witnesses the claim.

For part 2, suppose that u' satisfies the axiom *payoff* in a model, such that for all histories h, numbers n, there is a situation $s \subset h$ satisfying $interval_u(s, u'(h), n)$. We show that for all histories h, for every n, it is the case that $|u(h) - u'(h)| < 1/n$, which implies that $u(h) = u'(h)$. Let h, n be given and, by Part 1, choose a situation $s \subset h$ satisfying $interval_u(s, u'(h), n)$. By the definition of the relation $interval_u$, it follows that $h \in E_u(u'(h), n) = u^{-1}(O(u'(h), n))$. So $u(h) \in O(u'(h), n)$, which by the definition of $O(u'(h), n)$ implies that $|u(h) - u'(h)| < 1/n$. Since this holds for any history h and number n, the claim that $u(h) = u'(h)$ follows. □

We conclude this section with two remarks: First we observe that if a VM game G has only a finite set of histories, any utility function u is continuous, so Theorem 6.1 guarantees that payoff functions for finite games are definable in the situation calculus. Second our discussion assumed that the payoff function u is defined for any infinite action sequence. In a game tree, it is only defined for possible action sequences. Impossible action sequences are easily accommodated by the Axiom $\neg possible(\mathbf{h}) \rightarrow \mathbf{u}(\mathbf{h}) = \perp$ using the "don't care" constant $\perp$, and restricting the axiom *payoff* so that it applies only to action sequences $\mathbf{h}$ such that $possible(\mathbf{h})$ holds.

7 An Application: Axioms for a Variant of "Clue"

Our results so far demonstrate that the situation calculus is an appropriate formalism for representing game-theoretic structures; they show that for a large class of games, the situation calculus can supply axioms characterizing the game structure

Fluent	Meaning
$\mathbf{holds}(\mathbf{i}, \mathbf{c_x}, \mathbf{s})$	Player i holds card x
$\mathbf{askPhase}(\mathbf{s})$	a query may be posed
$\mathbf{Know}(\mathbf{i}, \phi, \mathbf{s})$	Player i knows that ϕ

Table 2: Three fluents used to axiomatize MYST

exactly (categorically). This section outlines the representation of a fairly complex parlour game to illustrate what such axiomatizations are like, and to give a sense of the usefulness of the situation calculus in describing game structures. The discussion indicates how the situation calculus can take advantages of invariances and localities present in an environment to give a compact representation of the environmental dynamics. We show how a state successor axiom can define memory assumptions like the game-theoretic notion of perfect recall in a direct and elegant manner.

In previous work, we have studied a variant of the well-known board game "Clue", which we call MYST. For our present purposes, a brief informal outline of MYST suffices; for more details we refer the reader to [Bart, Delgrande, & Schulte, 2001], [Bart, 2000]. In particular, we will not go into the details here of defining formally the terms of our language for describing MYST. The game MYST begins with a set of cards $c_1, .., c_n$. These card are distributed among a number of players 1,2, ..., k and a "mystery pile". Each player can see her own cards, but not those of the others, nor those in the mystery pile. The players' goal is to guess the contents of the mystery pile; the first person to make a correct guess wins. The players gain information by taking turns querying each other. The queries are of the form "do you hold one of the cards from C?", where C is a set of cards. If the queried player holds none of the cards in C, he answers "no". Otherwise he shows one of the cards from C to the player posing the query. In what follows, we discuss some aspects of the axiomatization of MYST; a fuller discussion is [Bart, Delgrande, & Schulte, 2001], and [Bart, 2000] offers a full set of axioms.

Table 2 shows the fluents we discuss below together with their intended meaning.

A central fluent in our axiomatization is the *card holding fluent* $\mathbf{holds}(\mathbf{i}, \mathbf{c_x}, \mathbf{s})$, read as "Player $\mathbf{i}$ holds card $\mathbf{x}$ in situation $\mathbf{s}$". We write $\mathbf{holds}(\mathbf{0}, \mathbf{c_x}, \mathbf{s})$ to denote that card $\mathbf{x}$ is in the mystery pile in situation $\mathbf{s}$. In MYST, as in Clue, cards do not move around. This is a static, invariant aspect of the game environment that the situation calculus can capture concisely in the following successor state axiom:

$$\mathbf{holds}(\mathbf{i}, \mathbf{c_x}, \mathbf{s}) \equiv \mathbf{holds}(\mathbf{i}, \mathbf{c_x}, \mathbf{do}(\mathbf{a}, \mathbf{s})).$$

Another fact that remains invariant across situations, and that is crucial to rea-

soning about MYST, is that every card is held by exactly one player or the mystery pile. We may express this with two separate axioms.

Exclusiveness $\mathbf{holds(i, c_x, s)} \rightarrow \forall \mathbf{j} \neq \mathbf{i}.\neg\mathbf{holds(j, c_x, s)}$.

> If Player i holds card $\mathbf{c_x}$, then no other Player j (or the mystery pile) holds $\mathbf{c_x}$. If the mystery pile holds card $\mathbf{c_x}$, then $\mathbf{c_x}$ is not held by any player.

Exhaustiveness $\bigvee_{\mathbf{i=0}}^{\mathbf{p}} \mathbf{holds(i, c_x, s)}$.

> Every card is held by at least one player (or the mystery pile).

