Universidade de Aveiro Electrónica, Telecomunicações e Informática,
2011

Gustavo Martins Pereira Pires

Diplomacy - Tomada de Decisão Baseada numa Base de Dados de Movimento

Diplomacy - Decision Making Based on a Movement Database

Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática,
2011

Gustavo Martins Pereira Pires

Diplomacy - Tomada de Decisão Baseada numa Base de Dados de Movimento
Diplomacy - Decision Making Based on a Movement Database

Dissertacão apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica de Pedro Lopes da Silva Mariano Professor do Departamento de Informática da Universidade de Lisboa e Luís Filipe De Seabra Lopes, Professor do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

o júri / the jury

presidente / president

Prof. Doutor António Rui de Oliveira e Silva Borges
Professor Associado da Universidade de Aveiro

vogais / examiners committee

Prof. Doutor Luis Miguel Parreira e Correia
Professor Associado com Agregação do Departamento de Informática da Faculdade de Ciências da Universidade de Lisboa

Prof. Doutor Luís Filipe De Seabra Lopes
Professor Auxiliar da Universidade de Aveiro (Coorientador)

Prof. Doutor Pedro Lopes da Silva Mariano
Professor Auxiliar Convidado do Departamento de Informática da Faculdade de Ciências da Universidade de Lisboa (Orientador)

agradecimentos / acknowledgements

I would like to thank everyone involved in my learning process these past few years.
I would also like to thank Dr. Loeb for sending me a few articles that I was having difficulty obtaining.
Finally I would also like to give a special thanks to Dr. Pedro Mariano for his tireless support, and without whom I would surely be poorer in knowledge.

resumo / abstract

Diplomacy is a multiplayer board game, with simultaneous turn-based movements and its game-tree complexity is staggeringly large. Several approaches have been developed to handle this, such as multi-agent systems which seem to be the standard approach. This document describes an implemention of an approach to handle this problem by using stored results in a database to approximate a sub-game perfect equilibrium.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Diplomacy

Diplomacy is a board game created by Allan B. Calhamer and subsequently published by a number of companies, the latest of which is Avalon Hill, a division of Wizards of the Coast, itself a subsidiary of Hasbro[1]. Calhamer describes that he designed the game in mind for the players to achieve an equilibrium, never having anyone succeed in a decisive breakthrough. Although his ultimate goal was unrealized, the game gained increasing popularity and still has a large player base today[2]. Since 1988, there has been an annual convention, the WorldDipCon (World Diplomacy Convention), during which a tournament is organized for the title of World Champion. There have been a number of other conventions, most notable of which seem to be the DipCon (Diplomacy Convention) in the United States and the European DipCon in Europe[3][4].

There are also some online resources, such as the DipPouch[5] and its online magazine, which enable players to discuss different aspects of the game, from game tree complexity[9] to endgame instructions[10]. With the checkers having been weakly solved[11] and chess playing programs becoming fairly strong[12], most research has transitioned to other games with a larger game complexity, and as such, there has been an intensified effort to build a strong diplomacy playing program, with the DAIDE project being the most noteworthy effort.

The standard diplomacy board, as shown in Figure 1.1, depicts Europe, parts of North Africa and the Middle East and divides them into seventy-five provinces, thirty-four of which are designated as Supply Centers. These are differentiated from the rest by being depicted with pentagrams on them.

