An influence diagram framework for acting under influence by agents with unknown goals

Nicolaj Søndberg-Jeppesen and Finn Verner Jensen

Aalborg University,

Department of Computer Science

September 17, 2008

2 agents with conflicting plans

- We are dealing with scenarios where 2 agents interact.
- The agents do not know the other agent's goals.
- The goals may be conflicting.
- We will let each agent have an "assignment" which determines its goals. The assignments are hidden for the other player.

2 agents with conflicting plans

- We are dealing with scenarios where 2 agents interact.
- The agents do not know the other agent's goals.
- The goals may be conflicting.
- We will let each agent have an "assignment" which determines its goals. The assignments are hidden for the other player.

2 agents with conflicting plans

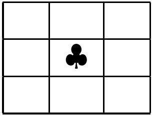
- We are dealing with scenarios where 2 agents interact.
- The agents do not know the other agent's goals.
- The goals may be conflicting.
- We will let each agent have an "assignment" which determines its goals. The assignments are hidden for the other player.

2 agents with conflicting plans

- We are dealing with scenarios where 2 agents interact.
- The agents do not know the other agent's goals.
- The goals may be conflicting.
- We will let each agent have an "assignment" which determines its goals. The assignments are hidden for the other player.

Simple example environment

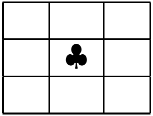
- An example scenario: The Grid Game.
- 2 players move 1 piece on a grid.



- The players prefer to move the piece to some cells more than others.
- How much a player prefers a cell is determined by her Assignment.
- The player's assignment is unknown to the opponent.

Simple example environment

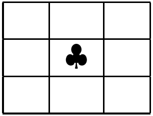
- An example scenario: The Grid Game.
- 2 players move 1 piece on a grid.



- The players prefer to move the piece to some cells more than others.
- How much a player prefers a cell is determined by her Assignment.
- The player's assignment is unknown to the opponent.

Simple example environment

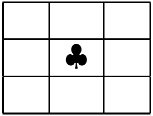
- An example scenario: The Grid Game.
- 2 players move 1 piece on a grid.



- The players prefer to move the piece to some cells more than others.
- How much a player prefers a cell is determined by her Assignment.
- The player's assignment is unknown to the opponent.

Simple example environment

- An example scenario: The Grid Game.
- 2 players move 1 piece on a grid.



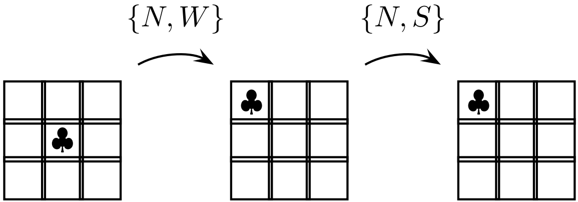
- The players prefer to move the piece to some cells more than others.
- How much a player prefers a cell is determined by her Assignment.
- The player's assignment is unknown to the opponent.

The rules

Moving the piece:

- Possible moves: N, E, S, W.
- The effect on the piece is the combination of the 2 player's moves.
- If one player chooses a move which makes the joint move impossible the piece is only moved in the direction the other player has chosen if that can be carried out.

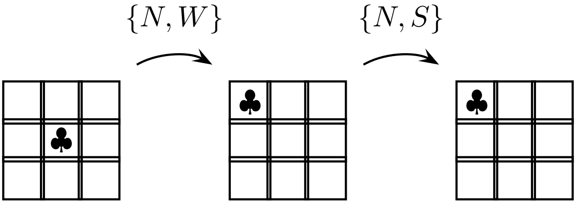
Grid game example



Assignment 1			Assignment 2		
8.55	6.82	1.15	-1.66	0.42	6.84
-6.82	0.0	4.12	-0.51	0.0	-0.42
-8.55	-1.15	-4.12	-6.84	0.51	1.66

Figure: An example of the game Grid. In the first move, P^1 chooses to move N while P^2 chooses to move W. In the second turn, P^1 and P^2 moves N and S respectively, cancelling each other's effect.

