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

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

Introduction

Much recent research has shown that systems of interacting software agents are a useful basis for 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 scientists, economists, political scientists and others have used game theory to model hundreds 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 makes a number of contributions to Artificial Intelligence. First, it shows how to construct a sound and complete set of situation calculus axioms for a given application domain. Bart *et al.* give an extended construction of such a set of axioms for a version of the board game "Clue" (Bart, Delgrande, & Schulte 2001), which we outline in the penultimate section. Second, the result establishes that the situation calculus is a very general language for representing multi-agent interactions. 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.

The paper is organized as follows. We begin with the definition of a sequential VM game. The next 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. Finally, 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 this discussion in the next-to-last section, before the more abstract results in the main sections.)

Sequential Games

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 sequences 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 0.1 (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 of 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.)*

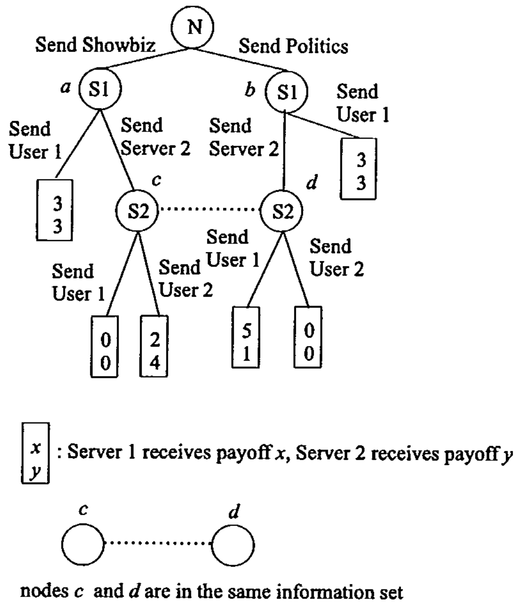

Figure 1: A game-theoretic model of the interaction between two internet servers, S_1 and S_2, with two users.

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*** *$u_i : Z \to R$ 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 S_1 and S_2, and two users—journalists—U_1 and U_2. Both servers are connected to each other and to user U_1; server S_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 S_1 4 cents to send a message to S_2, and it costs S_2 2 cents to send a message to U_2. If U_2 receives a showbiz message from S_2 via S_1, he pays S_1 and S_2 each 6 cents. So in that case the overall payoff to S_1 is $-4 + 6 = 2$, and the overall payoff to S_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 S_1 are $\{a, b\}$, and its information partition is $\mathcal{I}_i = \{\{a\}, \{b\}\}$. Bicchieri *et al.* assume that the messages sent from S_1 to S_2 are encoded so that S_2 does not know which type of message it receives. Therefore the information partition of S_2 is $\mathcal{I}_2 = \{\{c, d\}\}$. If S_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 S_1's payoffs shown on top. Thus $u_1(\emptyset*\text{Showbiz}*\text{send } S_2*\text{send } U_2) = 2$, and $u_2(\emptyset*\text{Showbiz}*\text{send } S_2*\text{send } U_2) = 4$.

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 S_2 has to make a decision independent of what news item S_1 sends it. In the latter case it would be more natural to reorder the game tree so that S_2 chooses first, then the environment, finally S_1. But the insight from game theory here is that for rational choice 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.202, 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 choice.

The Situation Calculus With Infinite Histories

Our presentation of the situation calculus follows (Levesque, Pirri, & Reiter 1998). To the foundational situation calculus axioms we add elements for referring to infinite sequences of actions. Our axiomatization result in the next section applies to 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 the intended meaning. To emphasize the distinction between syntactic elements and their interpretations, we use boldface type, which will always indicate a symbol in a formal language (although conversely, we do not use boldface for all elements of the situation calculus.)

The **situation calculus with infinite histories** is a multi-sortal language that contains, in addition to the usual connectives and quantifiers, 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 *inhist* for infinite sequences of actions, with variables $\mathbf{h}, \mathbf{h}'$ etc.
4. A sort *objects* for everything else.
5. A function $do : action \times situation \rightarrow situation$.
6. A distinguished constant $\mathbf{S}_0 \in situation$.
7. A relation $\sqsubseteq$: $situation \times (situation \cup inhist)$. 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.
8. A predicate $poss(\mathbf{a}, \mathbf{s})$, intended to indicate that action $\mathbf{a}$ is possible in situation $\mathbf{s}$.
9. A predicate $possible(\mathbf{s})$ and $possible(\mathbf{h})$.

