# Modeling a multi-agent environment combining influence diagrams

Sam Maes, Karl Tuyls, and Bernard Manderick
Computational Modeling Lab (COMO), Department of Computer Science, Vrije Universiteit Brussel , Belgium
E-mail: {sammaes@,ktuyls@,bernard@arti.}vub.ac.be

**Abstract**

*In this paper we show how influence diagrams (IDs) can be combined for modeling a multi-agent environment. Influence diagrams are a compact representation of a joint probability distribution where actions and utilities are explicitly modeled.*

*Combining different IDs which represent individual agents into a global coherent network representing the entire multi-agent system is a difficult problem. For this purpose we introduce two approaches and compare them.*

## 1 Introduction

In this paper we address the problem of modeling the environment and other agents in multi-agent systems (MAS). Existing formalisms such as the Markov game model [1, 2] suffer from combinatorial explosion, since they learn values for combinations of actions of all the agents. We suggest to use influence diagrams [3] (also known as decision networks) to avoid this problem of tractability.

We will discuss the problem of modeling the environment and other agents acting in the environment in the context of Markov Games [1, 2]. The Markov game model is defined by a set of states S, and a collection of action sets $A_1, ..., A_n$ (one set for every agent). The state transition function $S \times A_1 \times ... \times A_n \rightarrow P(S)$ maps a state and an action from every agent to a probability on S. Each agent has an associated reward function $R_i$ :

$$S \times A_1 \times ... \times A_n \rightarrow \Re \qquad (1)$$

where $S \times A_1 \times ... \times A_n$ constitutes a product space. The reward function for an agent $A_i$ calculates a value which indicates how desired the state $S$ and the actions of the agents $A_1, \ldots, A_n$ are for agent $A_i$.

This model can become intractable, because in Markov games each agent learns values for combinations of actions $(A_1, \ldots, A_n)$ of all the agents in the environment, as opposed to only learning values for its own actions. This means that learning is done in a product space and when the number of agents increases this becomes prohibitively large.

Nowe and Verbeeck [4] avoid this problem by forcing an agent to model only the agents that are relevant to him. A similar approach is adopted in [5].

In this paper we will investigate the possibilities of a technique, more precisely influence diagrams, for modeling the reward function in a more efficient way to solve the combinatorial explosion mentioned above.

## 2 Modeling the multi-agent environment

In this section we introduce our solution to the problem of combinatorial explosion of the product space.

### 2.1 Influence diagrams modeling an agent

It is our idea to use influence diagrams (IDs) for modeling the other agents acting in the environment. An influence diagram [3] is a graphical knowledge representation of a decision problem. It may be viewed as an extension to a Bayesian network [6][3], with additional node types for decisions and utilities. IDs have three types of nodes: random nodes, decision nodes and utility nodes. Just as in Bayesian networks, random nodes are associated with random variables, which represent the agent's possibly uncertain beliefs about the world. A decision node holds the choice of actions an agent has, thus represents the agent's capabilities. A utility node represents the agent's preferences. The links between the nodes summarize their dependence relationships.

We propose to use such influence diagrams for each agent to model the other agents in the domain. The resulting model will describe how the different agents influence each other and how they influence the reward the agent receives. An influence diagram is a factorization of the joint probability distribution in the following way:

$$P(var_1, \ldots, var_n) = \prod_{i=1 \ldots n} P(var_i | parents(var_i))$$

where $parents(var_i)$ denotes the nodes in the graph that are connected with $var_i$ through an edge ending in $var_i$. So when ID's are used to model a multi-agent environment, the independencies in the domain can be exploited to make the product space (see equation 1) more tractable.

Now we shall clarify these ideas with an example. Consider three agents : $A_1$, $A_2$ and $A_3$. They act in the same environment, more precisely a market situation where they can buy five kinds of goods. The goods are : rice, milk, potatoes, coffee and tomatoes. The price of each good is governed by the laws of supply and demand, and also by an environmental variable $inflation$, which determines the value of money in the environment. Each agent is interested in buying particular goods. Agent $A_1$ is only interested in rice, milk and potatoes, agent $A_2$ is interested in coffee and tomatoes and agent $A_3$ is interested in milk and coffee. Each agent will be represented by a combination of influence diagrams as a model of equation 1. Each influence diagram represents one specific agent.

In this example the state S of the environment consists of the states of the three agents, the prices of the goods and the inflation. We make the following distinction concerning the factors that constitute the possible states (S in the product space) of the environment :

- *Local* factors, those factors relevant to a limited subset of agents. As an example these are an agent's funds and his stock of goods. They will be included in the ID representing the agent.

