On using Multi-agent Systems in Playing Board Games

Stefan J. Johansson
Department of Software Systems
School of Engineering
Blekinge Institute of Technology
sja@bth.se

ABSTRACT

Computer programs able to play different kinds of games (aka bots) is a growing area of interest for the computer game industry as the demand for better skilled computerized opponents increase. We propose a general architecture of a Multi-agent System (MAS) based bot able to play complex board games and show that this solution is able to outperform other bots in two quite different games, namely no-press Diplomacy and Risk. Based on these results, we formulate a hypothesis of the applicability of MAS based bots in the domain of board games and identify the need for future investigations in the area.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: [Multiagent systems, Coherence and coordination]

General Terms

Algorithms

Keywords

Multiagent system architecture, Board games, RISK, Diplomacy

1. INTRODUCTION

Creating programs able to play classical strategic games such as Othello, Chess, Risk, Diplomacy and Go has since the early days of AI been a challenge [6]. Some of these, Chess, and especially Othello, Checkers, Backgammon are now considered *solved* in the sense that the programs are able to beat the top human players [1, 10, 14, 19], while games such as Go, Risk and Diplomacy still remain among the ones that we have a hard time to find an optimal computerized player for. The main problem of the latter games is the branching of possible moves. To take one example, the branching factor of chess is 20 for the first move ($20^2 = 400$ for the first round of moves), whereas in Diplomacy, there are 4430690040914420 unique openings [11]. This makes it intractable to use traditional minimax-based algorithms to search for good moves in the games.

Previous attempts to make agent based solutions to games include:

- Chess - Drogoul [3] and Fransson [4] have tried to create MASs playing Chess based on various kinds of negotiations between the Chess pieces, rather than traditional minimax search algorithms, but the performance of these systems is not overwhelming.
- Stratego - Treijtel made a MAS based Stratego player where the system implemented a distributed rule based decision system [22].
- Diplomacy - Kraus and Lehmann [9] and lately also Shaheed [15] have made heterogeneous MASs of agents able to negotiate with other players.
- No-press Diplomacy - Johansson and Håård show promising results in this version of Diplomacy with a bot based on a homogeneous MAS [7].
- Risk - Johansson and Olsson made a MAS playing Risk which showed to be successful [8].

The latter two will be described more in detail in Section 5. In the next section we will describe a general model of games, followed by a general MAS architecture in Section 3. We will then in Section 6 discuss how to evaluate MAS based bots before we finish with a discussion and draw some conclusions.

2. GENERAL MODEL OF THE GAME

We have chosen to limit this investigation to games which have the following properties:

1. It has a set of starting players S, where $|S|$ is either fixed, or within a certain interval.
2. The game has a set of territories T with an adjacency set $A \subseteq T \times T$.
3. The game has a set P of different types of pieces.
4. Each type of piece $p \in P$, may move according to $A_p \subseteq A$
5. The game is divided into a number of synchronized turns $T = \{t_0, t_1, ...\}$.

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.

6. Each turn may be carried out in the sequence of the players, or in parallel.

7. The turns are divided into (at least) three phases: initial phase, action phase(s), and finish phase.

8. In the initial phase, I, new resources (if available) are entered into the system and preparations are made for the actions of the coming phases.

9. In an action phase, A, the player decides what actions to take and these are then performed according to the rules of the game.

10. In the finish phase, F, the player make preparations for the next phase.

In addition to this, each game may have a number of other properties, e.g.

- Typically, the state of the game restricts A_p even more.
- The state of the game may be partially unknown to the players.
- The territories may be occupied by none, one or several pieces at the same time.
- The adjacency matrix may be symmetric, or arbitrary.
- The game may have non-deterministic outcomes of the actions of the players.
- Each turn may consist of a single move, or a (dynamically set) sequence of moves.
- The game can be exogenous, zero-sum, or endogenous (in the sense that resources are enriched by the game, staying equal throughout the game, or being consumed in the game).