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.

$$\neg \mathbf{s} \sqsubset \mathbf{S}_0 \tag{1}$$

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

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

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

$$\forall P.[P(\mathbf{S}_0) \wedge \forall \mathbf{a}, \mathbf{s}.P(\mathbf{s}) \rightarrow P(do(\mathbf{a}, \mathbf{s}))] \rightarrow \forall \mathbf{s}.P(\mathbf{s}) \tag{4}$$

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.

$$\mathbf{S}_0 \sqsubset \mathbf{h} \tag{5}$$

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

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

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

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

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.

$$possible(\mathbf{S}_0) \tag{10}$$

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

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 apt 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. The tuple $\langle N, H, player, f_c, \{I_i\}\rangle$ is the **game form** of G (Osborne & Rubinstein 1994, p.201), which we denote as $F(G)$. 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 with countably many actions $A(F)$. We consider axiomatizations of payoff functions in a later section. Our construction proceeds as follows. 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 0.1 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 the section on Defining Continuous Payoff Functions 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 $basis(P)$ stand for the formula $P(\mathbf{S}_0)$, let $inf(P)$ stand for $\forall \mathbf{s}.P(\mathbf{s}) \rightarrow \exists \mathbf{s}'.\mathbf{s} \sqsubset \mathbf{s}' \wedge P(\mathbf{s}')$, let $closed(P)$ stand for $(\mathbf{s}' \sqsubset \mathbf{s} \wedge P(\mathbf{s})) \rightarrow P(\mathbf{s}')$, and finally let $Seq(P)$ stand for $basis(P) \wedge inf(P) \wedge closed(P)$. The completeness axiom corresponding to Clause 2c is then the following:

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

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 the section on the introduction to the situation calculus, 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 $\mathbf{c}, \mathbf{S}_1, \mathbf{S}_2$. 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 $\mathbf{c} \neq \mathbf{S}_1, \mathbf{c} \neq \mathbf{S}_2, \mathbf{S}_1 \neq \mathbf{S}_2$. 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 $\forall \mathbf{p}.\mathbf{p} = \mathbf{c} \vee \mathbf{p} = \mathbf{S}_1 \vee \mathbf{p} = \mathbf{S}_2$.

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 $\mathbf{U}_1, \mathbf{U}_2, \mathbf{2S}_2, \mathbf{Show}, \mathbf{Poli}$ for each action and then include axioms such as $\mathbf{U}_1 \neq \mathbf{U}_2, \mathbf{Show} \neq \mathbf{Poli}$, 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 $\mathbf{a}_0^{+\cdots+}$ 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

$$\forall P.[P(\mathbf{a}_0) \wedge \forall \mathbf{a}.P(\mathbf{a}) \rightarrow P(\mathbf{a}^+)] \rightarrow \forall \mathbf{a}.P(a). \qquad (13)$$

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 $\lceil \mathbf{Show} \rceil$ = Showbiz, $\lceil \mathbf{2S}_2 \rceil$ = send S_2, $\lceil \mathbf{U}_1 \rceil$ = send U_1, etc.

Situations. Let $\widehat{do}(\mathbf{a}_1, ..., \mathbf{a}_n)$ denote the result of repeatedly applying the do constructor to the actions $\mathbf{a}_1, ..., \mathbf{a}_n$ from the initial situation. Thus $\widehat{do}(\mathbf{a}_1, ..., \mathbf{a}_n)$ is shorthand for $do(\mathbf{a}_1, do(\mathbf{a}_2, ..., do(\mathbf{a}_n))))...)$. We stipulate that

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

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.

$\widehat{do}(\emptyset) = \mathbf{S}_0$ so that $\widehat{do}(\mathbf{a}_1, ..., \mathbf{a}_n) = \mathbf{S}_0$ if $n = 0$. The constant expressions of the form $\widehat{do}(\mathbf{a}_1, ..., \mathbf{a}_n)$ serve as our names for situations. We extend the denotation $\lceil \mathbf{a}_i \rceil$ for actions to situations in the obvious way: $\lceil \widehat{do}(\mathbf{a}_1, ..., \mathbf{a}_n) \rceil = \lceil \mathbf{a}_1 \rceil * \cdots * \lceil \mathbf{a}_n \rceil$. For example, $\lceil \widehat{do}(\mathbf{Show}, \mathbf{2S_2}) \rceil =$ Showbiz$*$ send S_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. For example, in the server game we have that $\mathbf{K}(\widehat{do}(\mathbf{Show}, \mathbf{2S2}), \widehat{do}(\mathbf{Poli}, \mathbf{2S2}))$ holds.

Triviality Axioms. We add axioms to ensure that all functions of situations take on a "don't care" value $\bot$ for impossible situations. For example, $\mathbf{player}(\widehat{do}(\mathbf{Show}, \mathbf{U1}, \mathbf{U2})) = \bot$ 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 $\widehat{do}(\mathbf{a}_1, ..., \mathbf{a}_n)$. 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 $I(F) = \langle N, A, A^{<\omega}, A^\omega, \bot, poss, possible, \subseteq, *, player', K\rangle$, 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. $player'(s) = \bot$ if $\neg possible(s)$ and $player'(s) = player(s)$ otherwise.

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 0.1 *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 1-1onto mapping between the situations in $\mathcal{M}$ and $A^{<\omega}$. (3) Axioms 5–8 ensure that every object in the sort $inhist$ 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 $inhist$ 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 $inhist$ 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 0.1 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 the section Defining Continuous Payoff Functions. 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 $\mathbf{f_c(a, s) = p}$ whenever $f_c(\ulcorner \mathbf{a} \urcorner | \ulcorner \mathbf{s} \urcorner) = \ulcorner \mathbf{p} \urcorner$, as well as appropriate triviality axioms for $\mathbf{f_c}$.