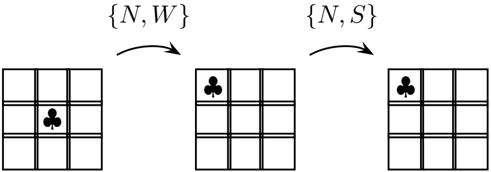
Grid game example



Assignment 1			Assignment 2		
8.55	6.82	1.15	-1.66	0.42	6.84
-6.82	0.0	4.12	-0.51	0.0	-0.42
-8.55	-1.15	-4.12	-6.84	0.51	1.66

Figure: An example of the game Grid. In the first move, P^1 chooses to move N while P^2 chooses to move W. In the second turn, P^1 and P^2 moves N and S respectively, cancelling each other's effect.

The scores



Assignment 1			Assignment 2		
8.55	6.82	1.15	-1.66	0.42	6.84
-6.82	0.0	4.12	-0.51	0.0	-0.42
-8.55	-1.15	-4.12	-6.84	0.51	1.66

$$P^1 : 8.55 - (-1.66) = 10.21$$

$$P^2 : -1.66 - 8.55 = -10.21$$

Opponent Modeling

- To deal successfully with this kind of game requires opponent modeling.
- Equip each agent with a model of its opponent.
- Each model will contain models of the other agent which in turn will contain a model of the first agent.
- This inevitably results in an infinite regress.
- Classical solutions to that is to find Nash Equilibria.

Recursive modeling

- Instead of solving Nash Equilibria we use the recursive modeling method (RMM) (Gmytrasiewicz et al. [1991]).
- In RMM the recursion is ended at a certain level.
- A "flat" model is inserted at the deepest level, i.e. a model that does not contain models of other players.

Covert Interference (CIF)

We propose a framework which we call Covert Interference (CIF).

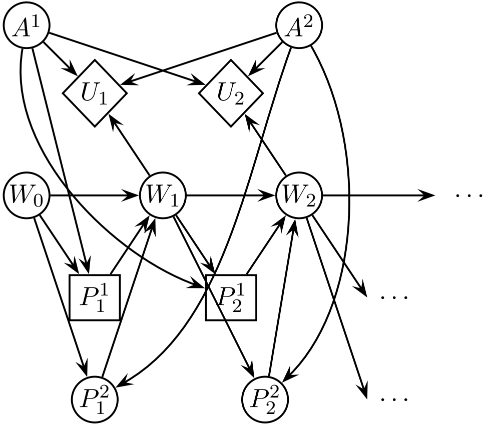


Figure: Covert Interference.

Covert Interference (CIF)

The model explains :

- How the hidden assignments are modeled using the chance nodes A and B.
- The transition between states as a function of the two player's actions
- The utility functions.
- The opponent's strategy - represented by a chance node.

The model **does not** tell us anything about:

- How many future time steps the agent considers.
- How many future time steps the opponent is assumed to consider
- How deep a modeling level the opponent can be assumed to have.

Covert Interference (CIF)

The model explains :

- How the hidden assignments are modeled using the chance nodes A and B.
- The transition between states as a function of the two player's actions
- The utility functions.
- The opponent's strategy - represented by a chance node.

The model **does not** tell us anything about:

- How many future time steps the agent considers.
- How many future time steps the opponent is assumed to consider
- How deep a modeling level the opponent can be assumed to have.

Definition of a (Perfect Recall) Player

In order to describe the missing parts of the model we give the following definition:

Definition (RMM Player)

A player P is a pair defined as follows:

1. $P = (h, NIL)$ is a player with time horizon h and modeling level 0.
2. Given a player O, with modeling level $i - 1$, $P = (h, O)$ is a player with time horizon h and modeling level i.

Examples

Thus,

- a (2,NIL) player is a player that takes into account 2 future time steps and does not have a model of the opponent (assumes random play).
- A (3,(2,NIL)) model takes 3 future time steps into account while it assumes the opponent uses a (2,NIL) model.
- A (3,(3,(2,NIL))) model also takes into account 3 future time steps while she assumes the opponent is a (3,(2,NIL)) model.
- etc.