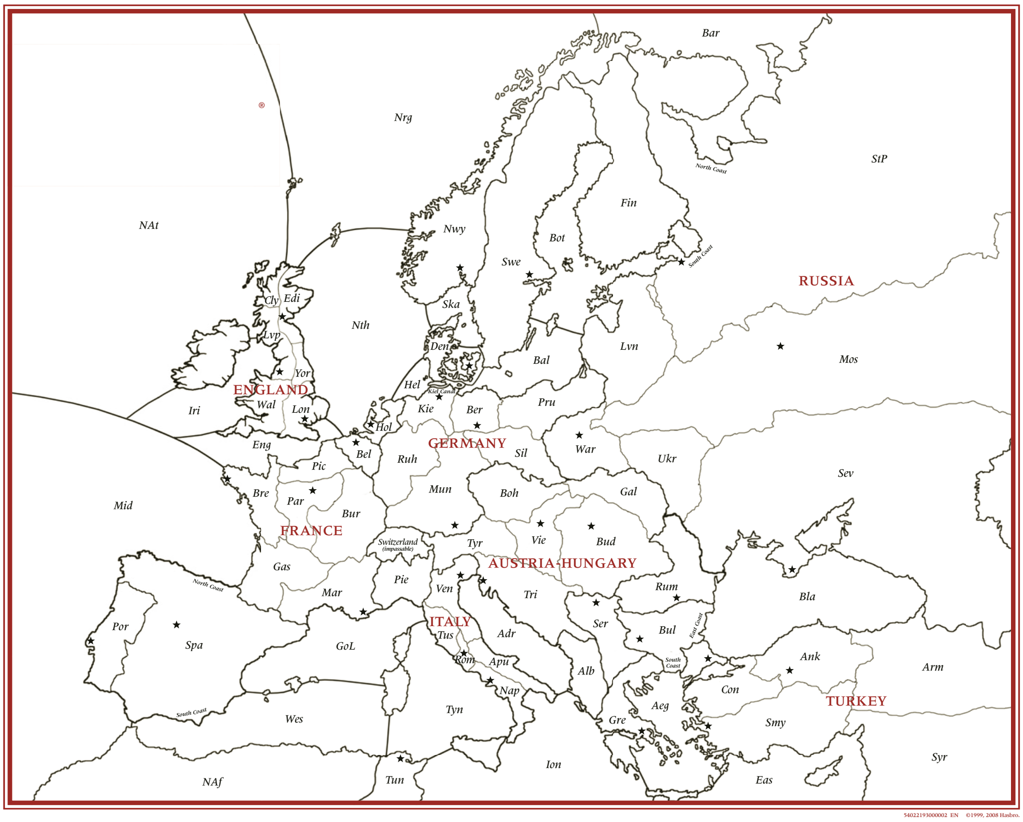

Figure 1.1: Standard Diplomacy Map
[1]

1.1.1 Basic Rules

The objective of the game to control over half of the Supply Centers on the map. Units can occupy a single province, and each province can only be occupied by a single unit at any time. There are three types of province. Land provinces, which can only be occupied by land units, known as armies. Sea provinces which can only be occupied by sea units, known as fleets. Finally there are coastal provinces, which can be occupied by both armies and fleets. These two types of units are the only ones used in the standard rules. Coastal provinces can also be multi-coastal provinces, limiting a occupying fleet's movement, depending on what coast the fleet is occupying. The standard map limits the number of coasts a province has to two, these being Spain, Bulgaria and St. Petersburg, making these particularly rare.

The game is divided into two turns, Spring and Fall. Each of these turns is further divided into 3 phases, with Fall having an additional phase at the end. A full year can then be described as the following:

1. Spring
    (a) Negotiation Phase
    (b) Movement Phase
    (c) Retreat Phase
2. Fall
    (a) Negotiation Phase
    (b) Movement Phase
    (c) Retreat Phase
    (d) Build Phase

1.1.1.1 Negotiation Phase

The rulebook actually doesn't mention any rules for this phase other than it lasting at most 30 minutes for the first one of these, and at most 15 minutes for the remaining ones. It also mentions that players may attempt to spy on each other during this phase, although for some reason this seems to be frowned upon. In this phase, like the name suggests, each player tries to convince other players, either in private or in public conversation, to act in a way that will ultimately favor them. Since diplomatic statuses are not defined, these relations can become quite complex and form the central theme of the game, diplomacy.

1.1.1.2 Movement Phase

The main phase of the game, which allows players to make a single movement with all of their pieces. These movements are:

Move
: A unit attempts to move to an adjacent province. A special case is the convoy move where an army attempts to move from a coastal province to another by means of a corridor of convoying fleets.