It is straightforward to specify preconditions for actions in MYST. For example, consider asking queries, one of the main actions in this game. We write $\mathbf{asks(i, q)}$ to denote that Player i takes the action of asking query $\mathbf{q}$. The fluent $\mathbf{askPhase(s)}$ expresses that the situation $\mathbf{s}$ is in an "asking phase", i.e., that no query has been posed yet. Then the precondition for asking a query is given by

$$\mathbf{poss(asks(i, Q), s)} \equiv \mathbf{askPhase(s)} \wedge \mathbf{player(s)} = \mathbf{i}.$$

We use $\mathbf{Know(i, \phi, s)}$[2] to denote that Player i knows ϕ in situation $\mathbf{s}$. Players gain information by observing the results of queries. For example, if Player 1 shows card 3 to Player 2, then Player 1 comes to know that Player 2 holds card 3. Using a fluent $\mathbf{shows(j, i, c_x)}$, this effect is a consequence of the the following knowledge axiom:

$$\mathbf{Know(i, holds(j, c_x, s), do(shows(j, i, c_x), s))}.$$

Indeed, the fact that Player 1 holds card 3 becomes *common knowledge* among the two players after Player 1 shows card 3 to Player 2. We use a common knowledge fluent indexed by a set of players to express this principle. The same fluent allows us to capture the fact that in MYST, after Player 1 shows card 3 to Player 2 in response to query $\mathbf{q}$, it becomes common knowledge among all players that Player 1 has *some* card matching the query (for more details, see [Bart, Delgrande, & Schulte, 2001, Sec. 3], [Bart, 2000]). This illustrates that the situation calculus together with epistemic logic provides resources to capture some quite subtle differential effects of actions on the knowledge and common knowledge of agents.

In our analysis of strategic reasoning in MYST, we assume that agents do not forget facts once they come to know them. This is part of the game-theoretic notion

[2] $\mathbf{Know(i, \phi, s)}$ abbreviates $\forall \mathbf{s}.\mathbf{K_i(s', s)} \rightarrow \phi[\mathbf{s}']$; recall the discussion in Section 4

of perfect recall.[3] We may render this assumption about the memory of the players by the following axiom for the knowledge fluent:

$$\mathbf{Know}(\mathbf{i}, \phi, \mathbf{s}) \rightarrow \mathbf{Know}(\mathbf{i}, \phi, \mathbf{do}(\mathbf{a}, \mathbf{s}))$$

for a sentence ϕ. Note that this definition of perfect recall relies on the fact that an agent's knowledge is defined even when it is not her turn to move, whereas in the traditional game tree, the agent's knowledge is explicitly specified only when it is her turn to move (cf. Section 5). This is an example of how the situation calculus provides resources for describing agent's knowledge that go beyond those offered by VM game trees, which in this case allow us to give a straightforward inductive definition of perfect recall.

8 Game-Theoretic Reasoning

After representing what agents know about their interaction in the axiomatization of a game tree, the next step is to describe optimal strategies in the multi-agent system. In this section, we show how to define in the situation calculus the most prominent solution concept of game theory, namely Nash equilibrium, as well as an important refinement of Nash equilibrium called subgame-perfect equilibrium. Even though we do not go into the details, the axioms we give below are such that adding them to a categorical axiomatization of a game tree yields a categorical set of axioms, one that has only isomorphic models.

8.1 Strategies and Nash Equilibria

We begin with the concept of a strategy (called "policy" in the theory of Markov Decision Processes; [Sutton & Barto, 1998, Ch.3]). We introduce two new sorts, *sit*$_i$ and *strategy*$_i$. The sort *sit*$_i$ contains the situations belonging to Player i; in terms of our axiomatization, $\mathbf{player}(\mathbf{s}) = \mathbf{i}$ for all situations $\mathbf{s} \in sit_i$. The variables $\mathbf{s}_i, \mathbf{s'}_i$ range over the sort sit_i. A strategy for Player i is a function $\pi_i : sit_i \rightarrow action$; the sort $strategy_i$ contains the strategies for Player i. In a 2-player game, the variables $\pi_{\mathbf{1}}, \pi_{\mathbf{1}}{}'$ range over strategies for Player 1 and the variables $\pi_{\mathbf{2}}, \pi_{\mathbf{2}}{}'$ range over strategies for Player 2; in a general n-player game we have variables of the form π_i for each player i.

[3] For a formal definition of perfect recall in game theory, see [Osborne & Rubinstein, 1994]. The game-theoretic definition requires that an agent should not only remember facts, but also her actions. This can be represented in the epistemic extension L_e of the situation calculus by requiring that $\mathbf{K_i}(\mathbf{s}, \mathbf{s}')$ does not hold whenever situations $\mathbf{s}, \mathbf{s}'$ differ with respect to an action by agent i.

To reduce notational clutter, free variables should be read as universally quantified, as before. Moreover, occurrences of a subscript i indicate an axiom schema in which the constants $1..n$ referring to the players may take the place of i. For example, in a 2-player game, a statement of the form $P(\pi_i) \rightarrow Q(\pi_i)$ is equivalent to $\forall \pi_{\mathbf{1}}.P(\pi_{\mathbf{1}}) \rightarrow Q(\pi_1) \wedge \forall \pi_{\mathbf{2}}.P(\pi_{\mathbf{2}}) \rightarrow Q(\pi_{\mathbf{2}})$.

Not all functions $\pi_i : sit_i \rightarrow action$ represent feasible strategies in a game tree. First, a feasible strategy cannot prescribe moves that are impossible in a given situation. Second, game theory assumes that an agent's action depends on an agent's knowledge; in terms of the game tree, if a strategy π assigns action a to history h and history h' is in the same information set as h, then π also assigns action a to history h' [Osborne & Rubinstein, 1994, Ch. 11.1]. We introduce a predicate $\mathbf{valid}$ and give the following axiom to ensure that $\mathbf{valid}$ applies only to feasible strategies.

$$\mathbf{valid}(\pi_{\mathbf{i}}) \equiv \forall \mathbf{s_i}.[\exists \mathbf{a}.\mathbf{poss}(\mathbf{a}, \mathbf{s_i}) \rightarrow \mathbf{poss}(\pi(\mathbf{s_i}), \mathbf{s_i}) \wedge \forall \mathbf{s_i}.\mathbf{K}(\mathbf{s_i}, \mathbf{s'_i}) \rightarrow \pi(\mathbf{s_i}) = \pi(\mathbf{s'_i})]$$

The first conjunct says that if a strategy π assigns action $\mathbf{a}$ to situation $\mathbf{s}$, then $\mathbf{a}$ is possible. However, a qualification is necessary because at some situations no action is possible (cf. Section 4—these are situations that don't correspond to any nodes in the game tree). So a valid strategy has to prescribe possible actions only when some possible action is available. The second conjunct captures the requirement that the agent's knowledge determines the agent's choice: if two situations are indistinguishable for the agent (i.e., if $\mathbf{K}(\mathbf{s}, \mathbf{s}')$ holds), then the agent chooses the same action at both. As David Poole puts it, "it [the robot] can only condition its action choices on what it senses ... and what it remembers" [Poole, 1998, Sec. 2.8].

