Multi-Agent Influence Diagrams for Representing and Solving Games

Daphne Koller
Computer Science Dept.
Stanford University
Stanford, CA 94305-9010
koller@cs.stanford.edu

Brian Milch*
Computer Science Dept.
Stanford University
Stanford, CA 94305-9010
milch@cs.stanford.edu

Abstract

Game theory provides a theoretical basis for defining rational behavior in multi-agent scenarios. However, the computational complexity of finding equilibria in large games has limited game theory's practical applications. In this paper, we explore the application of structured probabilistic models to multi-agent scenarios. We define *multi-agent influence diagrams (MAIDs)*, which represent games in a way that allows us to take advantage of independence relationships among variables. This representation allows us to define a notion of *strategic relevance*: D' is *strategically relevant* to D if, to optimize the decision rule at D, the decision maker needs to know the decision rule at D'. We provide a sound and complete graphical criterion for determining strategic relevance. We then show how strategic relevance, which is made explicit by the MAID structure, can be exploited in algorithms for computing equilibria to obtain exponential savings in certain classes of games.

Introduction

Game theory (Fudenberg & Tirole 1991) provides a mathematical framework for determining what behavior is rational for agents interacting with each other in a partially observable environment. However, the computational complexity of finding equilibria in games has hindered the application of game theory to real-world problems. Most algorithms for finding equilibria operate on the strategic form of a game, whose size is typically exponential in the size of the game tree (McKelvey & McLennan 1996). Even algorithms that operate directly on the game tree (Romanovskii 1962; Koller, Megiddo, & von Stengel 1994) are not a solution, as for games involving many decisions, the game tree can be prohibitively large. As an extreme case, if the game tree is symmetric (i.e., all paths to leaves have the same sequence of decisions), the number of leaf nodes is exponential in the number of decisions.

This problem of state space explosion also occurs in decision trees, the single-agent versions of game trees. *Influence diagrams* (IDs) (Howard & Matheson 1984) allow single-agent decision problems to be represented and solved without this exponential blow-up. Like Bayesian networks (Pearl 1988), influence diagrams represent the conditional independencies among variables. These formalisms allow us to take advantage of the structure in real-world problems to solve them more efficiently. In this paper, we present an extension of influence diagrams to the multi-agent setting, with the goal of providing compact representations and more efficient solution algorithms for game-theoretic problems.

Multi-agent influence diagrams (MAIDs) represent decision problems involving multiple agents in a structured way. We show that MAIDs allow us to exploit not only the dependencies between the probabilistic attributes in the domain (as in Bayesian networks), but also the dependencies between *strategic* variables. We define a notion of *strategic relevance*: a decision variable D strategically relies on another decision variable D' when, to optimize the decision rule in D, the decision-making agent needs to know the decision rule in D'. We provide a graph-based criterion, which we call *s-reachability*, for strategic relevance, and show that it is sound and complete in the same sense that d-separation is sound and complete for probabilistic dependence. We also provide a polynomial time algorithm for computing s-reachability.

The notion of strategic relevance allows us to define a data structure we call the *relevance graph* — a directed graph that indicates when one decision variable in the MAID relies on another. We then show that this data structure can be exploited to provide more efficient algorithms for computing equilibria: it allows a large game to be broken up into several smaller games, whose equilibria can be combined to obtain a global equilibrium.

Multi-Agent Influence Diagrams (MAIDs)

We will introduce MAIDs using a simple two-agent scenario:

Example 1 *Alice is considering building a patio behind her house, and the patio would be more valuable to her if she could get a clear view of the ocean. Unfortunately, there is a tree in her neighbor Bob's yard that blocks her view. Being somewhat unscrupulous, Alice considers poisoning Bob's tree, which might cause it to become sick. Bob cannot tell whether Alice has poisoned his tree, but he can tell if the tree is getting sick, and he has the option of calling in a tree doctor (at some cost). The attention of a tree doctor reduces*

*Now at Google Inc.

the chance that the tree will die during the coming winter. Meanwhile, Alice must make a decision about building her patio before the weather gets too cold. When she makes this decision, she knows whether a tree doctor has come, but she cannot observe the health of the tree directly. A MAID for this scenario is shown in Figure 1.

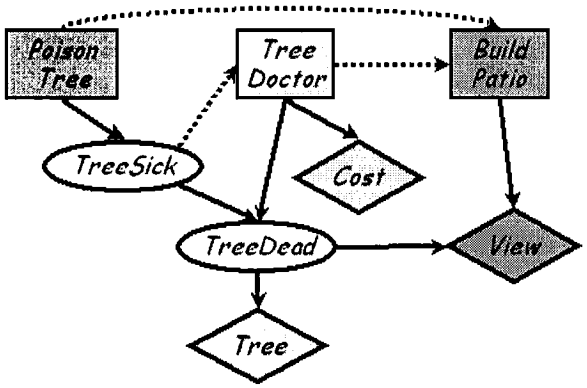

Figure 1: MAID for the Tree Killer example. Alice's decision and utility variables are in dark gray and Bob's in light gray.

To define a MAID more formally, we begin with a set $\mathcal{A}$ of agents. The world in which the agents act is represented by the set $\mathcal{C}$ of *chance variables*, and a set $\mathcal{D}_a$ of *decision variables* for each agent $a \in \mathcal{A}$. Chance variables correspond to moves by nature, and are represented in the diagram as ovals. The decision variables for agent a are variables whose values a gets to choose, and are represented as rectangles in the diagram. We use $\mathcal{D}$ to denote $\bigcup_{a \in \mathcal{A}} \mathcal{D}_a$, and $\mathcal{V}$ to denote $\mathcal{C} \cup \mathcal{D}$. Each variable $V \in \mathcal{V}$ has a finite set $dom(V)$ of possible values, called its *domain*. The agents' utility functions are specified using *utility variables*: For each agent $a \in \mathcal{A}$, we have a set $\mathcal{U}_a$ of utility variables, represented as diamonds in the diagram, that take on real numbers as values.