Examples

Thus,

- a (2,NIL) player is a player that takes into account 2 future time steps and does not have a model of the opponent (assumes random play).
- A (3,(2,NIL)) model takes 3 future time steps into account while it assumes the opponent uses a (2,NIL) model.
- A (3,(3,(2,NIL))) model also takes into account 3 future time steps while she assumes the opponent is a (3,(2,NIL)) model.
- etc.

Examples

Thus,

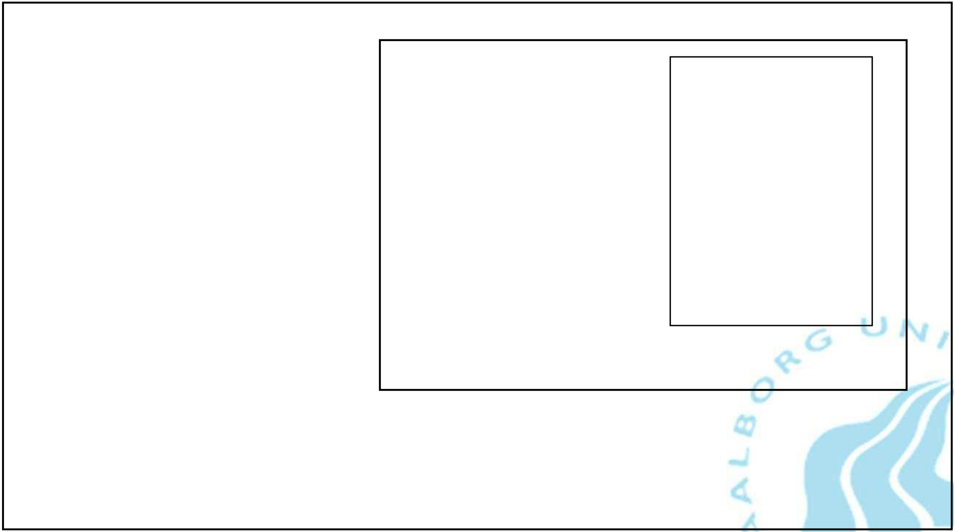
- a (2,NIL) player is a player that takes into account 2 future time steps and does not have a model of the opponent (assumes random play).
- A (3,(2,NIL)) model takes 3 future time steps into account while it assumes the opponent uses a (2,NIL) model.
- A (3,(3,(2,NIL))) model also takes into account 3 future time steps while she assumes the opponent is a (3,(2,NIL)) model.
- etc.

Examples

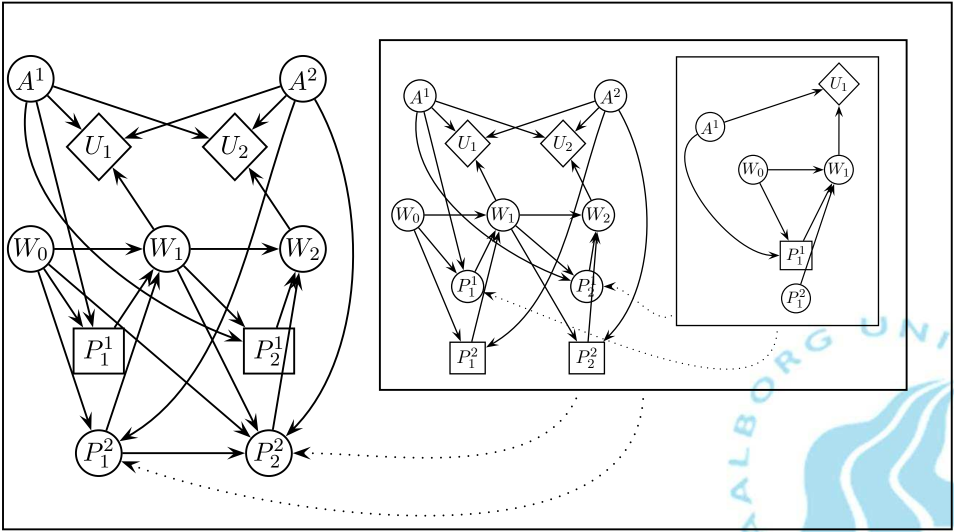
Thus,

- a (2,NIL) player is a player that takes into account 2 future time steps and does not have a model of the opponent (assumes random play).
- A (3,(2,NIL)) model takes 3 future time steps into account while it assumes the opponent uses a (2,NIL) model.
- A (3,(3,(2,NIL))) model also takes into account 3 future time steps while she assumes the opponent is a (3,(2,NIL)) model.
- etc.