In what follows, we treat 2-player games; the generalization to n-player games is obvious. We define the play sequence associated with a pair of strategies, one for each player. To attain some useful generality, we consider the play sequence that results when two strategies are "started" in a particular situation s. Thus we introduce another sort $play : strategy_1 \times strategy_2 \times situation \rightarrow infhist$. The intended meaning of $\mathbf{play}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}) = \mathbf{h}$ is that starting in situation s, if strategy π_1 and strategy π_2 are executed, then the resulting sequence of actions is h. Axiomatically, the value of $\mathbf{play}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s})$ is defined as follows.

$$\begin{array}{rcl}
\forall \mathbf{s}'.(\mathbf{s}' \sqsubseteq \mathbf{s}) & \rightarrow & (s' \sqsubset \mathbf{play}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s})). \\
(\mathbf{do}(\mathbf{a}, \mathbf{s}') \not\sqsubseteq \mathbf{s}) & \rightarrow & [(\mathbf{do}(\mathbf{a}, \mathbf{s}') \sqsubset \mathbf{play}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s})) \equiv \\
 & & \quad [\mathbf{s}' \sqsubset \mathbf{play}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}) \quad \wedge \\
 & & \quad (\mathbf{player}(\mathbf{s}') = \mathbf{1} \rightarrow \pi_{\mathbf{1}}(\mathbf{s}') = \mathbf{a}) \quad \wedge \\
 & & \quad (\mathbf{player}(\mathbf{s}') = \mathbf{2} \rightarrow \pi_{\mathbf{2}}(\mathbf{s}') = \mathbf{a})]].
\end{array}$$

The first axiom says that s and any situation preceding s is part of the infinite action sequence $play(\pi_1, \pi_2, s)$; that is, play begins with the actions in the history s. The second axiom asserts that if a situation s^* follows a situation s', then s^* is part of $play(\pi_1, \pi_2, s)$ iff:

1. s' is part of the play sequence, and
2. the action leading from s' to s^* is the one prescribed by π_1 or by π_2, depending on which player moves at s'.

It is clear from the definition that for every situation s', the two axioms determine whether or not $s' \subset play(\pi_1, \pi_2, s)$. So because an infinite action sequence h is uniquely determined by the set of situations that are initial segments of h (by Axiom 8), it follows that the two axioms given do indeed associate a unique action sequence $play(\pi_1, \pi_2, s)$ to each triple π_1, π_2, s.

Next we define the payoff to each player that results when two strategies are paired in situation $\mathbf{s}$, which is just their utility from the resulting action sequence.

$$\mathbf{U}_i(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}) = \mathbf{u}_i(\mathbf{play}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s})).$$

For the notion of Nash equilibrium we need the concept of a best reply.

$$\begin{aligned}
\mathbf{bestReply_1}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}) &\equiv \forall \pi'_{\mathbf{1}}.\mathbf{valid}(\pi'_{\mathbf{1}}) \rightarrow \mathbf{U_1}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}) \geq \mathbf{U_1}(\pi'_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}). \\
\mathbf{bestReply_2}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}) &\equiv \forall \pi'_{\mathbf{2}}.\mathbf{valid}(\pi'_{\mathbf{2}}) \rightarrow \mathbf{U_2}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}) \geq \mathbf{U_2}(\pi_{\mathbf{1}}, \pi'_{\mathbf{2}}, \mathbf{s}).
\end{aligned}$$

A pair of strategies (π_1, π_2) forms a Nash equilibrium in an overall game just in case each strategy is a best response to the other. The payoff from two strategies in the overall game is just the result of starting in the initial situation S_0. Thus we have

$$\begin{aligned}
\mathbf{nash}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}) \equiv\ & \mathbf{valid}(\pi_{\mathbf{1}}) \wedge \mathbf{valid}(\pi_{\mathbf{2}}) \wedge \\
& \mathbf{bestReply_1}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{S_0}) \wedge \mathbf{bestReply_2}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{S_0}).
\end{aligned}$$

Harrenstein *et al.* define formulas in propositional dynamic logic describing Nash equilibria [Harrenstein *et al.*, 2002]. Unlike our axiomatization, their work applies only in games of *perfect information*. In game-theoretic terms, a game tree T has perfect information if every information set I is a singleton. In terms of our axiomatizion of game trees using the epistemic situation calculus, we may define a game tree T to have perfect information iff $\forall \mathbf{s}, \mathbf{s}'.\mathbf{K}(\mathbf{s}, \mathbf{s}') \rightarrow \mathbf{s} = \mathbf{s}'$ is true in the natural model associated with the game tree (cf. Section 4). Figure 4 below shows a game tree with perfect information.

		Player 2	
		Don't confess	confess
Player 1	Don't confess	-3,-3	0,-4
	confess	-4,0	-1,-1

Figure 2: Payoffs for the Prisoner's Dilemma

Harrenstein *et al.* also consider an important special class of Nash equilibria known as *subgame perfect equilibria* [Selten, 1965], [Osborne & Rubinstein, 1994, Ch.6.2]. Subgame perfect equilibria are those strategy pairs (π_1, π_2) that are best replies to each other at every information set that is a *subgame*. A history h is the root of a subgame of a game tree T just in case for every information set I of T, either

1. all members of I are descendants of h, or
2. no members of I are descendants of h.