A MAID defines a directed acyclic graph with its variables as the nodes, where each variable X is associated with a set of parents $Pa(X) \subset \mathcal{V}$. Note that utility variables cannot be parents of other variables. For each chance variable $X \in \mathcal{C}$, the MAID specifies a *conditional probability distribution (CPD)*: a distribution $\Pr(X \mid \mathbf{pa})$ for each instantiation $\mathbf{pa}$ of $Pa(X)$. For a decision variable $D \in \mathcal{D}_a$, $Pa(D)$ is the set of variables whose values agent a knows when he chooses a value for D. Thus, the choice agent a makes for D can be contingent only on these variables. The value of a utility variable U is a deterministic function of the values of $Pa(U)$; we use $U(\mathbf{pa})$ to denote the utility value for node U when $Pa(U) = \mathbf{pa}$. For each utility variable $U \in \mathcal{U}$, f specifies, for each instantiation $\mathbf{pa}$ of $Pa(U)$, the corresponding utility value $f[U](\mathbf{pa})$, which we abbreviate using $U(\mathbf{pa})$. The total utility that an agent a derives from an instantiation of $\mathcal{V}$ is the sum of the utilities specified by the utility variables in $\mathcal{U}_a$ for this instantiation. Thus, by breaking an agent's utility function into several variables, we are simply defining an additive decomposition of the agent's utility function (Howard & Matheson 1984; Keeney & Raiffa 1976).

The behavior of the agents is defined by a set of *decision rules*.

Definition 1 *A* decision rule *for a decision variable D is a function that maps each instantiation $\mathbf{pa}$ of $Pa(D)$ to a probability distribution over $dom(D)$. An assignment of decision rules to every decision $D \in \mathcal{D}_a$ for a particular agent $a \in \mathcal{A}$ is called a* strategy.

An assignment σ of decision rules to every decision $D \in \mathcal{D}$ is called a *strategy profile*. A *partial strategy profile* $\sigma_{\mathcal{E}}$ is an assignment of decision rules to a subset $\mathcal{E}$ of $\mathcal{D}$. We will also use $\sigma_{\mathcal{E}}$ to denote the restriction of σ to $\mathcal{E}$, and $\sigma_{-\mathcal{E}}$ to denote the restriction of σ to variables not in $\mathcal{E}$.

Note that a decision rule has exactly the same form as a CPD. Thus, if we have a MAID $\mathcal{M}$, then a partial strategy profile σ that assigns decision rules to a set $\mathcal{E}$ of decision variables induces a new MAID $\mathcal{M}[\sigma]$ where the elements of $\mathcal{E}$ have become chance variables. That is, each $D \in \mathcal{E}$ corresponds to a chance variable in $\mathcal{M}[\sigma]$ with $\sigma(D)$ as its CPD. When σ assigns a decision rule to every decision variable in $\mathcal{M}$, the induced MAID is simply a BN: it has no more decision variables. This BN defines a joint probability distribution $P_\sigma^{\mathcal{M}}$ over all the variables in $\mathcal{M}$.

With this probability distribution, we can now write an equation for the utility that agent a expects to receive in a MAID $\mathcal{M}$ if the agents play a given strategy profile σ:

$$\mathrm{EU}_a(\sigma) = \sum_{U \in \mathcal{U}_a} \sum_{\mathbf{pa} \in \times Pa(U)} U(\mathbf{pa}) P_\sigma^{\mathcal{M}}(\mathbf{pa}) \qquad (1)$$

where $\times Pa(U)$ is the joint domain of $Pa(U)$.

In the game-theoretic framework, we typically consider a strategy profile to represent rational behavior if it is a *Nash equilibrium* (Nash 1950). Intuitively, a strategy profile is a Nash equilibrium if no agent has an incentive to deviate from the strategy specified for him by the profile, as long as the other agents do not deviate from their specified strategies.

Definition 2 *A strategy profile σ is a* Nash equilibrium *for a MAID $\mathcal{M}$ if for all agents $a \in \mathcal{A}$ and all strategies $\sigma'_{\mathcal{D}_a}$: $EU_a(\sigma) \geq EU_a\left((\sigma_{-\mathcal{D}_a}, \sigma'_{\mathcal{D}_a})\right)$.*

MAIDs and Games

A MAID provides a compact representation of a scenario that can also be represented as a game in strategic or extensive form. In this section, we discuss how to convert a MAID into an extensive-form game. We also show how, once we have found an equilibrium strategy profile for a MAID, we can convert it into a behavior strategy profile for the extensive form game. The word "node" in this section refers solely to a node in the tree, as distinguished from the nodes in the MAID.

We use a straightforward extension of a construction of (Pearl 1988) for converting an influence diagram into a decision tree. The basic idea is to construct a tree with splits for decision and chance nodes in the MAID. We need to split on a chance variable before its value is observed by

some decision node; we need only split on chance variables that are observed at some point in the process. More precisely, the set of variables included in our game tree is $\mathcal{G} = \mathcal{D} \cup \bigcup_{D \in \mathcal{D}} Pa(D)$.

We begin by defining a total ordering $\prec$ over $\mathcal{G}$ that is consistent with the topological order of the MAID: if there is a directed path from X to Y, then $X \prec Y$. Our tree $\mathcal{T}$ is a symmetric tree, with each path containing splits over all the variables in $\mathcal{G}$ in the order defined by $\prec$. Each node is labeled with a partial instantiation $inst(N)$ of $\mathcal{G}$, in the obvious way. For each agent a, the nodes corresponding to variables $D \in \mathcal{D}_a$ are decision nodes for a; the other nodes are all chance nodes. To define the information sets, consider two decision nodes M and M' that correspond to a variable D. We place M and M' into the same information set if and only if $inst(M)$ and $inst(M')$ assign the same values to $Pa(D)$.

To determine the probabilities for the chance nodes, we must do probabilistic inference. It turns out that we can choose an arbitrary fully mixed strategy profile σ for our MAID $\mathcal{M}$ (one where no decision has probability zero), and do inference in the BN $\mathcal{M}[\sigma]$ induced by this strategy profile. Consider a chance node N corresponding to a chance variable C. For each value $c \in dom(C)$, let N_c be the child of N corresponding to the choice $C = c$. Then we define the probability of going from N to N_c as $P_\sigma^{\mathcal{M}}(inst(N_c) \mid inst(N))$. Note that if we split on a decision variable D before C, then the decision rule σ_D does not affect this computation because $inst(N)$ includes values for D and all its parents. If we split on D after C, then D cannot be an ancestor of C in the MAID, and $inst(N)$ cannot specify evidence on D or any of its descendants. Therefore, σ_D cannot affect the computation. Hence, the probabilities of the chance nodes are well-defined.