Example: a $(2,(2,(1,NIL)))$ model



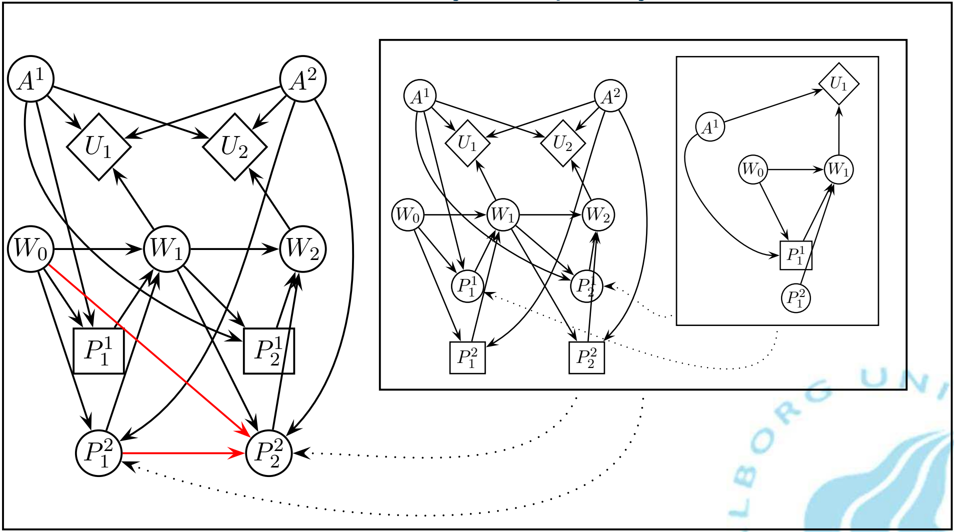
Example: a $(2, (2, (1, NIL)))$ model



Memory Complexity Problems

- Notice the extra arcs in the previous Figure!

Memory Complexity Problems



Memory Complexity Problems

- Notice the extra arcs in the previous Figure!
- We assume No-forgetting.
- P^2's decision in P_1^2 may reveal something about his assignment.
- Thus, all previous board states and all our previous actions are relevant to the next decision.
- The memory complexity becomes forbidding for playing the game.

Memory Complexity Problems

- Notice the extra arcs in the previous Figure!
- We assume No-forgetting.
- P^2's decision in P_1^2 may reveal something about his assignment.
- Thus, all previous board states and all our previous actions are relevant to the next decision.
- The memory complexity becomes forbidding for playing the game.

Memory Complexity Problems

- Notice the extra arcs in the previous Figure!
- We assume No-forgetting.
- P^2's decision in P_1^2 may reveal something about his assignment.
- Thus, all previous board states and all our previous actions are relevant to the next decision.
- The memory complexity becomes forbidding for playing the game.

Memory Complexity Problems

- Notice the extra arcs in the previous Figure!
- We assume No-forgetting.
- P^2's decision in P^2_1 may reveal something about his assignment.
- Thus, all previous board states and all our previous actions are relevant to the next decision.
- The memory complexity becomes forbidding for playing the game.

Memory Complexity Problems

- Notice the extra arcs in the previous Figure!
- We assume No-forgetting.
- P^2's decision in P^2_1 may reveal something about his assignment.
- Thus, all previous board states and all our previous actions are relevant to the next decision.
- The memory complexity becomes forbidding for playing the game.

Limited memory influence diagrams

- Lauritzen and Nilsson [2000] have proposed Limited memory influence diagrams (LIMIDS).
- They give up the no-forgetting assumption.
- The syntax is like IDs but the only thing known at decisions are represented by information-arcs into that decision node.
- Lauritzen and Nilsson [2000] propose a solution algorithm for LIMIDS, namely Single Policy Update (SPU).