In a game of perfect information, every history h is the root of a subgame, whereas games of imperfect information typically do not have many subgames. Therefore subgame perfection is of special interest in perfect information games. We introduce a predicate $\mathbf{spNash}$ that applies to pairs of strategies forming a subgame perfect equilibrium. In games of perfect information, we define subgame perfection with the following axiom.

$$\begin{aligned}\mathbf{spNash}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}) \equiv\ & \mathbf{valid}(\pi_{\mathbf{1}}) \wedge \mathbf{valid}(\pi_{\mathbf{2}}) \wedge \\ & [\forall \mathbf{s}.\mathbf{poss}(\mathbf{s}) \rightarrow (\mathbf{bestReply_1}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}) \wedge \\ & \quad \mathbf{bestReply_2}(\pi_{\mathbf{1}}, \pi_{\mathbf{2}}, \mathbf{s}))].\end{aligned}$$

Example: The Prisoner's Dilemma As a simple example of the concepts defined so far, consider the well-known *Prisoner's Dilemma* in which two persons are arrested for a crime. They are separated, and each told that if they both confess each will receive three years in prison. Further, if neither confesses, each will be sentenced to one year in jail on a technicality. If one confesses and the other does not, then the confessor will be set free, and the evidence subsequently given will send the other to prison for four years. These payoffs are summarised in Figure 2.

Each agent can take one of two actions, $\mathbf{confess}$ or $\mathbf{dontConfess}$. Figure 3 gives one game tree representing the game, expressed in terms of the situation calculus and showing possible situations. Vertices are labelled by situation; thus for example situation $\mathbf{S}_4$ is the situation arrived at following the action sequence

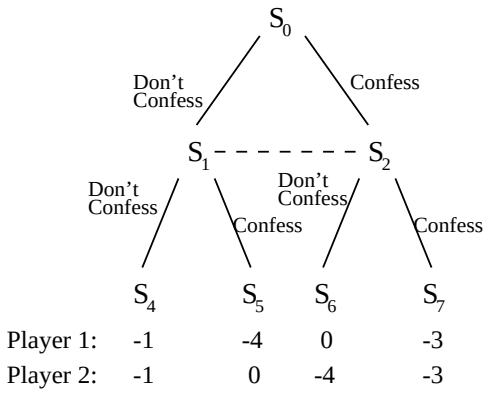

Figure 3: Payoffs for the Prisoner's Dilemma, Sequential Version

$\mathbf{do}(\mathbf{dontConfess}, \mathbf{do}(\mathbf{dontConfess}, \mathbf{S_0}))$.[4] Player 1 is assumed to make the first "move", followed by Player 2. However, in a *strategic* game like the Prisoner's Dilemma, the implicit assumption is that neither player is aware of the other's moves, or that the players move simultaneously (cf. Section 2).

Hence for Player 2, situations $\mathbf{S}_1$ and $\mathbf{S}_2$ are indistinguishable, and so Player 2 would have information partition $\mathcal{I}_2 = \{\{S_1, S_2\}\}$ and $\mathbf{K}(\mathbf{S_1}, \mathbf{S_2})$ would hold. Player 1 would have strategies

$$\begin{aligned} \pi_1^1 : \mathbf{S_0} &\rightarrow \mathbf{dontConfess} \\ \pi_1^2 : \mathbf{S_0} &\rightarrow \mathbf{confess} \end{aligned}$$

Player 2 would have a total of four strategies

$$\begin{aligned} \pi_2^1 : \mathbf{S_1} &\rightarrow \mathbf{dontConfess} \\ \mathbf{S_2} &\rightarrow \mathbf{dontConfess} \\ \pi_2^2 : \mathbf{S_1} &\rightarrow \mathbf{dontConfess} \\ \mathbf{S_2} &\rightarrow \mathbf{confess} \\ \pi_2^3 : \mathbf{S_1} &\rightarrow \mathbf{confess} \\ \mathbf{S_2} &\rightarrow \mathbf{dontConfess} \\ \pi_2^4 : \mathbf{S_1} &\rightarrow \mathbf{confess} \\ \mathbf{S_2} &\rightarrow \mathbf{confess} \end{aligned}$$

[4] Recall that there is only one situation constant, $\mathbf{S_0}$. In the example, $\mathbf{S}_4$ should be taken as *abbreviating* the term $\mathbf{do}(\mathbf{dontConfess}, \mathbf{do}(\mathbf{dontConfess}, \mathbf{S_0}))$ and not as a constant in the object language.

However, we also derive that π_2^1 and π_2^4 are the only $valid$ strategies for Player 2. Further we can derive that, for example,

$$\begin{aligned}\mathbf{play}(\pi_{\mathbf{1}}^{\mathbf{1}}, \pi_{\mathbf{2}}^{\mathbf{1}}, \mathbf{S_0}) &= \mathbf{do}(\mathbf{dontConfess}, \mathbf{do}(\mathbf{dontConfess}, \mathbf{S_0})) \quad \text{and} \\ \mathbf{play}(\pi_{\mathbf{1}}^{\mathbf{1}}, \pi_{\mathbf{2}}^{\mathbf{2}}, \mathbf{S_0}) &= \mathbf{do}(\mathbf{dontConfess}, \mathbf{do}(\mathbf{confess}, \mathbf{S_0})).\end{aligned}$$

It follows that $\mathbf{bestReply_1}(\pi_{\mathbf{1}}^{\mathbf{1}}, \pi_{\mathbf{2}}^{\mathbf{4}}, \mathbf{S_0})$ and $\mathbf{bestReply_2}(\pi_{\mathbf{1}}^{\mathbf{1}}, \pi_{\mathbf{2}}^{\mathbf{4}}, \mathbf{S_0})$ are true, and this along with $\mathbf{valid}(\pi_{\mathbf{1}}^{\mathbf{1}})$ and $\mathbf{valid}(\pi_{\mathbf{2}}^{\mathbf{4}})$ allows us to conclude $\mathbf{nash}(\pi_{\mathbf{1}}^{\mathbf{1}}, \pi_{\mathbf{2}}^{\mathbf{4}})$; that is $(confess * confess)$ is a play sequence followed in a Nash equilibrium. It can be proven that this is the only instance of $\mathbf{nash}$.