Hold
: A unit attempts to stay in the province they are currently occupying.

Support
: A unit attempts to support another unit. It can be thought of as adding strength to the supported movement. There are two types of support, support to move, and the support to hold.

Convoy

A fleet attempts to support an army attempt to transverse a sea province. A legal convoy needs a path from one coastal province to another to be composed of fleet units explicitly saying what move they are supporting.

At the end of the Movement Phase, after each player has written down their orders, an adjudicator is responsible to check which units succeeded. While a comprehensive algorithm would be too large to detail here, the following is a list of key points that help the adjudicator resolve the movement phase:

- Every movement that fails, with the exception of hold, makes the unit hold. If a hold fails, the unit is dislodged.
- A player can't dislodge his own units.
- When multiple moves of equal strength, IE. same number of supports, involve the same destination province, every one of them will fail.
- A move will only dislodge a unit if it has a greater strength than the unit holding the province.
- A support is only valid if it mentions the exact move the unit it is supporting will make.
- A support is cut if the unit is attacked, IE. there's a move against that province from another player, regardless of the success of the attack, and even if the attacked unit is dislodged.

1.1.1.3 Retreat Phase

After each movement phase, follows a retreat phase if there were any dislodged units. During this phase, the dislodged units can attempt to retreat to a non-occupied adjacent province, with the exception of the province from where the unit that dislodged them moved, or be disbanded.

1.1.1.4 Build Phase

The build phase, also known as the adjustment phase, is characterized, as the name suggests, by the addition and/or removal of units from the board. During the build phase, the player is allowed to add as many units as the number of supply centers owned minus the number of units the player has on the board. If this value is negative, the player needs to remove the corresponding amount of units from the board. The units added are placed in whatever home Supply Center the player pleases, so long as he respects the one unit per province rule. In case the player can't or doesn't want to add units, he can waive his build rights for that turn.

1.1.1.5 Additional Notes

Although the rulebook only names the two turns after seasons, players usually name the phases instead of the turns. While there are many naming conventions, we would like in addition of giving a description of a two season year, as defined in the Basic Rules above, to give a description of a 5 season year, both of which enable us to understand the variable jargon most written documents about Diplomacy use.

In the five season year description, the negotiation and the movement phases of each turn inherit their name, becoming the Spring Movement phase and the Fall Movement phase. The retreat phases become the Summer Retreat phase and the Winter Retreat phase. And the build becomes the Winter Adjustment phase. A full year becomes:

1. Spring Movement
2. Summer Retreat
3. Fall Movement
4. Winter Retreat
5. Winter Adjustment

1.2 Motivation

The study of games has given us tools with which to study real life situations, which in turn can be modeled as games to reduce their inherent complexity. Many situations, such as auctions[6], elections[7] and even the evolution of certain traits[8], have been studied with help from these tools.

Diplomacy presently holds academic interest mainly due to three reasons:

- The game tree is extremely large, as an example, just the first phase's number of moves amounts to over five quadrillion[9].
- It is a multi-player game, and relations between player any two players are not necessarily zero-sum (unless, of course, in the trivial case where the two are the last remaining players).

- There is a diplomatic side to the game, in which the need for cooperation and negotiation between distinct entities with conflicting interests arises.

All these challenges have been tackled in different ways. This thesis deals with an approach attempting to meet the first of these problems.

Although significant advances have been made in two player games, with increasingly stronger programs being made for such games[13], many of which can now go toe to toe with the best human players in the world, multi-player games haven't had such advances.

Most approaches to this have been to try to generalize n-player games to two player games, with limited success. Most of the difficulties with this have to do with the fact that in n-player games, interactions between any two player may not in fact be zero-sum, which invalidates or otherwise makes harder to employ techniques used in two player games.

1.3 Objectives

With this work, we intend to implement an agent capable of playing diplomacy, which uses a previously developed database[31] to its advantage, and check the feasibility of this approach for future diplomats.