2.1 An example of a game

As an exemplification of our general model, let us take a closer look at the well-known game of chess. It is a game with six types of pieces (King, Queen, Bishop, Knight, Tower and Pawn). Each type has its own adjecency matrix[1], but the moves that are possible to make are further restricted by the state of the game (e.g. a pawn may only move straight forward unless the space in front of it is not occupied). Each turn, which are carried out in sequence, consists of an empty initial phase, a one-move action phase, and an empty finish phase (unless the players are using a clock). The outcome of an action is deterministic and the state of the game is known to its two players. The game is of the endogenous type and the territories may at most hold at most one piece at a time.

3. GENERAL MODEL OF THE BOT

The general idea is to have a multi-agent system where each important unit of the player on the board corresponds to an agent in the system. These agents may either correspond to the pieces or the territories of the player(s). The agents will then negotiate about what actions to take in the system, possibly helped by a mediator agent taking care of the actual negotiation (e.g. carry out an auction) through a negotiator mechanism. The architecture is depicted in Figure 1.

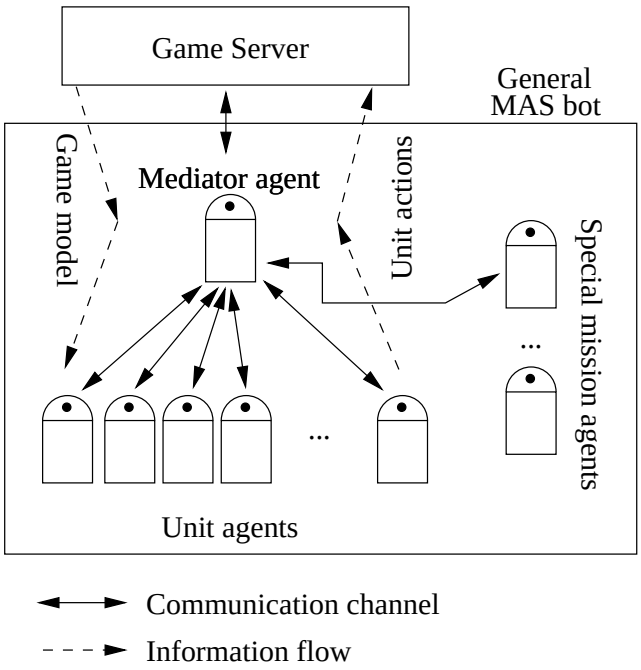

Figure 1: The agent system of the general model.

Typically, the mediator agent is also propagating the information from the agent system (i.e. what actions to perform) to the game server, as well as spreading the information from the server (e.g. the moves of the opponents) to the agent system (see the information router in Figure 2). This includes translating the information from its internal representation, to the external format used by the present game server and vice versa. This task is dedicated to the game server interface.

There may also be other agents in the system with dedicated missions, such as keeping track of the time (if it is a time critical game) or holding properties of the player that is not associated with a certain unit agent, e.g. opponent modeling.

In addition, there may be environment variables or parameters common to the whole MAS. These may be static throughout the game, or dynamically changed by the game server, or the agents of the system. Examples of such variables may be attack and defence weights, territory valuations, or algorithmic parameters (such as search depths).

Note that the proposed architecture is not restricted to the general model of the game outlined in Section 2. It would be possible to have play games that involve skipping turns or change the order of the play.

4. TWO INSTANCES OF GAMES

We will now (briefly) describe two games, no-press Diplomacy and Risk[2], which on the surface seems to be similar, but that are quite different in the way they are played. Among the similarities, the goal of the players in both games

[1]In fact, the bishop has two: one for the white squares, and one for the black squares

[2]Both games are ©Hasbro inc.