We define the payoffs at the leaves by computing a distribution over the parents of the utility nodes, given an instantiation of $\mathcal{G}$. For a leaf N, the payoff for agent a is:

$$\sum_{U \in \mathcal{U}_a} \sum_{\mathbf{pa} \in \times Pa(U)} U(\mathbf{pa}) P_\sigma^{\mathcal{M}}(\mathbf{pa} \mid inst(N))$$

The mapping between MAIDs and trees also induces an obvious mapping between strategy profiles in the different representations. A MAID strategy profile specifies a probability distribution over $dom(D)$ for each pair $(D, \mathbf{pa})$, where $\mathbf{pa}$ is an instantiation of $Pa(D)$. The information sets in the game tree correspond one-to-one with these pairs, and a behavior strategy in the game tree is a mapping from information sets to probability distributions. Clearly the two are equivalent.

Based on this construction, we can now state the following equivalence lemma:

Lemma 1 *Let $\mathcal{M}$ be a MAID and $\mathcal{T}$ be its corresponding game tree. Then for any strategy profile σ, the payoff vector for σ in $\mathcal{M}$ is the same as the payoff vector for σ in $\mathcal{T}$.*

The number of nodes in $\mathcal{T}$ is exponential in the number of decision variables, and in the number of chance variables that are observed during the course of the game. While this blowup is unavoidable in a tree representation, it can be quite significant in certain games. Thus, a MAID can be exponentially smaller than the extensive game it corresponds to.

Example 2 *Suppose a road is being built from north to south through undeveloped land, and n agents have purchased plots of land along the road. As the road reaches each agent's plot, the agent needs to choose what to build on his land. His utility depends on what he builds, on some private information about the suitability of his land for various purposes, and on what is built north, south, and across the road from his land. The agent can observe what has already been built immediately to the north of his land (on both sides of the road), but he cannot observe further north; nor can he observe what will be built across from his land or south of it.*

The MAID representation is very compact. There are n chance nodes, corresponding to the private information about each agent's land, and n decision variables. Each decision variable has at most three parents: the agent's private information, and the two decisions regarding the two plots to the north of the agent's land. Thus, the size of the MAID is linear in n. Conversely, any game tree for this situation must split on each of the n chance nodes and each of the n decisions, leading to a representation that is exponential in n.

A MAID representation is not always more compact. If the game tree is naturally asymmetric, a naive MAID representation can be exponentially larger than the tree. We return to this problem in Section .

Strategic Relevance

To take advantage of the independence structure in a MAID, we would like to find a global equilibrium through a series of relatively simple local computations. The difficulty is that in order to determine the optimal decision rule for a single decision variable, we usually need to know the decision rules for some other variables. In Example 1, when Alice is deciding whether to poison the tree, she needs to compare the expected utilities of her two alternatives. However, the probability of the tree dying depends on the probability of Bob calling a tree doctor if he observes that the tree is sick. Thus, we need to know the decision rule for *CallTreeDoctor* to determine the optimal decision rule for *PoisonTree*. In such situations, we will say that *PoisonTree (strategically) relies on CallTreeDoctor*, or that *CallTreeDoctor* is *relevant to PoisonTree*. On the other hand, *CallTreeDoctor* does not rely on *PoisonTree*. Bob gets to observe whether the tree is sick, and *TreeDead* is conditionally independent of *PoisonTree* given *TreeSick*, so the decision rule for *PoisonTree* is not relevant to Bob's decision.

We will now formalize this intuitive discussion of strategic relevance. Suppose we have a strategy profile, and we would like to find a decision rule for a single decision variable $D \in \mathcal{D}_a$ that maximizes a's expected utility, assuming the rest of the strategy profile remains fixed.

Definition 3 *Let δ be a decision rule for a variable D in a MAID $\mathcal{M}$. Then δ is* optimal for a strategy profile σ *if, in the induced MAID $\mathcal{M}[\sigma_{-D}]$ (where the only remaining decision node is D), the strategy profile (δ) is a Nash equilibrium.*

Given this definition of optimality for a strategy profile, we can now define strategic relevance.

Definition 4 *A decision variable D* strategically relies on *a decision variable D' in a MAID $\mathcal{M}$ if there are two strategy profiles σ and σ' such that σ and σ' differ only at D', but some decision rule for D is optimal for σ and not for σ'.*

If D does *not* rely on D', then a decision rule for D that is optimal for a strategy profile σ is also optimal for any strategy profile σ' that differs from σ only at D'.

It turns out that strategic relevance corresponds to a simple graph-theoretic property in a MAID. To begin with, suppose we have a strategy profile σ for a MAID $\mathcal{M}$, and consider finding an optimal decision rule for D in $\mathcal{M}[\sigma_{-D}]$, where D is the only decision node. The optimality of the decision rule at D depends only on the utility nodes $\mathcal{U}_D$ that are descendants of D in the MAID. The other utility nodes are irrelevant, because the decision at D cannot influence them. Now, based on the independence properties implied by the graph structure, we can prove the following lemma:

Lemma 2 *Let δ be a decision rule for a decision variable $D \in \mathcal{D}_a$ in a MAID $\mathcal{M}$, and let σ be a strategy profile for $\mathcal{M}$. Then δ is optimal for σ if and only if for every instantiation $\boldsymbol{pa}_D$ of $Pa(D)$ where $P_\sigma^{\mathcal{M}}(\boldsymbol{pa}_D) > 0$, the probability distribution $\delta(D \mid \boldsymbol{pa})$ is:*

$$\operatorname*{argmax}_{P^*} \sum_{U \in \mathcal{U}_D} \sum_{\boldsymbol{pa}_U \in \times Pa(U)} U(\boldsymbol{pa}_U) \cdot \sum_{d \in dom(D)} P^*(d) P_\sigma^{\mathcal{M}}(\boldsymbol{pa}_U \mid d, \boldsymbol{pa}_D) \quad (2)$$

So to be optimal for a strategy profile σ, a decision rule δ just has to satisfy Equation 2. If the expression being maximized in Equation 2 is independent of the decision rule that σ assigns to another decision variable D', then D does not rely on D'. Thus, we would like a graphical criterion for detecting when this expression is independent of the decision rule for some node. The appropriate formal notion turns out to be that of a *requisite probability node*.