A LIMID Player

- With a lot of inspiration from LIMIDs we introduce a Limited Memory Player (LIMID Player).
- A LIMID player has a certain look-ahead and modeling level, just like perfect recall players, but
  as opposed to perfect recall players they have a limited memory.
- With a memory of m the LIMID player remembers the last m decisions and the last m world
  states.

Definition of a LIMID Player

Definition (RMM LIMID Player)

A RMM LIMID player L is a triple defined as follows:

1. $L = (h, m, NIL)$ is a LIMID player with time horizon h, memory m and modeling level 0.
2. Given a player or a LIMID player O, with modeling level $i - 1$, $P = (h, m, O)$ is a LIMID player with time horizon h, memory m, and modeling level i.

An example of a LIMID player

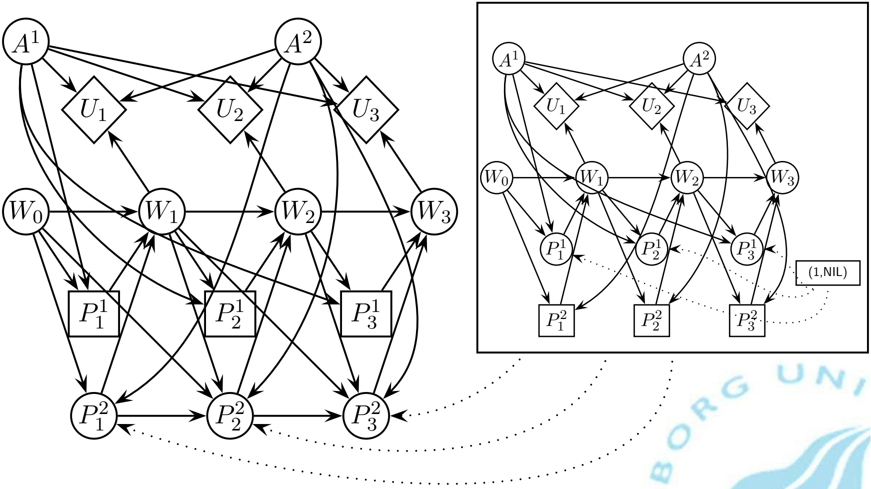


Figure: An example of a (3,2,(3,1,(1,NIL))) model.

An example of a LIMID player

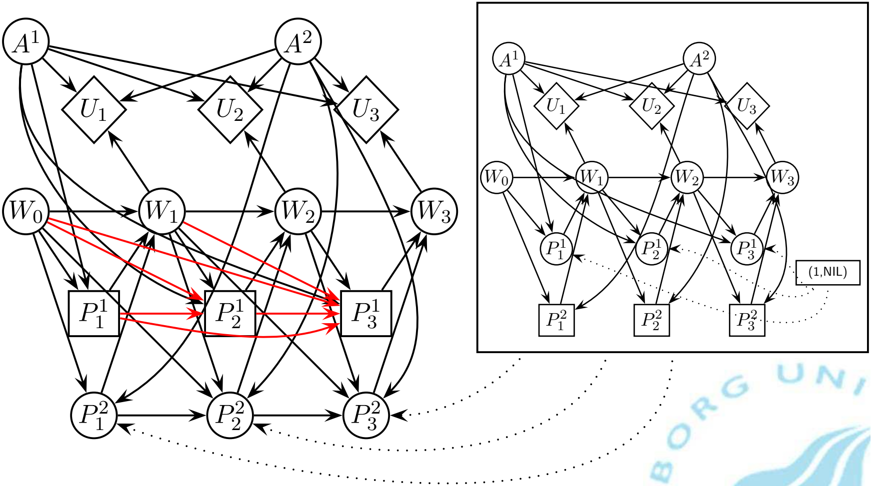


Figure: An example of a (3,2,(3,1,(1,NIL))) model.