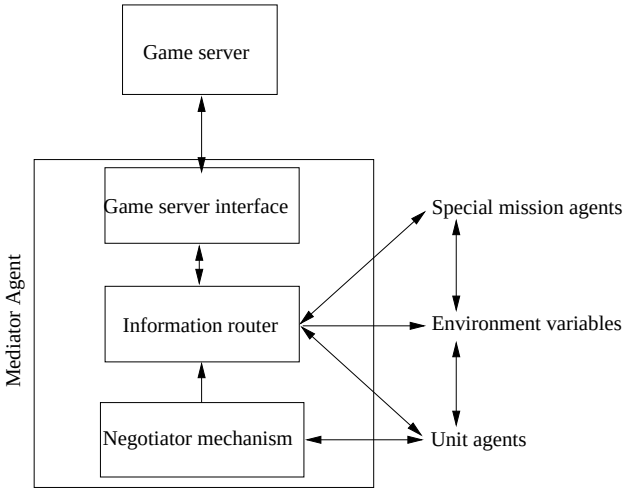

Figure 2: The mediator agent of the general bot

Property	Diplomacy	Risk	Chess
# players	7	2–6	2
Phases	$IAFAF$	IA^*F	A
Moves	parallel one-step	sequential multi-step	sequential one-step
Outcome	opponent dependent	dice-based	deterministic
Pieces	two types	single type	six types
Board	map	map	8x8
Pieces/prov.	0–1	1–∞	0–1

Table 1: Differences between the games Diplomacy, Risk and Chess. The phases are *Initial, Action* and *Finish* (IAF).

is to conquer the map by defeating their opponents, but that is where the differences begin. As a reference we have included Chess as well in the comparison (see Table 1).

We will now describe some central details of the games.

4.1 The Characteristics of no-press Diplomacy

In no-press Diplomacy, seven players each control one of the great powers in the 1901 Europe (see Figure 3). The players control armies and fleets and the goal is to take control over the majority of some special territories (the production centers). All pieces of all players act in parallell in the action phases, either by moving, supporting, cutting opponent support (all of these to neighbouring territories only), or standing still. For a move to be successful, the moving piece must be superior in uncut support if the opponent also tries to move to that position (or already resides there). After each phase where the pieces try to act, a finish phase decides what moves that were successful, and what moves that were not. After two rounds of movements, the production center ownerships decide what powers that will get new pieces to place at their territories (this finish one year in the game). To avoid stale mates in the games (especially when bots are playing), a common solution is to finish the game in 1950 or when no production centers have changed owners for the last five years.

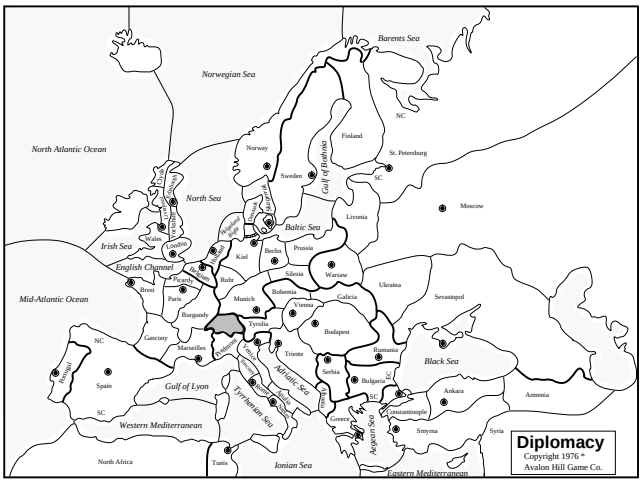

Figure 3: The map of the game of Diplomacy

Figure 4: The map of the game of RISK

4.2 The Characteristics of Risk

Risk is a game of two to six players, where the objective is to take control over the world (see Figure 4). This is achieved through invading opponent territories, and taking control over continents.[3] The players act in turns of three phases. In the initial phase, the player will get extra armies to place in its territories based upon the number of territories and the continents that they own. During the action phases, the player battles against opponents and tries to invade neighbour territories. This may be repeated for as long as the player has any armies left to attack with. In the finish phase, the player may move its armies. However, each army may only move one step and there must be at least one army left in each territory. If at least one battle has been successful, the player gets a bonus card that may be used in combination with other cards in the coming initial phases to get extra armies.

5. TWO INSTANCES OF BOTS

Following the general architecture of Section 3, we will now describe two bots able to play no-press Diplomacy and

[3]Many modern rule variations use special mission cards (where different players get different missions) to shorten the time it takes to play the game. However, we will not use missions cards in this work, but regard the total world domination as the common goal of all players.