Definition 5 *Let $\mathcal{X}$ and $\mathcal{Y}$ be sets of variables in the directed acyclic graph defined by a BN or MAID. Then a node Z is a* requisite probability node *for the query $P(\mathcal{X} \mid \mathcal{Y})$ if there exist two functions P_1 and P_2 that assign CPDs to all the nodes in the graph, such that P_1 and P_2 differ only at Z, but $P_1(\mathcal{X} \mid \mathcal{Y}) \neq P_2(\mathcal{X} \mid \mathcal{Y})$.*

Intuitively, the decision rule at D' is only relevant to D if D' (viewed as a chance node) is a relevant probability node for $P(\mathcal{U}_D \mid D, Pa(D))$.

Geiger *et al.* (1990) provide a graphical criterion for testing whether a node Z is a requisite probability node for a query $P(\mathcal{X} \mid \mathcal{Y})$. We add to Z a new "dummy" parent $\widehat{Z}$ whose values correspond to CPDs for Z, selected from some set of possible CPDs. Then Z is a requisite probability node for $P(\mathcal{X} \mid \mathcal{Y})$ if and only if $\widehat{Z}$ can influence $\mathcal{X}$ given $\mathcal{Y}$. This condition can easily be checked using the standard notion of active paths in the BN. Thus, Z is a requisite probability node for $P(\mathcal{X} \mid \mathcal{Y})$ if and only if there would be an active path from a new parent of Z to $\mathcal{X}$, given $\mathcal{Y}$.

Based on this analysis, we can define *s-reachability*, a graphical criterion for detecting strategic relevance. Note that unlike d-separation in Bayesian networks, s-reachability is not necessarily a symmetric relation.

Definition 6 *A node D' is* s-reachable *from a node D in a MAID if there is some utility node $U \in \mathcal{U}_D$ such that if a new parent $\widehat{D'}$ were added to D', there would be an active path from $\widehat{D'}$ to $Pa(U)$ given $Pa(D) \cup \{D\}$.*

As we now show, s-reachability is sound and complete for strategic relevance in the same sense that d-separation is sound and complete for independence in Bayesian networks.

Theorem 1 (Soundness) *If D and D' are two decision nodes in a MAID and D' is not s-reachable from D in the MAID, then D does not rely on D'.*

Theorem 2 (Completeness) *If a node D' is s-reachable from a node D in a MAID, then there is some MAID with the same graph structure in which D relies on D'.*

As for BNs, the completeness result is somewhat weaker: s-reachability does not imply relevance in *every* MAID. We can choose the probabilities and utilities in the MAID in such a way that the influence of one decision rule on another does not manifest itself. However, s-reachability is the most precise graphical criterion we can use: it will not identify a strategic relevance unless that relevance actually exists in some MAID that has the given graph structure. Our proof of this theorem involves constructing an appropriate assignment of CPDs and utility functions to the MAID, and depends on the completeness proof in (Geiger, Verma, & Pearl 1990).

Since strategic relevance is a binary relation, we can represent it as a directed graph. As we show below, this graph turns out to be extremely useful.

Definition 7 *The* relevance graph *for a MAID $\mathcal{M}$ is a graph whose nodes are the decision nodes of $\mathcal{M}$, and whose edges are the pairs of nodes (D, D') such that D relies on D'.*

To construct the graph for a given MAID, we need to determine, for each decision node D, the set of nodes D' that are s-reachable from D. Using an algorithm such as Shachter's Bayes-Ball (Shachter 1998), we can find this set for any given D in time linear in the number of nodes in the MAID. By repeating the algorithm for each D, we can derive the relevance graph in time quadratic in the number of MAID nodes.

It is interesting to consider the relevance graphs of various simple games. In the examples in Figure 2, the decision node D belongs to agent a, and D' belongs to agent b. Example (a) represents a perfect-information game. Since agent b can observe the value of D, he does not need to know the decision rule for D in order to evaluate his options. Thus, D' does not rely on D. On the other hand, agent a cannot observe D' when she makes decision D, and D' is relevant to a's utility, so D relies on D'. Example (b) represents a game where the agents do not have perfect information: agent b cannot observe D when making decision D'. However, the

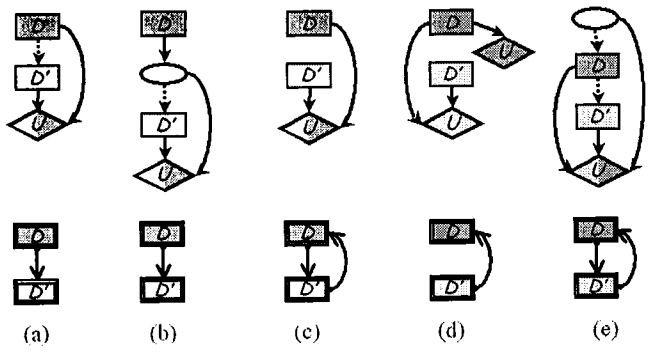

Figure 2: Five simple MAIDs (top), and their relevance graphs (bottom). A two-color diamond represents a pair of utility nodes, one for each agent, with the same parents.

information is "perfect enough": the utility for b does not depend on D directly, but only on the chance node, which b can observe. Hence D' does not rely on D.

Examples (c) and (d) represent scenarios where the agents move simultaneously, and thus neither can observe the other's move. In (c), each agent's utility node is influenced by both decisions, so D relies on D' and D' relies on D. Thus, the relevance graph is cyclic. In (d), however, the relevance graph is acyclic despite the fact that the agents move simultaneously. The difference here is that agent a no longer cares what agent b does, because her utility is not influenced by b's decision. In graphical terms, there is no active path from D' to a's utility node given D.

One might conclude that a decision node D' never relies on a decision node D when D is observed by D', but the situation is more subtle. Consider example (e), which represents a simple card game: agent a observes a card, and decides whether to bet (D); agent b observes only agent a's bet, and decides whether to bet (D'); the utility of both depends on their bets and the value of the card. Even though agent b observes the actual decision in D, he needs to know the decision rule for D in order to know what the value of D tells him about the chance node. Thus, D' relies on D; indeed, when D is observed, there is an active path from D that runs through the chance node to the utility node.