8.2 Backward Induction and Minimax Search

The classic backward induction procedure due to Zermelo is an algorithm for computing optimal strategies in a finite game of perfect information [Zermelo, 1913]. Intuitively, backward induction starts at the leaves of the tree and chooses an optimal action a at each penultimate history h. Then backward induction replaces each history h by the leaf $h * a$, yielding a tree whose height is reduced by 1, and repeats the process. In zero-sum games, backward induction is essentially the well-known minimax search algorithm [Russell & Norvig, 1994, Ch. 5.2]. In a game with perfect information, the result of backward induction coincides with the outcomes of subgame-perfect equilibria [Osborne & Rubinstein, 1994, Prop. 99.2]. Thus our definition of subgame perfect equilibrium implicitly defines the result of backward induction. We now define backward induction more directly by recursively assigning a value to situations for each player as well as a "best move". We introduce a predicate $\mathbf{leaf}$ that applies to situations which represents leaves in the game tree. Two properties define leaves:

1. no move is possible at a leaf, and
2. the leaf itself is a possible situation (cf. Axiom 11).

Thus:

$$\mathbf{leaf}(\mathbf{s}) \equiv \forall \mathbf{a}.\neg \mathbf{poss}(\mathbf{a}, \mathbf{s}) \wedge (\forall \mathbf{s}' \sqsubset \mathbf{s}.\exists \mathbf{a}.\mathbf{poss}(\mathbf{a}, \mathbf{s}')).$$

Next we employ a function $value_i : situation \rightarrow \text{real} \cup \{\perp\}$ that assigns a value for Player i to situations. For a given situation s, the number $value_i(s)$ is the payoff to Player i that would result if both players follow backward induction starting in situation s. At a leaf s, the value of s for Player i is just the payoff at s. Inductively, if s is not a leaf, $value_i(s)$ is the value of s to Player i that results when some optimal action a is executed at s, that is $value_i(s) = value_i(do(a, s))$ where

a is a move that maximizes the payoff for the player moving in situation s. In general games with imperfect information, there may not be a maximizing move, in which case we set $value_i(s) = \perp$, indicating that backward induction does not yield a value for situation s. (Even in games with perfect information, backward induction may not yield a unique value if there are ties among outcomes. However, Zermelo showed that in zero-sum games like chess, backward induction assigns a unique value to each node in the game tree; see [Osborne & Rubinstein, 1994, Ch.1.6].) Formally, we employ the following axioms characterizing the $value$ function. First, we introduce a predicate $\mathbf{bestMove}$ that applies to maximizing moves in a given situation.

$$\begin{aligned}\mathbf{bestMove}(\mathbf{a},\mathbf{s}) \quad \equiv \quad & [\mathbf{poss}(\mathbf{a},\mathbf{s}) \wedge \mathbf{player}(\mathbf{s}) = \mathbf{i} \quad \wedge \\ & \forall \mathbf{a}'.\mathbf{poss}(\mathbf{a}',\mathbf{s}) \rightarrow \mathbf{value_i}(\mathbf{do}(\mathbf{a},\mathbf{s})) \geq \mathbf{value_i}(\mathbf{do}(\mathbf{a}',\mathbf{s})) \quad \wedge \\ & \forall \mathbf{a}.\mathbf{poss}(\mathbf{a},\mathbf{s}) \rightarrow \mathbf{value_i}(\mathbf{do}(\mathbf{a},\mathbf{s})) \neq \perp]\end{aligned}$$

Informally, this axiom says that a possible action a is a best move in situation s, given a value function $value_i$ for Player i moving at s, iff the situation $do(a, s)$ that results from choosing move a yields a value that's at least as high as that from any other possible action a'. In order for $bestMove(a, s)$ to hold, $value_i(do(a', s))$ must be a real number (i.e., well-defined by backward induction) for all possible moves a'. The next axioms stipulate that at a nonleaf s, backward induction yields a definite value for Player i just in case it yields the same value no matter what best move is chosen at s.

$$\begin{aligned}\neg\mathbf{leaf}(\mathbf{s}) \quad \rightarrow \quad & [\mathbf{value_i}(\mathbf{s}) = \mathbf{x} \equiv \exists \mathbf{a}.(\mathbf{bestMove}(\mathbf{a},\mathbf{s}) \wedge \\ & \quad \forall \mathbf{a}'.\mathbf{bestMove}(\mathbf{a}',\mathbf{s}) \rightarrow \mathbf{value_i}(\mathbf{a}',\mathbf{s}) = \mathbf{x}))]. \\ \neg\mathbf{leaf}(\mathbf{s}) \quad \rightarrow \quad & [\mathbf{value_i}(\mathbf{s}) = \perp \equiv \neg\exists \mathbf{a}.(\mathbf{bestMove}(\mathbf{a},\mathbf{s}) \wedge \\ & \quad \forall \mathbf{a}'.\mathbf{bestMove}(\mathbf{a}',\mathbf{s}) \rightarrow \mathbf{value_i}(\mathbf{a}',\mathbf{s}) = \mathbf{value_i}(\mathbf{a},\mathbf{s})))]\end{aligned}$$

These conditions on the $\mathbf{value_i}$ function define the results of backward induction given an assignment of values to leaves. One way to specify an assignment of values to leaves is to employ names for the situations that are leaves (cf. Section 4) and add axioms that give the payoffs at the leaves. In terms of the utility functions u_i defined for infinite histories, it is possible to specify $value_i(s)$ for a leaf by adding the axiom $[\forall \mathbf{x}.\forall \mathbf{h}.(\mathbf{u}_i(\mathbf{h}) \neq \perp \wedge \mathbf{s} \sqsubset \mathbf{h}) \rightarrow \mathbf{u}_i(\mathbf{h}) = \mathbf{x}] \rightarrow \mathbf{value}_i(\mathbf{s}) = \mathbf{x}$. This axiom says that if all infinite histories beginning with a situation s assign utility x for Player i (whenever the utility is defined, i.e., whenever $\mathbf{possible}(\mathbf{h})$ holds - see Section 6), then the value for situation s is x.