1.4 Document's Organization

This document is divided into four parts. In Chapter 2 an overview of the existing diplomats and related works is given. In chapter 3 we explain the architecture of our diplomat, and how each component works and communicates with the others. In chapter 4 we explain the concept of the created database, how it works and how it was built. In chapter 5 we present our experiments and their results. In chapter 6 we use the show what we concluded from the experiments and how future work may use these.

Chapter 2

Existing Works

2.1 DAIDE Project

The DAIDE (Diplomacy Artificial Intelligence Development Environment) Project was an attempt at making a framework, where diplomats, short name for diplomacy playing programs, didn't have to understand free text but could negotiate using a common language. The framework includes an adjudicator, the DAIDE Server, which also handles the communication between the different diplomats, and a Mapper which helps users to visualize the state of the board and even enables them to play against other users connected to the DAIDE Server, be it human or program. Table 2.1 presents the level of orders used by the DAIDE common language.

Every DAIDE diplomat must at least implement level 0 messages, which are used to communicate with the server hosting the game, ex. request the state of the board at the beginning of the turn, sending orders, etc. Also, because of the increasingly complex syntax, every diplomat wishing to implement a level should also implement the levels below that one. At level 10, messages focus on simple peace and alliance arrangements, without any sort of conditions attached. Level 20 messages focus on order and DMZ (demilitarized zone) proposals where a diplomat proposes orders for other diplomats to follow, or a zone where no unit is allowed to enter. Level 30 messages introduce more complex arrangements, where acceptance of the full arrangement requires the acceptance of all specific arrangements (AND statement) or of at least one of them (OR statement) and so on. The goal is to encourage diplomats to have an orderly improvement towards free text press, or as it is more commonly recognized as, natural language.

This language's syntax and structure was based on the DPP language, by Daniel Loeb, which was a previous attempt to provide a common simplified language for diplomats.

Level 0	No Press
Level 10	Peace and Alliances
Level 20	Order Proposals
Level 30	Multi-part Arrangements
Level 40	Sharing out Supply Centers
Level 50	Nested Multi-part Arrangements
Level 60	Queries and Insistences
Level 70	Request for Suggestions
Level 80	Accusations
Level 90	Future Discussions
Level 100	Conditionals
Level 110	Puppets and Favors
Level 120	Forwarding Press
Level 130	Explanations
Level 8000	Free Text Press

Table 2.1: DAIDE Press Levels

2.2 Existing Diplomats

While most advances in the area have been made in two player games, Diplomacy has attracted a large following in research. As such, a considerable number of playing programs have been made.

2.2.1 Non-DAIDE Diplomats

These diplomats were primarily made to face human players or to check test positions and were amongst the first diplomacy playing programs.

2.2.1.1 Israeli Diplomat

One of the first attempts at building a diplomat, which set a few trends for later diplomats to follow. As shown in Figure 2.1, it used a modular, distributed approach by delegating tasks and assigning them to sub-agents. For example, the Foreign Office creates and assigns one agent per player and when receiving strategies from the Ministry of Defense related to a specific player will redirect them to that players assigned agent to use as a basis for negotiation[14][15].

Supposedly it was capable of beating human players. Sadly, no implementation of it is currently available.