Computing Equilibria using Divide & Conquer

The computation of a Nash equilibrium for a game is arguably the key computational task in game theory. In this section, we show how the structure of the MAID can be exploited to provide substantially faster algorithms for finding equilibria.

The key insight behind our algorithm is the use of the relevance graph to break up the task of finding an equilibrium into a series of subtasks, each over a much smaller game. Since algorithms for finding equilibria in general games have complexity that is superlinear in the number of levels in the game tree, breaking the game into smaller games will significantly improve the complexity of finding a global equilibrium.

The algorithm is a generalization of existing backward induction algorithms for decision trees and perfect information games (Zermelo 1913) and for influence diagrams (Jensen, Jensen, & Dittmer 1994). The basic idea is as follows: in order to optimize the decision rule for D, we need to know the decision rule for all decisions D' that are relevant for D. For example, the relevance graph for the Tree Killer example (Figure 3(a)) shows that to optimize *PoisonTree*, we must first decide on the decision rules for *BuildPatio* and *TreeDoctor*. However, we can optimize *TreeDoctor* without knowing the decision rules for either of the other decision variables. Having decided on the decision rule for *TreeDoctor*, we can now optimize *BuildPatio* and then finally *PoisonTree*.

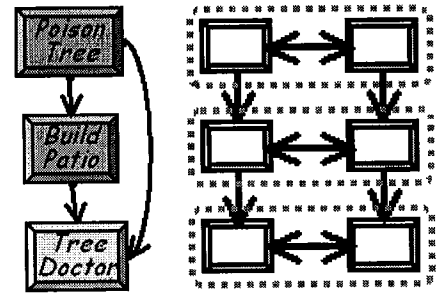

Figure 3: Relevance graphs for (a) the Tree Killer example; (b) the Road example with $n = 6$.

In this simple case, the relevance graph is acyclic, allowing us to process the variables one at a time. In general, however, we might have cycles in the relevance graph; e.g., Figure 3(b) contains the relevance graph for the Road example with $n = 6$. Here, we cannot sequentially optimize individual decisions, because at various points in the process, there is no decision node that only relies on decisions for which we have already obtained a decision rule. However, we can perform a backward induction process over subsets of nodes:

Definition 8 *A set S of nodes in a directed graph is a* strongly connected component *(SCC) if for every pair of nodes $D \neq D' \in S$, there exists a directed path from D to D'. A* maximal *SCC is an SCC that is not a strict subset of any other SCC.*

The maximal SCCs are outlined in Figure 3(b). We can find the maximal SCCs of a graph in linear time, constructing a *component graph* whose nodes are the maximal SCCs of the relevance graph. There is an edge from component C to component C' in the component graph if and only if there is an edge from some element of C to some element of C' in the relevance graph. The component graph is always acyclic (Cormen, Leiserson, & Rivest 1990). Thus, we can define an ordering $C_1, \ldots, C_m$ over the maximal SCCs of the relevance graph, such that whenever $i < j$, no element of C_j relies on any element of C_i.

Based on this definition, we can now provide a divide-and-conquer algorithm for computing Nash equilibria in MAIDs.

1 Let σ^0 be an arbitrary fully mixed strategy profile
2 For $i = 0$ through $m - 1$:
3 Let τ be a partial strategy profile for $\mathcal{C}_{(m-i)}$ that is a Nash equilibrium in $\mathcal{M}\left[\sigma^i_{-\mathcal{C}_{(m-i)}}\right]$
4 Let $\sigma^{i+1} = (\sigma^i_{-\mathcal{C}_{(m-i)}}, \tau)$
5 Output σ^m as an equilibrium of $\mathcal{M}$

The algorithm iterates backwards over the SCC's, finding an equilibrium strategy profile for each SCC in the MAID induced by the previously selected decision rules (with arbitrary decision rules for some decisions that are not relevant for this SCC). Finding the equilibrium in this induced MAID requires the use of a subroutine for finding equilibria in games. We simply convert the induced MAID into a game tree, as described in Section , and use a standard game-solving algorithm (McKelvey & McLennan 1996).

If the relevance graph is acyclic, each SCC consists of a single node, so each decision is optimized by itself. In this case, the algorithm is very similar to the standard backward induction algorithm in perfect information games. However, we obtain acyclic relevance graphs in a wider range of situations. For example, the relevance graph of the Tree Killer example is acyclic, although the game does not have perfect information.

The proof that this algorithm is correct requires a sequence of lemmas. So far we have considered strategic relevance only as a binary relation between nodes. Suppose δ_1 is a decision rule for D_1 that is optimal for a strategy profile σ, and D_1 relies on neither D_2 nor D_3. Then changing σ at either D_2 or D_3 does not affect the optimality of δ. But one might worry that changing σ at *both* D_2 and D_3 might cause δ to lose optimality. The following lemma shows that such a thing cannot happen: if we have a set of decisions none of which are individually relevant, then the entire set is not relevant.

Lemma 3 *Let σ be a strategy profile, D be a decision node, and δ be a decision rule for D that is optimal for σ. If σ' is another strategy profile such that $\sigma(D') = \sigma'(D')$ whenever D relies on D', then δ is also optimal for σ'.*

Proof: We proceed by induction on the number k of nodes where σ and σ' differ. For the base case, $k = 0$ and $\sigma = \sigma'$, so it is obvious that δ is also optimal for σ'. Now as an inductive hypothesis, assume the lemma holds whenever σ and σ' differ on exactly k nodes. Then suppose σ and σ' differ on $k+1$ nodes. Select any node D' such that $\sigma(D') \neq \sigma'(D')$. By the conditions of the lemma, this means D does not rely on D'. Now construct a strategy profile $\tilde{\sigma}$ that agrees with σ on all nodes but D', and agrees with σ' on D'. Because σ and $\tilde{\sigma}$ differ only on D', δ must also be optimal for $\tilde{\sigma}$; otherwise D would rely on D'. But $\tilde{\sigma}$ and σ' differ on only k nodes, so by the inductive hypothesis, δ is optimal for σ' as well. ■

Thus, we can change the decision rules for a whole set of nodes that are irrelevant for D without affecting the optimality of a decision rule for D. But in our divide and conquer algorithm, we are concerned with the optimality of a partial strategy profile τ for an entire set $\mathcal{C}$ of decision nodes. Clearly, if none of the decision nodes in $\mathcal{C}$ rely on a node $D' \notin \mathcal{C}$, then changing the decision rule for D' would not give any agent an incentive to deviate at a single decision node in $\mathcal{C}$. But might an agent want to deviate at several decision nodes simultaneously?