- *Global* factors, those factors that are of concern for all agents. In the example these are inflation and the prices of the goods. They will be included in a separate network.

To understand how an agent's state can be represented by his local factors using an influence diagram, we will examine agent $A_3$ in more detail. $A_3$ is interested in milk and coffee, thus in the influence diagram there will be random variables $coffee$ and $milk$, representing the amounts of coffee and milk the agent has in stock. Since these goods have to be bought, there will also be a random variable $funds$, which represents the amount of money the agent possesses. The decision node for $A_3$ in this simple example will consist of $\{buy\_milk, buy\_coffee, other\}$[1] and the utility node will consist of a function which returns a value depending on the amount of money, milk and coffee $A_3$ has.

In Figure 1 we depicted an influence diagram representation for agent $A_3$ of the example described above.

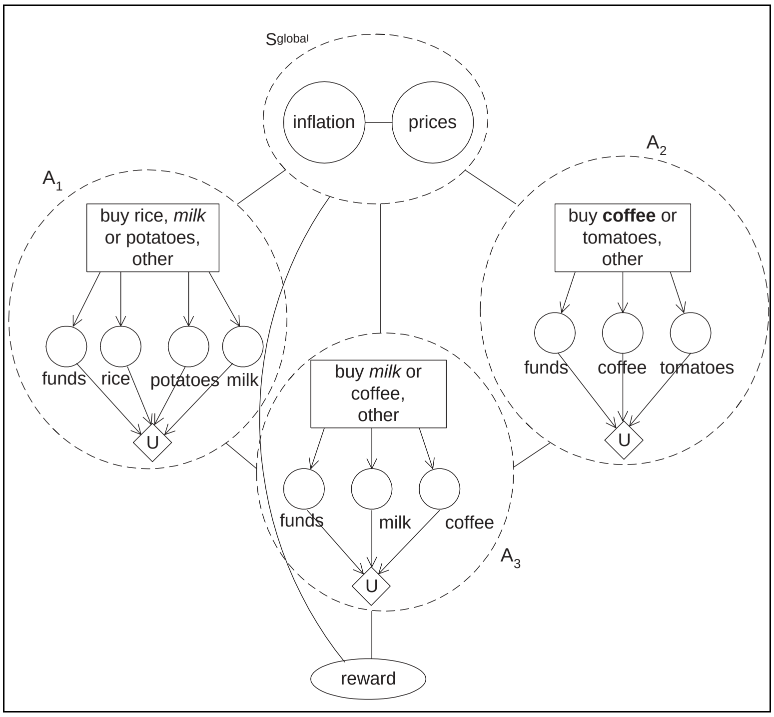


Figure 1: A detailed influence diagram model for agent $A_3$. For the sake of clarity the prices of the goods are depicted as one single variable, instead of one variable for each good. IDs have three types of nodes: random nodes (circle), decision nodes (rectangle) and utility nodes (diamonds).

As we can see in the picture, agent $A_3$'s model of the reward function becomes a network of different influence diagrams (one for each agent in the environment), some global environmental variables and a reward variable.

The links between agents $A_1$ and $A_3$ and between agents $A_2$ and $A_3$ state that they have a direct influence on each other. This is because they have a common interest.

[1]Since a decision node must be exhaustive, the *other* action specifies all unknown or unlikely actions of agent $A_3$ that are not explicitly represented.

Remark that there is no link between $A_1$ and $A_2$, because these agents do not share interests. Further there are links from $S_{global}$ to each agent, meaning that each agent has a direct notion of the state of the variables specified in the global environment. Because the actual reward is also directly influenced by the state of the environment, there is a direct connection between the environment node $S_{global}$ and the reward node.

## 2.2 Combining IDs to model the multi-agent environment

Above we mentioned that a link between two agents signifies direct influence upon each other. In our approach individual agents will be represented by IDs. A major difficulty is to develop a technique that propagates the changes in an ID to all his neighboring IDs, which are the representations of the agents that are directly influenced by him.

In this subsection we introduce two approaches to combine the IDs representing the different agents into a coherent network. This means that influences should be propagated in a way that reflects the real effects that the agents have on each other.

The first approach is to combine the individual influence diagrams into a global coherent network. This means that we have to find which parts of the IDs have a direct influence on each other and should be connected[2]. The disadvantage of this approach is that it can be difficult to know precisely which variables influence each other and should thus be connected with an edge.