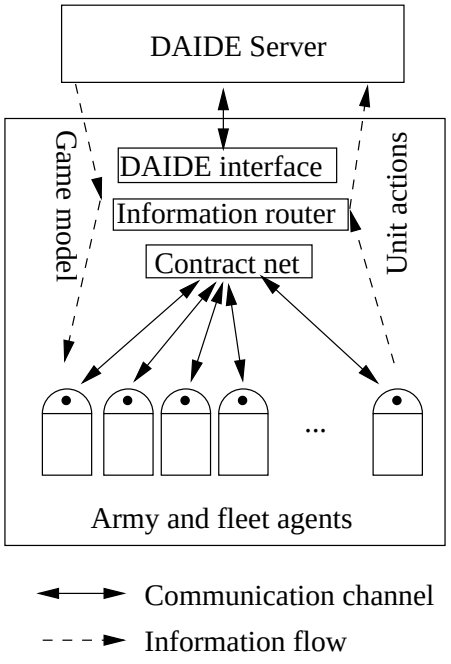

Figure 5: The no-press Diplomacy bot HaAI

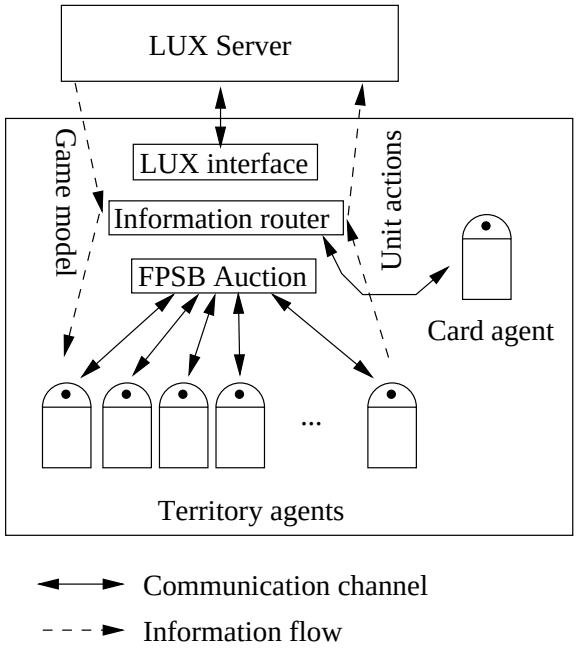

Figure 6: The RISK bot MARS

Risk respectively. For further details on the implementation we refer to previous work by Johansson and Håård [7] and Johansson and Olsson [8].

5.1 HaAI — A no-press Diplomacy Bot

The HaAI bot consists of a mediator agent and a set of unit agents. The mediator agent has an interface to the game server (in our case the DAIDE server [2]) and an information router that distributes the information going in and out of the system. To coordinate the actions of the units, a contract net-mechanism is used [17]. Here the units suggest prioritized tasks and adapt to support such tasks until all agents are doing the best they can to expand their territories and defend themselves from the opponent forces, see Figure 5.

The unit agents correspond to the armies and fleets of the player, so that each unit has the possibility to decide by itself what action to take (to move, to support the movement of another unit, or to cut the support of an opponent unit). A number of parameters are used in the evaluation of the different candidate actions [7].

5.2 MARS — A Risk Bot

The MARS bot consists of a mediator agent with a LUX interface, an information router, and a first price sealed bid (FPSB) auction mechanism that handles the coordination between the unit agents. In MARS, each territory has an agent (as opposed to each piece owned by the player, as was the case in the HaAI bot). A dedicated card agent keeps track of the cards of the player and exchange them for new armies in an optimal way (see Figure 6).