We can answer this question in the negative if we make the standard assumption of *perfect recall*: agents never forget their previous actions or observations. More formally:

Definition 9 *An agent a has* perfect recall *with respect to a total order $\prec$ over $\mathcal{D}_a$ if for all $D, D' \in \mathcal{D}_a$, $D \prec D'$ implies that $D \in Pa(D')$ and $Pa(D) \subset Pa(D')$.*

Note that if D and its parents are parents of D', then there cannot be an active path from a new parent $\widehat{D}$ of D to any descendant of D', given D' and $Pa(D')$. Thus, by Theorem 1, D' does not rely on D. This observation leads to the following lemma:

Lemma 4 *If an agent has perfect recall with respect to a total order $\prec$, and $D \prec D'$, then D' does not rely on D.*

We can now show that if a single agent has no incentive to deviate at a single decision node, then he has no incentive to deviate at any group of nodes.

Lemma 5 *Let $\mathcal{M}$ be a MAID with a single agent a, who has perfect recall. Let σ be a strategy for a in $\mathcal{M}$ such that for every $D \in \mathcal{D}$, $\sigma(D)$ is optimal for σ. Then σ is a Nash equilibrium in $\mathcal{M}$.*[1]

Proof: To show that σ is a Nash equilibrium, we must show that for all other strategies σ':

$$\mathrm{EU}_a(\sigma) \geq \mathrm{EU}_a(\sigma')$$

We proceed by induction on the number k of decisions where σ and σ' differ. If $k = 0$, then $\sigma = \sigma'$, so obviously $\mathrm{EU}_a(\sigma) = \mathrm{EU}_a(\sigma')$. As an inductive hypothesis, suppose that whenever σ' differs from σ on k or fewer nodes, $\mathrm{EU}_a(\sigma) \geq \mathrm{EU}_a(\sigma')$.

Now suppose σ' differs from σ on $k+1$ nodes. Since a has perfect recall, Lemma 4 tells us there is a total order $\prec$ over $\mathcal{D}$ such that whenever $D \prec D'$, D' does not rely on D. Let D^* be the last decision in this ordering such that $\sigma'(D^*) \neq \sigma(D^*)$. That is, if $\sigma'(D) \neq \sigma(D)$, then either $D = D^*$ or $D \prec D^*$. In either case, D^* does not rely on D. Let $\delta = \sigma(D)$, and $\delta' = \sigma'(D)$. We are given that δ is optimal for σ. But since σ' differs from σ only at nodes that D^* does not rely on, we know by Lemma 3 that δ is also optimal for σ'. In particular, δ yields at least as much expected utility as δ' in $\mathcal{M}\left[\sigma'_{-D^*}\right]$. So:

$$\mathrm{EU}_a\left(\sigma'_{-D^*}, \delta\right) \geq \mathrm{EU}_a(\sigma')$$

[1] The notion of Nash equilibrium in the single agent case reduces to the simpler notion of the agent acting optimally under uncertainty.

But the strategy (σ'_{-D^*}, δ) differs from σ at only k decision nodes, so by the inductive hypothesis:

$$\mathrm{EU}_a(\sigma) \geq \mathrm{EU}_a\left(\sigma'_{-D^*}, \delta\right)$$

So by transitivity:

$$\mathrm{EU}_a(\sigma) \geq \mathrm{EU}_a(\sigma')$$

■

Given this result, we can show that changing the decision rules for nodes that are not relevant to any node in a set C does not affect the optimality of a partial strategy profile over C.

Lemma 6 *Let σ be a strategy profile for a MAID $\mathcal{M}$ where every agent has perfect recall, and let τ be a partial strategy profile for a set C of decision nodes in $\mathcal{M}$. Suppose τ is a Nash equilibrium in $\mathcal{M}[\sigma_{-C}]$, and $\mathcal{E}$ is a set of decision nodes in $\mathcal{M}$ (disjoint from C) such that no node in C relies on any node in $\mathcal{E}$. Then if σ' is another strategy profile that differs from σ only on $\mathcal{E}$, τ is also a Nash equilibrium in $\mathcal{M}[\sigma'_{-C}]$.*

Proof: For each agent a, let $C_a = C \cap \mathcal{D}_a$. If $C_a \neq \emptyset$, let τ_a be the restriction of τ to C_a, and τ_{-a} be the restriction of τ to $C \setminus C_a$. By the definition of a Nash equilibrium, it suffices to show that in the induced MAID $\mathcal{M}[\sigma'_{-C}]$, no agent a has an incentive to deviate from τ_a to another strategy τ'_a, assuming the other agents adhere to τ. In other words, we must show that τ_a is a Nash equilibrium in the induced single-agent influence diagram $\mathcal{M}[\sigma'_{-C}, \tau_{-a}]$.

Consider an arbitrary decision $D \in C_a$. We are given that τ is an equilibrium in $\mathcal{M}[\sigma_{-C}]$, so the decision rule $\tau_a(D)$ is optimal for (σ_{-C}, τ). We are also given that σ' differs from σ only at nodes that are irrelevant for D. So by Lemma 3, $\tau_a(D)$ is also optimal for (σ'_{-C}, τ). This is equivalent to saying that $\tau_a(D)$ is optimal for the strategy τ_a in $\mathcal{M}[\sigma'_{-C}, \tau_{-a}]$. Since every $\tau_a(D)$ is optimal for τ_a, Lemma 5 implies that τ_a is a Nash equilibrium. ■

Recall that our algorithm produces a strategy profile σ^m. Lemma 6 implies that for each SCC C in the relevance graph, the partial strategy profile that σ^m specifies for C is a Nash equilibrium in $\mathcal{M}[\sigma^m_{-C}]$. Thus, no agent has an incentive to deviate from σ^m on the nodes within any single SCC. We now prove a lemma that generalizes Lemma 5, showing that in fact, no agent has an incentive to deviate from σ^m on any group of nodes.