The second approach is based on junction trees[7][8], this is a simplified representation of the original network, often used to allow inference in multiply connected networks. A junction tree is a Bayesian network representing the same joint probability distribution as the original network, but where nodes are clustered into larger nodes to remove cycles from the network [3]. In figure 2 we see the ID representing agent $A_1$ and his transformation to a junction tree.

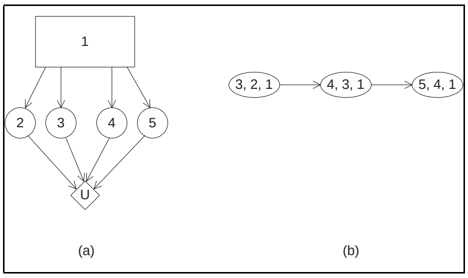


Figure 2: (a) A typical influence diagram. (b) The junction tree of the ID on the left.

The second approach consists of transforming each individual network into a junction tree, and then combining these simplified trees into a global network where inference can be done in an efficient way. It is our opinion that first transforming the IDs into junction

[2]For example when combining the IDs of $A_1$ and $A_3$ of figure 1, the decision node of $A_1$ will be connected with the milk node of $A_3$ and vice versa.

[3]Such a representation of the network is constructed by applying moralization and triangulation to the graph. The nodes of the junction tree correspond to cliques – maximal complete subgraphs – of the triangulated graph [7].

trees will facilitate the connection process. Intuitively this can be seen in figure 2, where the junction tree of the influence diagram representing agent $A_1$ in Figure 1 is depicted. It is obvious that the junction tree is much simpler than the original influence diagram and it might be easier to connect them. A possible way to do this is by connecting all leaf nodes of one network with all root nodes of the other network.

It is our intention to experiment with these two approaches and make a general comparison concerning the quality, complexity, efficiency and scalability of the results.

## 3 Conclusion

In this paper we proposed to use influence diagrams for modeling other agents to solve the combinatorial explosion associated with the product space. The advantages of IDs for modeling agents are:

1. Concise representation of the relations between the agents to avoid combinatorial explosion.
2. In contrast to the product space approach, information concerning the actions of only a subset of the other agents is enough to calculate a reward.
3. Actions and utilities of other agents are explicitly modeled.

Our first concern is to see whether a multi-agent system using influence diagrams for modeling other agents in the environment yields good results. It is our conviction that IDs can contribute a great deal to multi-agent systems, because IDs are a concise representation of the product space. If this is the case, the next step is to investigate whether these representations can be learned from observing the environment and the behavior of other agents in an acceptable time.

## References

[1] J. Hu & M.P. Wellman, Multiagent reinforcement learning in stochastic games, (Submitted for publication 1999)

[2] M.L. Litmann, Markov games as a framework for multi-agent reinforcement learning, *Proceedings of the Eleventh International Conference on Machine Learning*, 157-163, 1994.

[3] S. Russell & P. Norvig, *Artificial Intelligence: a Modern Approach*, Prentice Hall Series in Artificial Intelligence. Englewood Cliffs, New Jersey, 1995.

[4] A. Nowe & K. Verbeeck, *Learning Automata and Pareto Optimality for Coordination in MAS*, Technical report, COMO, Vrije Universiteit Brussel, 2000.

[5] A. Defaweux, T. Lenaerts, S. Maes, K. Tuyls, P. van Remortel & K. Verbeeck, Niching and Evolutionary Transitions in MAS. (Submitted to *ECOMAS-GECCO 2001*)

[6] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, CA, 1988.

[7] F.V. Jensen, F. Jensen & S.L. Dittmer, From Influence Diagrams to Junction Trees, *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, R. Lopez de Mantaras & D. Poole (Eds), Morgan Kaufmann, San Francisico, 367-373, 1994.

[8] F.V. Jensen & A.L. Madsen, LAZY propagation: A junction tree inference algorithm based on lazy evaluation, *Artificial Intelligence*, Vol. 113, 203-245, 1999.

[9] D. Heckerman, et. al., Dependency Networks for Inference, Collaborative Filtering, and Data Visualization, *Proc. of the Workshop on Statistical Modelling for Data Mining*, 57-87, 2000.

[10] R.S. Sutton & A.G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.

[11] N.L. Zhang, Probabilistic inference in influence diagrams, *Proc. of the 14th Conference on UAI*, 514-522, 1998.