For an example, consider a sequential variant of the Prisoner's Dilemma. In this case, Player 1 decides first whether or not to confess. Prisoner 2 observes her choice and then decides.

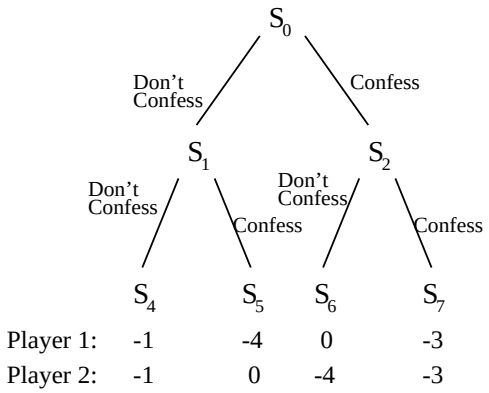

Figure 4: Payoffs for the Prisoner's Dilemma, Sequential Version with Perfect Information

Figure 4 illustrates this variant; the one difference is that we now would have $\mathcal{I}_2 = \{\{S_1\}, \{S_2\}\}$.

We derive $\mathbf{leaf}(l)$ for $l \in \{\mathbf{S_4}, \mathbf{S_5}, \mathbf{S_6}, \mathbf{S_7}\}$.

Consider $\mathbf{S_1}$: we can derive

$$\mathbf{poss}(\mathbf{confess}, \mathbf{S_1}) \quad \wedge \quad \mathbf{player}(\mathbf{S_1}) = \mathbf{2} \quad \wedge$$
$$\forall \mathbf{a}'.\mathbf{poss}(\mathbf{a}', \mathbf{S_1}) \rightarrow \mathbf{value_2}(\mathbf{do}(\mathbf{confess}, \mathbf{S_1})) \geq \mathbf{value_2}(\mathbf{do}(\mathbf{a}', \mathbf{S_1}))$$

Thus we derive $\mathbf{bestMove}(\mathbf{confess}, \mathbf{S_1})$; similarly we derive $\mathbf{bestMove}(\mathbf{confess}, \mathbf{S_2})$; Thus Player 2's best move in either $\mathbf{S_1}$ or $\mathbf{S_2}$ is to confess. Given $\neg\mathbf{leaf}(\mathbf{S_1})$, we obtain $\mathbf{value_2}(\mathbf{S_1}) = \mathbf{0}$; analogously we get $\mathbf{value_2}(\mathbf{S_2}) = \mathbf{3}$.

For $\mathbf{S_0}$, we similarly derive $\mathbf{bestMove}(\mathbf{confess}, \mathbf{S_0})$, and $\mathbf{value_1}(\mathbf{S_0}) = \mathbf{3}$. Hence under backward induction, the result of the perfect information game is no different than in the Nash equilibrium of the original simultaneous move game.

8.3 Mixed Strategies

Game theorists consider two ways of introducing stochastic elements into an agent's behaviour. First, an agent may choose a strategy with a certain probability at the beginning of the game and then execute it. The result is called a *mixed strategy*. Second, in each situation an agent might choose an action with a certain probability. In game theory, such strategies are called *behavioural strategies*; they correspond to stochastic policies in Markov Decision Processes. A well-known theorem of Kuhn's states that in games with perfect recall, the two notions of randomizing are equivalent in the sense that the same distributions over game outcomes can be

achieved with both mixed and behavioural strategies [Osborne & Rubinstein, 1994, Prop. 214.1]. Game theorists focus mainly on mixed strategies; for example, Nash's celebrated existence theorem states in any finite game (with a finite game tree), there is an equilibrium in mixed strategies [Nash, 1951]. In what follows we define Nash equilibrium for mixed strategies.

Game theorists use the term *pure strategy* to distinguish the kind of deterministic strategies that we have discussed so far (that is, the sorts $strat_i$) from mixed strategies. A mixed strategy for Player i is a probability distribution over pure strategies for Player i. Correspondingly, we introduce a sort $mix_i : strategy_i \rightarrow [0,1]$ with variable p_i ranging over mix_i.

A valid mixed strategy satisfies the standard axioms for probability measures. If there are infinitely many pure strategies (as is typically the case with infinite game trees), the probability axioms require second-order logic and introduce complications that detract from our main goal of representing game-theoretic reasoning. To avoid such complications, we will assume that the game under consideration has n pure strategies for Player 1 and m pure strategies for Player 2. Let $\pi_{\mathbf{1}}^{\mathbf{1}}, ..., \pi_{\mathbf{1}}^{\mathbf{n}}$ be names for the valid strategies of Player 1. Similarly we introduce names $\pi_{\mathbf{2}}^{\mathbf{1}}, ..., \pi_{\mathbf{2}}^{\mathbf{m}}$ for the pure strategies of Player 2. We assume domain closure and unique names axioms for these constants like those in Section 4.

For each Player i, the valid mixed strategies are those satisfying the standard probability axioms. Therefore we define the valid mixed strategies of Player 1 by the axiom

$$\mathbf{valid(p_1)} \equiv \forall \pi_{\mathbf{i}}.\mathbf{p_i}(\pi_{\mathbf{i}}) \geq \mathbf{0} \wedge \sum_{\mathbf{j=1}}^{\mathbf{n}} \mathbf{p_i}(\pi_{\mathbf{1}}^{\mathbf{j}}) = \mathbf{1}.$$