Lemma 7 *Let $\mathcal{M}$ be a MAID where every agent has perfect recall, and let $C_1, \ldots, C_m$ be sets of decision nodes in $\mathcal{M}$ such that whenever $i < j$, no element of C_j relies on any element of C_i. If σ is a strategy profile for $\mathcal{M}$ such that for each $i \in [1, m]$, σ_{C_i} is a Nash equilibrium in $\mathcal{M}[\sigma_{-C_i}]$, then σ is a Nash equilibrium for $\mathcal{M}$.*

Proof: We must show that for any agent a and any alternative strategy σ'_a for $\mathcal{D}_a$, $\mathrm{EU}_a(\sigma) \geq \mathrm{EU}_a(\sigma_{-a}, \sigma'_a)$. We proceed by induction on the number of sets C_i where σ_a and σ'_a differ. The base case is where they differ on zero sets; then $\sigma_a = \sigma'_a$ and $\mathrm{EU}_a(\sigma) = \mathrm{EU}_a(\sigma, \sigma'_a)$. As an inductive hypothesis, suppose that whenever σ'_a differs from σ_a on k or fewer sets, $\mathrm{EU}_a(\sigma) \geq \mathrm{EU}_a(\sigma_{-a}, \sigma'_a)$.

Now suppose σ'_a differs from σ_a on $k+1$ sets. For each set C_i, let $C_{i,a} = C_i \cap \mathcal{D}_a$. Let C_j be the last set in the ordering where the partial strategy profiles $\sigma_a(C_{j,a})$ and $\sigma'_a(C_{j,a})$ are different. Let $\tau_a = \sigma_a(C_{j,a})$, and $\tau'_a = \sigma'_a(C_{j,a})$. Since $\sigma(C_j)$ is an equilibrium in $\mathcal{M}[\sigma_{-C_j}]$, we know that τ_a is an equilibrium in $\mathcal{M}[\sigma_{-a}, (\sigma_a)_{-C_j}]$. But σ'_a differs from σ_a only on sets C_i where $i < j$, and no node in C_j relies on any node in these sets. So by Lemma 6, τ_a is also an equilibrium in $\mathcal{M}[\sigma_{-a}, (\sigma'_a)_{-C_j}]$. In particular, τ_a yields at least as much expected utility as τ'_a in $\mathcal{M}[\sigma_{-a}, (\sigma'_a)_{-C_j}]$. So:

$$\mathrm{EU}_{\mathcal{M}}\left(\sigma_{-a}, (\sigma'_a)_{-C_j}, \tau_a\right) \geq \mathrm{EU}_{\mathcal{M}}(\sigma_{-a}, \sigma'_a)$$

But $((\sigma'_a)_{-SCC_j}, \tau)$ differs from σ_a on only k sets. So by the inductive hypothesis:

$$\mathrm{EU}_{\mathcal{M}}(\sigma) \geq \mathrm{EU}_{\mathcal{M}}\left(\sigma_{-a}, (\sigma'_a)_{-SCC_j}, \tau_a\right)$$

So by transitivity:

$$\mathrm{EU}_{\mathcal{M}}(\sigma) \geq \mathrm{EU}_{\mathcal{M}}(\sigma_{-a}, \sigma'_a).$$

■

Using this lemma, we can finally prove the correctness of our algorithm.

Theorem 3 *If $\mathcal{M}$ is a MAID where every agent has perfect recall, then the strategy profile σ^m derived by the algorithm above is a Nash equilibrium for $\mathcal{M}$.*

Proof: Consider any SCC $C_{(m-i)}$, where $0 \leq i < m$. By construction, σ^{i+1} assigns to $C_{(m-i)}$ a partial strategy profile τ that is a Nash equilibrium in $\mathcal{M}\left[\sigma^i_{-C_{(m-i)}}\right]$. Because $(m-i)$ decreases as i increases from one iteration of the algorithm to the next, σ^m differs from σ^i only on SCC's C_j where $j < (m-i)$. Because the ordering of SCC's is a topological ordering in the relevance graph, no node in $C_{(m-i)}$ relies on any node in C_j where $j < (m-i)$. Therefore σ^m agrees with σ^i on all nodes that are relevant for any node in $C_{(m-i)}$. So by Lemma 6, τ is a Nash equilibrium in $\mathcal{M}\left[\sigma^m_{-C_{(m-i)}}\right]$. Since this is true for every SCC, Lemma 7 implies that σ^m is a Nash equilibrium in $\mathcal{M}$. ■

Experimental Results

To demonstrate the potential savings resulting from our algorithm, we tried it on the Road example. As shown in the relevance graph in Figure 3(b), the decision for a given plot relies on the decisions about the plot across from it and the plot directly to the south. However, it does not rely on the decision about the land directly north of it, because this decision is observed. None of the other decisions affect this agent's utility directly. The SCC's in the relevance graph all have size 2: they correspond to pairs of decisions about plots that are across from each other.

Even for small values of n, it is infeasible to solve the Road example with standard game-solving algorithms.

Suppose the chance and decision variables each have three possible values, corresponding to three types of buildings. Then the game tree corresponding to the Road MAID has 3^{2n} terminal nodes. Since each agent (except the first two) can observe three ternary variables, he has 27 information sets. So the number of possible pure (deterministic) strategies for each agent is 3^{27}, and the number of pure strategy profiles for all n players is $(3^{27})^{(n-2)} \cdot (3^3)^2$. In the simplest interesting case, where $n = 4$, we obtain a game tree with 6561 terminal nodes, and standard solution algorithms would need to operate on a strategic-form game matrix with about 4.7×10^{27} entries (one for each pure strategy profile).

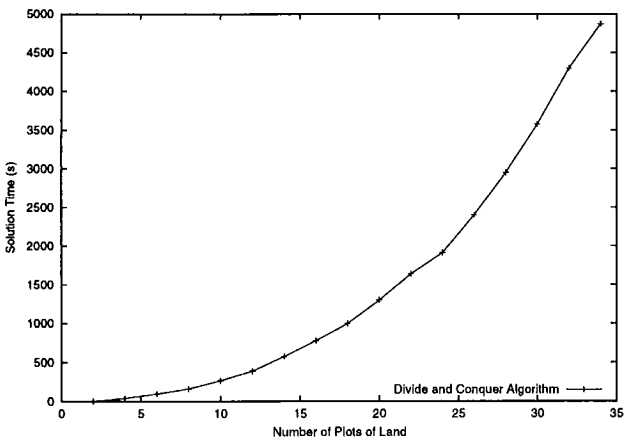

Figure 4: Performance results for the Road example.