The initial phase starts by letting all territories (the own as well as the ones held by the opponents) calculate the value of holding that particular territory [8]. Such an approach, using the territories rather than the pieces, is as far as we know novel in the field of bots for board games. In the case of the own territory, this is a defensive bid, in the case of an opponent territory, the information of how many armies that defends it and the value of owning it is propagated to all neighbours. Their values and armies are added, the path is added (possible cycles are removed), and the information is propagated further in a recursive way (up to a predefined depth). When the paths with the values and number of defending armies reach an own territory, it is considered by the territory as an offensive bid. The agents of the own territories then calculate the accumulated probabilities of reaching the goals given the different numbers of armies that they may use in the attacks. The bids with the highest ratio of value/(expected number of armies needed to succeed) will get their share of the new armies. The card agent will also possibly cash in some extra armies for the system to use at this stage.

An action phase consists of a battle, where the plans of the winning bids of the initial phase are realised. Such plans may be single step plans, but often they consist of several battles carried out in sequence.

In the finish phase, the own territory agents use a distributed max flow algorithm to find the optimal movements of the armies. If MARS has won any battle during the action phase(s), the card agent will get a new card.

6. ON THE EVALUATION OF BOTS

We have used open bot tournaments as a way to measure the performance of our solutions. In both the Diplomacy and the Risk communities, there are bot developers that are willing to try their solutions in competition with others. Although previous work has presented successful attempts to use humans in training and/or evaluating bots (see e.g. [20]), there are a few reasons why we have chosen not to compare the solution with human players in these tournaments:

1. Bot tournaments can be batched, humans not. It is easy to set up a round robin tournament with say 1000 matches using a simple script, but to get a human player to participate to the same extent is impossible.

2. Bots are in general faster than human players, thus decreasing the time to get to the result.

3. Bots are blind and objective, while humans may decide to eliminate the bots first (just because they are bots). This *bots first strategy* is highly developed in the Risk (LUX) community. Unfortunately, such a strategy makes it hard to do a fair evaluation of the strengths of the different players.

In both cases we called for the participation of an open tournament through the fora and mailing lists of the communities. We invited the designers of bots for the largest game server platforms of the games (DAIDE and LUX respectively). The tournament was then run in a Round Robin fashion (i.e. everyone met every other set of opponents) and the scores of all matches were noted. We also measured the time it took for the bots to play the games in order to be able to compare the efficiency of the different solutions.

6.1 The Results of the no-press Diplomacy Tournament

A total of 592 matches were run on DAIDE out of which 487 ended with a single winner and 105 matches were draws. In case of a draw (no one has won after 50 simulated years), the score of seven points is divided between the remaining players. We see (in Table 2) that the two versions of HaAI manage quite well, ending up at the first and third place in the tournament.

What the efficiency is concerned, according to Table 3, HaAI does not have the best figures (not the worst either). It is clearly the fastest Java-based bot among the participators, but not as fast as the C++-implemented *DumbBot 2*.

6.2 The Results of the RISK Tournament

The RISK tournament was run on the LUX platform which is the platform for which there are the greatest number of bots available [16]. Twelve bots participated in addition to our MARS bot and played 792 matches each. MARS was a clear winner of this tournament with more than 50 per cent more wins than the second and third placed *Bort* and *EvilPixie*.

When it comes to efficiency, MARS still comes out at the top of the others. If we compare the amount of calculations needed (at an average) to win a game, we see that the ratio Wins/(average time per turn) is higher for MARS (even though it is among the slowest bots, see Table 5). This shows that MARS uses an efficient way of calculating good RISK moves.

Bot	Matches won	Solo score	Total score
HaAI 0.63 Berserk	109	763	866.05
DiploBot 1.2	95	665	763.10
HaAI 0.63 Vanilla	81	567	724.64
Man'Chi AttackBot	75	525	658.32
DumbBot 2	87	609	651.34
Man'Chi DefenceBot	40	280	444.27
RandBot	0	0	0.06

Table 2: Scores of the Diplomacy tournament. The total score include the scores of the draw matches.

Bot type	Orders/s	Impl. Language
DumbBot 2	55.9	C++
HaAI 0.63	19.6	Java
Man'Chi 7	12.7	Java
DiploBot 1.2	3.7	Java

Table 3: Diplomacy bot performance measured in number of orders per second.