An example of a LIMID player

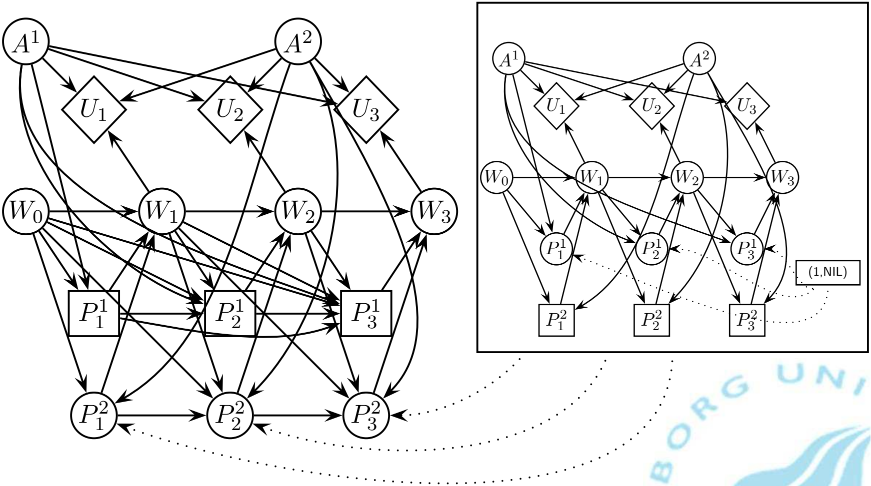


Figure: An example of a (3,2,(3,1,(1,NIL))) model.

An example of a LIMID player

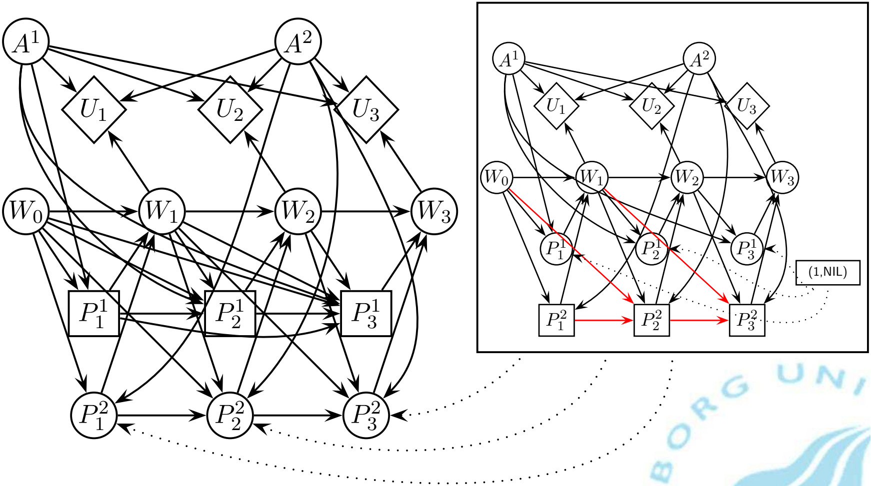


Figure: An example of a (3,2,(3,1,(1,NIL))) model.

An example of a LIMID player

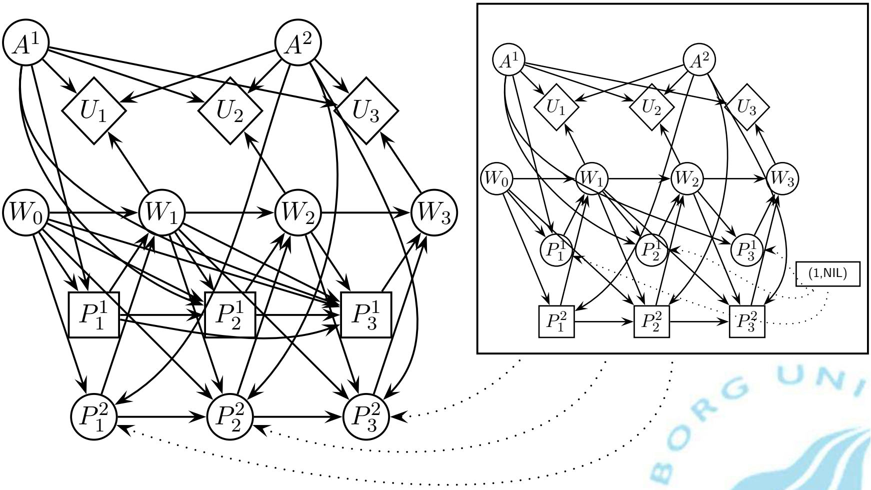


Figure: An example of a (3,2,(3,1,(1,NIL))) model.