Figure 4 shows how our divide-and-conquer algorithm performs on the Road example. We converted each of the induced MAIDs constructed during the algorithm into a small game tree, and used the game solver GAMBIT (McKelvey, McLennan, & Turocy 2000) to solve it. As expected, the time required by our algorithm grows polynomially with n. Thus, we can solve a Road MAID with 34 agents (corresponding to a game tree with 3^{68} terminal nodes) in about 1 hour and 21 minutes.

Discussion and Future Work

We have introduced a new formalism, multi-agent influence diagrams (MAIDs), for modeling multi-agent scenarios with imperfect information. MAIDs allow the conditional independence structure of a scenario to be represented explicitly, limiting the state space explosion which plagues both strategic-form and extensive-form games. We have also shown that MAIDs allow us to define a qualitative graph-based notion that represents strategic relevance, and to exploit it as the basis for algorithms that find equilibria efficiently.

Although the possibility of extending influence diagrams to multi-agent scenarios was recognized at least fifteen years ago (Shachter 1986), the idea seems to have been dormant for some time. Suryadi and Gmytrasiewicz (1999) have used influence diagrams as a framework for learning in multi-agent systems. The focus of their work is very different, and they do not consider the computational benefits derived from the influence diagram representation. Milch and Koller (2000) use multi-agent influence diagrams as a representational framework for reasoning about agents' beliefs and decisions. However, they do not deal explicitly with the conversion of a MAID into a game tree, and leave open the question of how to find equilibria. Nilsson and Lauritzen (2000) have done related work on limited memory influence diagrams, or LIMIDs. Although they mention that LIMIDs could be applied to multi-agent scenarios, they only consider the use of LIMIDs to speed up inference in single-agent settings.

MAIDs are also related to La Mura's (2000) game networks, which incorporate both probabilistic and utility independence. La Mura defines a notion of strategic independence, and also uses it to break up the game into separate components. However, his notion of strategic independence is an undirected one, and thus does not allow as fine-grained a decomposition as the directed relevance graph used in this paper, nor the use of a backward induction process for decisions that are not strategically independent.

This work leaves many interesting open questions. First, there is a great deal of work to be done in relating MAIDs to existing concepts in game theory, particularly equilibrium refinements. On the algorithmic front, the most pressing question is whether we can take advantage of independence structure within SCCs in the relevance graph. On the representational front, it is important to extend MAIDs to deal with asymmetric situations, where the decisions to be made and the information available depend on previous decisions or chance moves. Game trees represent such asymmetry in a natural way, whereas in MAIDs (as in influence diagrams and BNs), a naive representation of an asymmetric situation typically leads to unnecessary blowup. We may be able to avoid these difficulties in MAIDs by explicitly representing context-specificity, as in (Boutilier *et al.* 1996; Smith, Holtzman, & Matheson 1993), integrating the best of the game tree and MAID representations.

References

Boutilier, C.; Friedman, N.; Goldszmidt, M.; and Koller, D. 1996. Context-specific independence in Bayesian networks. In *Proc. 12th UAI*, 115–123.

Cormen, T. H.; Leiserson, C. E.; and Rivest, R. L. 1990. *Introduction to Algorithms*. Cambridge, MA: MIT Press.

Fudenberg, D., and Tirole, J. 1991. *Game Theory*. Cambridge, MA: MIT Press.

Geiger, D.; Verma, T.; and Pearl, J. 1990. Identifying independence in Bayesian networks. *Networks* 20:507–534.

Howard, R. A., and Matheson, J. E. 1984. Influence diagrams. In *Readings on the Principles and Applications of Decision Analysis*. Strategic Decisions Group. 721–762.

Jensen, F.; Jensen, F. V.; and Dittmer, S. L. 1994. From influence diagrams to junction trees. In *Proc. 10th UAI*, 367–373.

Keeney, R. L., and Raiffa, H. 1976. *Decisions with Mul-*

tiple Objectives: Preferences and Value Tradeoffs. John Wiley & Sons, Inc.

Koller, D.; Megiddo, N.; and von Stengel, B. 1994. Fast algorithms for finding randomized strategies in game trees. In *Proc. 26th STOC*, 750–759.

La Mura, P. 2000. Game networks. In *Proc. 16th UAI*, 335–342.

McKelvey, R. D., and McLennan, A. 1996. Computation of equilibria in finite games. In *Handbook of Computational Economics*, volume 1. Amsterdam: Elsevier Science. 87–142.

McKelvey, R. D.; McLennan, A.; and Turocy, T. 2000. GAMBIT software, California Institute of Technology. http://www.hss.caltech.edu/gambit/Gambit.html.

Milch, B., and Koller, D. 2000. Probabilistic models for agents' beliefs and decisions. In *Proc. 16th UAI*, 389–396.

Nash, J. 1950. Equilibrium points in n-person games. *Proc. National Academy of Sciences* 36:48–49.

Nilsson, D., and Lauritzen, S. L. 2000. Evaluating influence diagrams with LIMIDs. In *Proc. 16th UAI*, 436–445.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann.

Romanovskii, I. V. 1962. Reduction of a game with complete memory to a matrix game. *Soviet Mathematics* 3:678–681.

Shachter, R. D. 1986. Evaluating influence diagrams. *Operations Research* 34:871–882.

Shachter, R. D. 1998. Bayes-ball: The rational pastime. In *Proc. 14th UAI*, 480–487.

Smith, J. E.; Holtzman, S.; and Matheson, J. E. 1993. Structuring conditional relationships in influence diagrams. *Operations Research* 41(2):280–297.

Suryadi, D., and Gmytrasiewicz, P. J. 1999. Learning models of other agents using influence diagrams. In Kay, J., ed., *Proc. 7th Int'l Conf. on User Modeling*, 223–232.

Zermelo, E. 1913. Über eine Anwendung der Mengenlehre auf der Theorie des Schachspiels. In *Proceedings of the Fifth International Congress on Mathematics*.