Name	1st	2nd	3rd	4th	5th	6th	D
MARS	338	169	60	62	69	84	10
Bort	206	47	130	156	149	100	4
EvilP.	205	56	105	153	138	133	2
Boscoe	184	52	142	174	157	76	7
Yakool	135	57	118	162	167	146	7
Quo	177	103	102	132	134	142	2
Pixie	118	153	102	123	139	154	3
Shaft	117	370	157	61	36	38	13
Cluster	100	137	165	161	118	105	6
Nefar.	84	115	128	141	168	156	0
Comm.	16	278	199	130	85	74	10
Stinky	7	139	204	141	147	149	5
Angry	6	17	89	113	207	358	2

Table 4: The outcome of the RISK tournament. D is the number of draws. Note that each bot played 792 matches.

7. DISCUSSION

We will now comment the results, and discuss the possible general conclusions that may be drawn from the experiments.

7.1 Comments on the results

The results of the tournaments raise a lot of questions. Were they mostly due to an intelligent setting of the parameters of the environment? Did we use the best opponents? How did our choice of game platform effect the behavior of our solution? What are the drawbacks of the approach?

Let us start with the parameters. Both MARS and HaAI use a wide range of parameters.[4] In the case of HaAI, a first *ad hoc* setting of the parameters was made and minor adjustments led to the two versions that entered the tournament[5]. MARS used a slightly more structured approach where we iteratively narrowed the search space of the two major parameters — the bonus for being offensive and the threshold probability for attacking [13]. Of course the values of the parameters play an important role when it comes to the performance of the bot. However, we have no reason to believe that we are superior in this respect (compared to our competitors). On the contrary, we believe that great improvements can be achieved by using machine learning

[4]HaAI has about 11 main parameters, and MARS has about 14.

Name	$Win\%$	$\overline{t_{match}}$	$\frac{turns}{match}$	$\overline{t_{turn}}$	$\frac{Win\%}{\overline{t_{turn}}}$
MARS	42.7	79.72	32.0	2.49	17.1
Boscoe	23.2	40.04	27.1	1.48	15.7
EvilP.	25.9	44.10	25.1	1.76	14.7
Bort	26.0	45.75	25.5	1.79	14.5
Quo	22.3	33.87	21.6	1.57	14.2
Pixie	14.9	40.04	29.9	1.34	11.1
Yakool	17.0	37.62	22.7	1.66	10.3
Shaft	14.8	74.47	34.5	2.16	6.86
Cluster	12.6	54.32	25.4	2.14	5.90
Nefar.	10.6	46.82	18.7	2.51	4.22
Comm.	2.0	49.37	29.1	1.70	1.18
Stinky	1.1	48.88	24.0	2.04	0.54
Angry	1.0	41.33	14.7	2.82	0.36

Table 5: The win percentage, average runtimes (in seconds) per match and per turn, and number of wins per hour of processing time in the RISK tournament.

Game	# units	action space	players	action reliability
Chess	small	small	2	full
Diplomacy	medium	medium	7	depends on other players
Risk	high	medium	2–6	dice based
Go	high	medium	2	full
Settlers of Catan	small	medium	4	dice based
Monopoly	single	tiny	2–6	dice based
RoboRally	single	small	2–6	depends on other players
Backgammon	small	small	2	dice based

Table 6: Comparison between a number of different board games.

approaches to find the right set of parameters, but that has not been the focus of this project.

The platforms were chosen based on the number (and quality) of opponents, and the documentation of the system. In both cases, the chosen game platforms were the ones which had the largest number of competitive opponent bots (among the ones we chose from).

The set of opponents is of course questionable. We encouraged the communities to send in their bots to us for participation in the tournaments; thus we did what we could to get the best possible opponents. However, this does not mean that there are not any bots out there that for one reason or another did not wish to participate e.g. due to bugs, or the unwillingness to share their programming secrets with us (see e.g. Håård [5]).