Single policy updating

- Convert all decision nodes into chance nodes starting with uniform priors.
- Repeat until convergence:
  - Starting from the last decision, find the optimal policy for that decision given the parents and insert that in the chance node.
  - Proceed with the second last decision now knowing the policy for the last decision.
  - Continue finding local optimal policies down to the first decision.

Single policy updating

- Convert all decision nodes into chance nodes starting with uniform priors.
- Repeat until convergence:
  - Starting from the last decision, find the optimal policy for that decision given the parents and insert that in the chance node.
  - Proceed with the second last decision now knowing the policy for the last decision.
  - Continue finding local optimal policies down to the first decision.

Single policy updating

- Convert all decision nodes into chance nodes starting with uniform priors.
- Repeat until convergence:
  - Starting from the last decision, find the optimal policy for that decision given the parents and insert that in the chance node.
  - Proceed with the second last decision now knowing the policy for the last decision.
  - Continue finding local optimal policies down to the first decision.

Single policy updating

- Convert all decision nodes into chance nodes starting with uniform priors.
- Repeat until convergence:
  - Starting from the last decision, find the optimal policy for that decision given the parents and insert that in the chance node.
  - Proceed with the second last decision now knowing the policy for the last decision.
  - Continue finding local optimal policies down to the first decision.

Single policy updating

- Convert all decision nodes into chance nodes starting with uniform priors.
- Repeat until convergence:
  - Starting from the last decision, find the optimal policy for that decision given the parents and insert that in the chance node.
  - Proceed with the second last decision now knowing the policy for the last decision.
  - Continue finding local optimal policies down to the first decision.

Experiments

In experiments we have investigated:

1. The allowed time horizon for players with perfect recall compared to LIMID players.
2. The performance of the two models against a benchmark.
3. The importance of having the correct model of the opponent.

1. The allowed time horizon

Table: The maximal time horizons possible on our system with different sizes of the Grid game.

Board	ID max h	LIMID max h $(m = 1)$
3×3	4	32
5×5	3	8
7×7	3	8
9×9	2	8

2. The performance of the two models

Table: Average scores and standard deviations (σ) after 100 Grid games between different models against (2,(2,(1,NIL))).

	Model	P^1	σ
1	(2,(2,(2,(1,NIL))))	5.03	*10.1*
2	(2,1,(2,(2,(1,NIL))))	-0.248	*8.66*
3	(3,(2,(2,(1,NIL))))	7.32	*10,7*
4	(3,2,(2,(2,(1,NIL))))	0.252	*8,27*

3. The importance of having the correct model of the opponent

Table: Average scores and standard deviations (*italics*) obtained by players with $h = 3$ on different levels in a 3×3 instance of Grid.

Level	0	1	2	3	4
1	2.47	–	–	–	–
	5.41	–	–	–	–
2	-1.83	3.24	–	–	–
	6.39	*10.76*	–	–	–
3	-3.19	-4.50	9.29	–	–
	7.64	*10.9*	*10.5*	–	–
4	0.55	-4.60	-0.73	8.00	–
	7.06	*10.0*	*5.82*	*10.6*	–
5	0.572	1.18	-6.21	4.40	5.78
	6.36	*7.34*	*10.8*	*10.0*	*8.84*

Conclusions & Future Research

- We have proposed a framework called CIF for solving agent encounters when the goals of the opponents are uncertain.
- We have addressed the complexity problem caused by the amount of relevant information.
- The empirical results for the LIMID player has shown a loss in performance compared to perfect recall players.
- The modeling level of the opponent has turned out to be important in order to successfully win the game. (Adaptation.)
- Current research: Investigate alternative opportunities for model approximation.

Thank You!

References

P. J. Gmytrasiewicz, E. H. Durfee, and D. K. Wehe. A decision theoritic approach to coordinating multiagent interactions. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI 91)*, pages 62–68, 1991.

S. Lauritzen and D. Nilsson. Evaluating influence diagrams using LIMIDs. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 436–445, 2000.