Similarly for Player 2, $\mathbf{valid(p_2)} \equiv \forall \pi_{\mathbf{i}}.\mathbf{p_2}(\pi_{\mathbf{i}}) \geq \mathbf{0} \wedge \sum_{\mathbf{j=1}}^{\mathbf{m}} \mathbf{p_i}(\pi_{\mathbf{2}}^{\mathbf{j}}) = \mathbf{1}$. The expected value for Player i from a pair of mixed strategies (p_1, p_2) is the sum of all possible payoffs to Player i weighted by the probability of obtaining that payoff. On the standard assumption that the mixed strategies p_1 and p_2 are probabilistically independent, the probability of a pure strategy combination (π_1, π_2) is given by $p_1(\pi_1) \times p_2(\pi_2)$. So the expected utility to Player 1 when the game is started in situation s is defined by $\mathbf{eU_1(p_1, p_2, s)} = \sum_{\mathbf{j=1,k=1}}^{\mathbf{j=n,k=m}} \mathbf{p_1}(\pi_{\mathbf{1}}^{\mathbf{j}}) \times \mathbf{p_1}(\pi_{\mathbf{2}}^{\mathbf{k}}) \times \mathbf{U_1}(\pi_{\mathbf{1}}^{\mathbf{j}}, \pi_{\mathbf{2}}^{\mathbf{k}}, \mathbf{s})$. Similarly $\mathbf{eU_2(p_1, p_2, s)} = \sum_{\mathbf{j=1,k=1}}^{\mathbf{j=n,k=m}} \mathbf{p_1}(\pi_{\mathbf{1}}^{\mathbf{j}}) \times \mathbf{p_1}(\pi_{\mathbf{2}}^{\mathbf{k}}) \times \mathbf{U_2}(\pi_{\mathbf{1}}^{\mathbf{j}}, \pi_{\mathbf{2}}^{\mathbf{k}}, \mathbf{s})$. The notions of best reply and Nash equilibrium now may be defined as above, with expected utility in place of payoff.

$$\begin{aligned}
\mathbf{bestReply_1(p_1, p_2, s)} &\equiv \forall \mathbf{p_1'}.\mathbf{valid(p_1)} \rightarrow \mathbf{eU_1(p_1, p_2, s)} \geq \mathbf{eU_1(p_1', p_2, s)} \\
\mathbf{bestReply_2(p_1, p_2, s)} &\equiv \forall \mathbf{p_2'}.\mathbf{valid(p_2)} \rightarrow \mathbf{eU_2(p_1, p_2, s)} \geq \mathbf{eU_2(p_1, p_2', s)}.
\end{aligned}$$

Thus we have the following definition of Nash equilibrium for mixed strategies.

$$\begin{aligned}\mathbf{nash}(\mathbf{p_1}, \mathbf{p_2}) \equiv\ & \mathbf{valid}(\mathbf{p_1}) \wedge \mathbf{valid}(\mathbf{p_2}) \\ & \wedge \mathbf{bestReply_1}(\mathbf{p_1}, \mathbf{p_2}, \mathbf{S_0}) \wedge \mathbf{bestReply_2}(\mathbf{p_1}, \mathbf{p_2}, \mathbf{S_0}).\end{aligned}$$

Similarly the definition of subgame perfection is as before with mixed in place of deterministic strategies.

9 Conclusion

Von Neumann-Morgenstern game theory is a very general formalism for representing multi-agent interactions that encompasses single-agent decision processes as a special case. Game-theoretic models of many multi-agent interactions from economics and social settings are available, and the general mathematical theory of VM games is one of the major developments of the 20th century. The situation calculus is a natural language for describing VM games; we established a precise sense in which the situation calculus is well-suited to representing VM games: a large class of VM games has a sound and complete (categorical) set of axioms in the situation calculus. This result underscores the expressive power of the situation calculus. Moreover, we showed that the situation calculus also allows us to formalize aspects of game-theoretic reasoning and decision-making, such as the concepts of Nash equilibrium and backward induction. The connection between the situation calculus and game theory suggests fruitful applications of game-theoretic ideas to planning and multi-agent interactions. We considered in particular the use of the Baire topology for defining continuous utility functions, and the representation of concurrent actions. Conversely, the logical resources of the situation calculus allow us to exploit invariants in the dynamics of some environments to provide a more compact representation of their dynamics than a game tree would, as well as potentially enabling the object-level specification of strategic knowledge in a game.

We see two main avenues for further research. First, it is important to know what types of VM games permit compact and tractable axiomatizations in the situation calculus. Without restrictions on the possible game forms F, there is no bound on the computational complexity of an axiomatization $Axioms(F)$. There are a number of plausible subclasses of games that might have computationally feasible representations in the situation calculus. An obvious restriction would be to consider finite game trees, which have first-order categorical axiomatizations in the situation calculus. A weaker finitude requirement would be that each agent should have only finitely many information sets. An information set corresponds

to what in many AI formalisms is called a "state," because an information set defines an agent's complete knowledge at the point of making a decision. Games in which the agent has only finitely many information sets closely correspond to Markov Decision Processes in which the possible states of an agent are typically assumed to be finite. Several authors have noted the utility of the situation calculus for defining Markov Decision Processes ([Poole, 1998], [Boutilier *et al.*, 2000]), and the research into compact representations of Markov Decision Processes (cf. [Boutilier, Dean, & Hanks, 1999]) may well apply to compact representations of game trees as well.

Another approach would be to begin with a decidable or tractable fragment of the situation calculus, and then examine which classes of game trees can be axiomatized in a given fragment. There are results about computable fragments of the situation calculus, even with the second-order induction axiom, such as the fragment presented in [Ternovskaia, 1999]. The main restrictions in this result are the following:

1. There are only finitely many fluents $F_1, .., F_n$;
2. each fluent F is unary, i.e. of the form $F(s)$, like our $askPhase$ fluent;
3. the truth of fluents in situation s determines the truth of fluents in a successor situation $do(a, s)$ by state successor axioms of a certain restricted form.

Our experience with MYST, and more generally the literature on Markov Decision Processes, suggests that assumptions (1) and (3) hold in many sequential decision situations; we plan to explore in future research the extent to which restriction (2) can be relaxed, and how epistemic components such as the relation $\mathbf{K}(\mathbf{s}, \mathbf{s}')$ affect computational complexity.

Second, our axiomatization of optimal strategies and Nash equilibria makes it possible to implement game-theoretic reasoning as proving theorems about optimal actions from a specification of the structure of the environment or multi-agent system. On the theoretical side, standard techniques from logic could be applied to determine the complexity of this reasoning in a given environment ([Bart, Delgrande, & Schulte, 2001] contains some complexity results for our model of Clue). On the practical side, theorem proving techniques may yield efficient implementations of decision and game theoretic reasoning.