In all we believe that the successful results in the tournaments are to a great extent the results of a well implemented distributed problem solving mechanism based on the MAS architecture outlined in Section 3. Possibly the choice of parameters and the set of opponents may have had an additional positive effect on the results.

Among the drawbacks of our solution are that it may be hard (in its pure form) to implement strategic sacrifices, since all unit agents are trying to maximize their own utilities and they do only use a single step look-ahead. The way to address this problem may be to use special mission agents that focus on strategic issues through dynamic updates of the utility functions.

So, what are the key factors that makes our approach work better than the opponent bots in the investigated domains?

Our hypothesis is that game domains holding the following properties are especially well suited for MAS-based bots (compared to traditional search tree-based bots):

- *Large number of units in the game.* When there are a lot of units (i.e. armies or territories), each of which behavior may be modelled separately by an agent, the MAS bot seems to be a dynamic, but yet easily understood solution.
- *Large action space.* In both Risk and Diplomacy, the number of actions a player may take in each turn is considerably larger than e.g. in chess. It does of course not *help* us to have many options, but since our solutions do not rely on opponent modeling and thereby a search in the tree of possible outcomes (c.f. minimax), they do not get caught in the exponential explosion of possible states that such a search leads to.
- *Large number of players.* A large number of players furthermore increases the difficulty to predict the future state of the game since it increase the branching factor.
- *Unreliable outcome of the performed actions.* The more precisely the effect of the actions a player take can be predicted, the easier it is to predict the future state of the game.

Support for this hypothesis is to be found both in previous work on bots for variants of Diplomacy [7, 9, 15] and Risk [8], but also in the less successful attempts to make bots based on MAS principles for playing e.g. Chess [3, 4]. However, it is still too early to draw any conclusions based upon these initial studies. More work is needed in the area before we may consider MAS based bots in general to be strong solutions in complex board game playing while being less appropriate to use in games requiring long look-ahead.

8. CONCLUSIONS

Our conclusion is that the MAS architecture described here may be a suitable candidate for creating competitive bots able to play games with high branching factors that are distributed in nature. We support this claim by reporting positive results from testing our solution in two different domains, namely the game of Diplomacy and the game of Risk. Despite the success in these isolated domains, we identify the need for more research within the area in order to draw any wider conclusions.

9. FUTURE WORK

There is support for the claim that complex domains such as the ones outlined in Section 7.1 above are favoured by multi-agent solutions [12, 18]. MAS based bots have shown to work well in Diplomacy and Risk, but what about the games which do not hold all the properties of these two

games? Go is one such game. It has got a large number of possible moves and a large number of stones[5]. However, the game has only two players and the moves are fully deterministic. It is therefore an open problem whether Go can be successfully played by a MAS based bot.

Settlers of Catan[6] is another game that has reached a large audience during the last years, especially through its online versions, e.g. *Java Settlers* by Robert Shaun Thomas [21]. Thomas used single agent bots and although he does not rule out the possibility to build bots with distributed architectures, he claims the single agent solution to be simpler (and thus better referring to the *less is more* principle) [20].

We will in our future work examine what game properties that are the most significant when it comes to the results of MAS based solutions. This will be done by trying a number of different suitable board games, e.g. Go and Settlers of Catan.

There are also a number of possible improvements regarding the roles of the special mission agents, e.g.:

- *Opponent short term modeling*, i.e. dynamic real time updates of the environment variables in order to adjust to the current opponent strategies.

- *Opponent classification* By learning the characteristics of different players and store the classification between the games, we may at an earlier stage identify what player that is playing what power and update the dynamic variables faster than what would have been the case if we used short term modeling only.

- *Parameter optimization*; the parameters of the environment may be updated based on the results of the games. This may effect the willingness to attack, or the utility of holding certain territories in the longer perspective.

- *Added functionality* In Diplomacy (without the prefix no-press), the players are allowed (and encouraged) to cooperate in order to achieve their goals. The system of e.g. Kraus and Lehmann [9] uses additional agents such as *minister of defence*, and *secretary of the prime minister* to deal with the negotiations with the opponent players. Such functionality may be added to the proposed architecture as special mission agents.