Discussion and Related Work

We may evaluate a representation formalism such as the situation calculus with respect to two dimensions: Expressive power—how many 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 how great the expressive power of the situation calculus is. 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 tractable. An important research topic is what classes of games have tractable representations in the situation calculus (see the conclusions section).

To illustrate this point, we note that in extant applications of the situation calculus, the *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 the next section.) 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 $\mathbf{K(s, s')}$ 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 $K_0(s)$ whose intended interpretation is that the situation 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 $\mathbf{K}_i(s, s') \iff (\mathbf{player(s)} = \mathbf{player(s')} = i) \wedge \mathbf{K(s, s')}$. 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 $player(s) = i$, what another player j knows at s is undefined and so we cannot directly represent, for example, what i knows about j's knowledge at s. To represent at each situation what each player knows at that situation we would require a relation 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.

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). 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 more 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.

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, whenever all histories extending s share the same payoff. 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 by 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 $ison(light, s)$ holds. We may then define the reward fluent function $reward(s) = 1 \equiv ison(light, s)$ and $reward(0, s) \equiv ison(light, s)$ 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 $r_1, ..., r_n, ...$ consisting of 0s and 1s. If the agent manages to keep the light on all the time, $reward(s) = 1$ 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 $DropContainer(s)$ 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 $do(DropContainer, s)$ 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 $DropContainer$ 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 $u(h)$. The following theorem exploits the fact that the collection of these intervals 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 $\mathbf{interval_u(s, q, n)}$ is true just in case $interval_u(\lceil s \rceil, \lceil q \rceil, \lceil n \rceil)$ holds.

Theorem 0.2 *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* $\forall \mathbf{h}, \mathbf{n}. \exists \mathbf{s} \sqsubset \mathbf{h}. \mathbf{interval_u(s, u(h), n)}$. *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

Fluent	Meaning
$H(i, c_x, s)$	player i holds card x
$AskPhase(s)$	a query may be posed
$Know(i, \phi, s)$	player i knows that ϕ

Table 2: Three fluents used to axiomatize MYST

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 remark that if a VM game G has only a finite set of histories, any utility function u_i is continuous, so Theorem 0.2 guarantees that payoff functions for finite games are definable in the situation calculus.

An Application: Axioms for a Variant of "Clue"

Our results so far provide a general mathematical foundation for the claim that the situation calculus is a strong 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 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 previous work (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. 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* $H(i, c_x, s)$, read as "player i holds card x in situation s". We write $H(0, c_x, s)$ to denote that card x is in the mystery pile in situation 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:

$$H(i, c_x, s) \equiv H(i, c_x, do(a, s))$$

Another fact that remains invariant across situations, and that is crucial to reasoning 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 $H(i, c_x, s) \rightarrow \forall j \neq i. \neg H(j, c_x, s)$.

If player i holds card c_x, then no other player j (or the mystery pile) holds c_x. If the mystery pile holds card c_x, then c_x is not held by any player.

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

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 $asks(i, q)$ to denote that player i takes the action of asking query q. The fluent $AskPhase(s)$ expresses that the situation 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

$$Poss(asks(i, Q), s) \equiv AskPhase(s) \wedge player(s) = i$$

where $player(s)$ indicates whose turn it is in situation s.

We use an *epistemic fluent* $Know(i, \phi, s)$ to denote that player i knows ϕ in situation s, where ϕ is a sentence. 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 $shows(j, i, c_x)$, we may express this effect by the axiom $Knows(2, H(1, c_3, s), do(shows(1, 2, c_3), s))$. In general, we have the following knowledge axiom that is invariant across situations:

$$Know(i, H(j, c_x, s), do(shows(j, i, c_x), s).$$

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 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 of perfect recall.[1] We may render this assumption about the memory of the players by the following axiom for the knowledge fluent:

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

for a sentence ϕ.

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. 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 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.

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.

[1]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 $K_i(s, s')$ does not hold whenever situations s, s' differ with respect to an action by agent i.

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 that 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 (such as our axiom from the previous section for the card holding fluent; see (Levesque, Pirri, & Reiter 1998) for more on the proof-theoretic and computational leverage from state successor axioms). 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, the main purpose of providing an agent with a model of its planning problem or interaction with other agents is to help the agent make decisions. Game theorists have introduced various models of rational decision-makings in VM games, some of which have been applied by computer scientists (see (Koller & Pfeffer 1997), (Bicchieri, Ephrati, & Antonelli 1996)). It should be possible to axiomatize methods for solving games in the situation calculus, and to formalize the assumptions—such as common knowledge of rationality—that game theorists use to derive predictions for how agents will act in a given game.

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.

Acknowledgments

Previous versions of this paper were presented at JAIST (Japanese Advanced Institute for Science and Technology) and the 2001 Game Theory and Epistemic Logic Workshop at the University of Tsukuba. We are indebted to Eugenia Ternovskaia, Hiroakira Ono, Mamoru Kaneko, Jeff Pelletier and the anonymous AAAI referees for helpful comments. This research was supported by the National Sciences and Engineering Research Council of Canada.

References

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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