In sum, the combination of logical techniques, such as the situation calculus, and the advanced decision-theoretic ideas that we find in game theory provides a promising foundation for analyzing planning problems and multi-agent interactions.

Acknowledgements

Part of our results were presented at the 2002 AAAI workshop on Decision and Game Theory. Previous versions were presented at JAIST (Japanese Advanced Institute for Science and Technology), the 2001 Game Theory and Epistemic Logic Workshop at the University of Tsukuba, and the AI group at the University of British Columbia. We are indebted for helpful comments to the audiences of these presentations, and to Eugenia Ternovskaia, Hiroakira Ono, Mamoru Kaneko, Jeff Pelletier and the anonymous AAAI referees. This research was supported by the National Sciences and Engineering Research Council of Canada.

References

[Bart, Delgrande, & Schulte, 2001] Bart, B.; Delgrande, J.; and Schulte, O. 2001. Knowledge and planning in an action-based multi-agent framework: A case study. In *Advances in Artificial Intelligence*, Lecture Notes in Artificial Intelligence, 121–130. Berlin: Springer.

[Bart, 2000] Bart, B. 2000. Representations of and strategies for static information, noncooperative games with imperfect information. Master's thesis, Simon Fraser University, School of Computing Science.

[Bicchieri, Ephrati, & Antonelli, 1996] Bicchieri, C.; Ephrati, E.; and Antonelli, A. 1996. Games servers play: A procedural approach. In *Intelligent Agents II*. Berlin: Springer-Verlag. 127–142.

[Boolos & Jeffrey, 1974] Boolos, G., and Jeffrey, R. 1974. *Computability and Logic*. Cambridge: Cambridge University Press.

[Boutilier *et al.*, 2000] Boutilier, C.; Reiter, R.; Soutchanski, M.; and Thrun, S. 2000. Decision-theoretic, high-level agent programming in the situation calculus. In *AAAI-2000, Seventeenth National Conference on Artificial Intelligence*. Austin, Texas: AAAI Press.

[Boutilier, Dean, & Hanks, 1999] Boutilier, C.; Dean, T.; and Hanks, S. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research* 11:1–94.

[Duparc, Finkel, & Ressayre, 2001] Duparc, J.; Finkel, O.; and Ressayre, J.-P. 2001. Computer science and the fine structure of borel sets. *Theoretical Computer Science* 257:85–105.

[Finkel, 2001] Finkel, O. 2001. Topological properties of omega context-free languages. *Theoretical Computer Science* 262:669–697.

[Harrenstein *et al.*, 2002] Harrenstein, P.; van der Hoek, W.; Meyer, J.-J.; and Witteveen, C. 2002. On modal logic interpretations of games. In van Harmelen, F., ed., *Proceedings of the 15th European Conference on Artificial Intelligence*, 28–32. Amsterdam: IOS Press.

[Hintikka, 1962] Hintikka, J. 1962. *Knowledge and Belief: An Introduction to the Logic of the two Notions*. Cornell University Press.

[Koller & Pfeffer, 1997] Koller, D., and Pfeffer, A. 1997. Representations and solutions for game-theoretic problems. *Artificial Intelligence* 94(1):167–215.

[Levesque, Pirri, & Reiter, 1998] Levesque, H.; Pirri, F.; and Reiter, R. 1998. Foundations for the situation calculus. *Linköping Electronic Articles in Computer and Information Science* 3(18).

[Moschovakis, 1980] Moschovakis, Y. 1980. *Descriptive Set Theory*. New York: Elsevier-North Holland.

[Nash, 1951] Nash, J. F. 1951. Non-cooperative games. *Annals of Mathematics* 54:286–295.

[Osborne & Rubinstein, 1994] Osborne, M. J., and Rubinstein, A. 1994. *A Course in Game Theory*. Cambridge, Mass.: MIT Press.

[Parikh, 1983] Parikh, R. 1983. Propositional game logic. In *IEEE Symposium on Foundations of Computer Science*, 195–200.

[Poole, 1997] Poole, D. 1997. The independent choice logic for modelling multiple agents under uncertainty. *Artificial Intelligence* 94(1–2).

[Poole, 1998] Poole, D. 1998. Decision theory, the situation calculus, and conditional plans. *Linköping Electronic Articles in Computer and Information Science* 3(8).

[Royden, 1988] Royden, H. L. 1988. *Real Analysis*. New York: Macmillan, 3 edition.

[Russell & Norvig, 1994] Russell, S. J., and Norvig, P. 1994. *Artificial Intelligence: A modern approach*. Englewood Cliffs, NJ: Prentice Hall.

[Selten, 1965] Selten, R. 1965. Spieltheoretische behandlung eines oligopolmodells mit nachfrageträgheit. *Zeitschrift für die gesamte Staatswissenschaft* 121:301–324 and 667–689.

[Sutton & Barto, 1998] Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning: an introduction*. Cambridge, Mass.: MIT Press.

[Ternovskaia, 1997] Ternovskaia, E. 1997. Inductive definability and the situation calculus. In *Transactions and Change in Logic Databases*, volume 1472. LNCS.

[Ternovskaia, 1999] Ternovskaia, E. 1999. Automata theory for reasoning about actions. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*. Stockholm, Sweden: IJCAI. 153–158.

[van Benthem, 1983] van Benthem, J. 1983. *The Logic of Time*. Synthese Library, Volume 156. D. Reidel Pub. Co.

[van den Herik & Iida, 2001] van den Herik, H., and Iida, H. 2001. Foreword for special issue on games research. *Theoretical Computer Science* 252,1-2:1–3.

[von Neumann & Morgenstern, 1944] von Neumann, J., and Morgenstern, O. 1944. *Theory of Games and Economic Behavior*. Princeton, NJ: Princeton University Press.

[Zermelo, 1913] Zermelo, E. 1913. Über eine anwendung der mengenlehre auf die theorie des schachspiels. In *Proceedings of the Fifth International Congress of Mathematicians*, volume II. Cambridge: Cambridge University Press.