Last, but not least, we would like to carry out experiments with human players in order to measure the strength of our solution compared to the one of human players.

10. ACKNOWLEDGEMENTS

The author would like to thank Fredrik Håård and Fredrik Olsson for their hard work on the implementation of the bots and the creative discussions and co-authoring that led to our previous publications [7, 8]. We would also like to thank the game communities of DAIDE and LUX for the resistance in the tournaments, the reviewers of AAMAS'06 for the valuable comments and Blekinge Institute of Technology for funding this project.

[5]the pieces you put on the board in Go

[6]Trademark of Mayfair Games

11. REFERENCES

[1] Murray Campbell, Jr. A. Joseph Hoane, and Feng hsiung Hsu. Deep blue. *Artif. Intell.*, 134(1-2):57–83, 2002.

[2] DAIDE web page, http://www.daide.org.uk/, February 2006.

[3] A. Drogoul. When ants play chess (or can strategies emerge from tactical behviours?). In C. Castelfranchi and J.-P. Müller, editors, *From Reaction to Cognition — Fifth European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW-93 (LNAI Volume 957)*, pages 13–27. Springer-Verlag: Heidelberg, Germany, 1995.

[4] H. Fransson. Agentchess — an agent chess approach – can agents play chess? Master's thesis, Blekinge Institute of Technology, 2003.

[5] Fredrik Håård. Multi-agent diplomacy: Tactical planning using cooperative distributed problem solving. Master's thesis, Blekinge Institute of Technology, 2004.

[6] T.P. Hart and D.J. Edwards. The Tree Prune (TP) algorithm. Technical report, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1961. Artificial Intelligence Project Memo 30.

[7] S.J. Johansson and F. Håård. Tactical coordination in no-press diplomacy. In *Proceedings of Autonomous Agents and Multi-agent Systems (*AAMAS*)*, 2005.

[8] S.J. Johansson and F. Olsson. Mars a multi-agent system playing risk. In *Proceedings of Pacific Rim International Workshop on Multi-agents (PRIMA)*, 2005.

[9] Sarit Kraus and Daniel Lehmann. Designing and building a negotiating automated agent. *Computational Intelligence*, 11(1):132–171, 1995.

[10] Kai-Fu Lee and Sanjoy Mahajan. The development of a world class othello program. *Artif. Intell.*, 43(1):21–36, 1990.

[11] Daniel E. Loeb. Challenges in multi-player gaming by computers. *The Diplomatic Pouch Zine*, S1995M, 1995.

[12] M. Luck, P. McBurnley, and C. Preist. *Agent Technology: Enabling Next Generation Computing - A Roadmap for Agent Based Computing*. AgentLink, 2003. ISBN 0854 327886.

[13] Fredrik Olsson. A multi-agent system for playing the board game risk. Master's thesis, Blekinge Institute of Technology, 2005.

[14] J. Schaeffer. Solving checkers: First result. *International Computer Games Association (ICGA) Journal*, 28(1):32–37, 2005.

[15] J. Shaheed. Creating a diplomat. Master's thesis, Imperial College, London, UK, 2004.

[16] SillySoft. Lux v4.3, 2005. http://sillysoft.net/ URL last visited on 2006-01-24.

[17] R.G. Smith. The contract net protocol: High level communication and control in a distributed problem solver. *IEEE Transaction on Computers*, C-29(12):1104–1113, 1980.

[18] K. Sycara, K. Decker, A. Pannu, M. Williamson, and D. Zeng. Distributed intelligent agents. *IEEE Expert*, 11(6):36–46, 1996.

[19] G. Tesauro. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3), 1995.

[20] R.S. Thomas. *Real-time Decision Making for Adversarial Environments Using a Plan-based Heuristic*. PhD thesis, Northwestern University, Evanston, Illinois, 2003.

[21] R.S. Thomas. Jsettlers home page, 2006. http://catan.jsettlers.org/ URL last visited February '06.

[22] Caspar Treijtel. Multi-agent stratego. Master's thesis, Delft University of Technology, 2000.