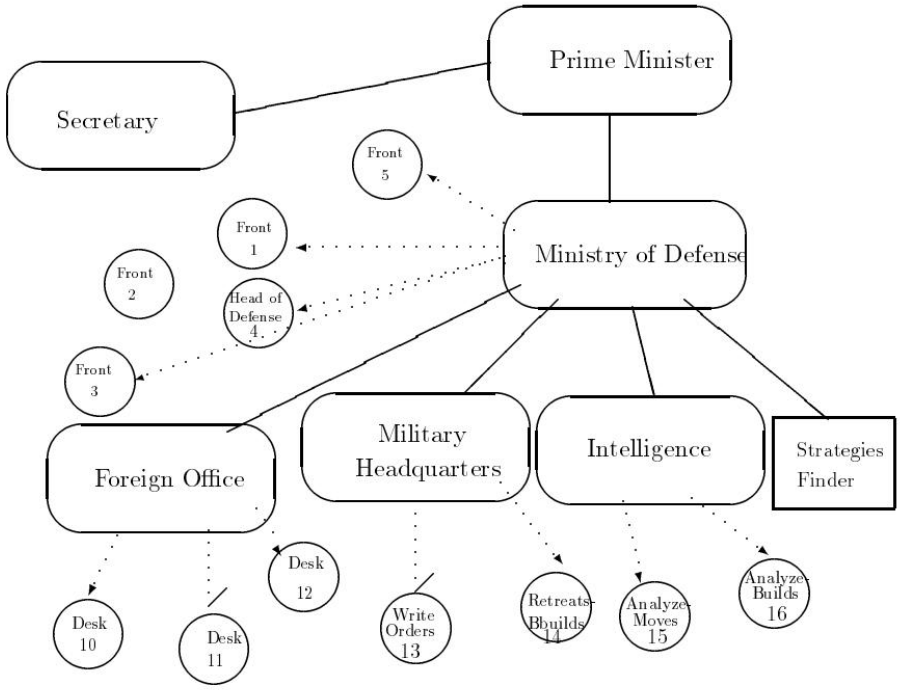

Figure 2.1: Israeli Diplomat's Arquitecture
[14]

2.2.1.2 The Bordeaux Diplomat

Made shortly after the Israeli diplomat, it used the province evaluator to create ʼfront linesʼ where it's units would move towards, and when they reached those positions, it would use a Best-First search coupled with an Evolutionary algorithm to transverse the game tree. Like the Israeli diplomat, no implementation of it is currently available[16].

2.2.2 DAIDE Diplomats

DAIDE Diplomats have a common simple negotiation language which enables them to attempt to negotiate with each other. Whilst most just use the server to act as a adjudicator in games, some have attempted to implement negotiations. Presently no diplomat exists with a press level above 30, and most diplomats don't implement above level 10.

2.2.2.1 DumbBot

Made as an example of an agent using the DAIDE protocol, this simple agent has received widespread usage as a sort of measuring stick between agents. According to its creator, it was an unexpected success, seeing as he only spent two hours on it. It has no negotiation capabilities. It uses a simple province evaluation heuristic described in further detail in 3.3

and then proceeds to calculate a value for each coast (IE. identical concept to the province node described in 3.3 based on attack potential, defense potential and nearby province values.

2.2.2.2 BlabBot

Using DumbBot as a basis, it is a remarkable example of what happens when you give a non-press agent a press heuristic. Whilst identical to DumbBot in non-press games, in press games its performance is much greater[17]. It begins the game by sending a peace proposal to every diplomat, and depending on the response, it adjusts the weights in DumbBot's heuristic to emulate the agreement. It is also capable of betraying other players if it considers a potential attack on their part to be a major threat[18].

2.2.2.3 HaAI

Demonstrating that a MAS approach was viable, it achieved a significant degree of strength, beating most of the competition at the time[19][20][21]. Each unit was assigned an agent which attempted to maximize its own payoff from a list of goals. This way the author attempted to obtain a globally optimal strategy from various locally optimum strategies.

2.2.2.4 Darkblade

This agent improved upon HaAI and showed an example of enemy movements' prediction[22][23]. It used potential fields to evaluate provinces and movement profit and a MAS approach to come up with different strategies to evaluate. As Figure 2.2 shows, it was also heavily influenced by the Israeli Diplomat's modular approach.

2.2.2.5 Albert

It probably is, at the time of writing (2011), the strongest diplomat currently available. It is, according to its creator, the logical follow-up to his previous agent, KissMyBot. It is also currently the only diplomat capable of level 30 press[24].

2.3 Similar concepts

There have been similar attempts to use databases, although mainly in two player games. While the goal of these attempts is very different, normally to solve the game, it's usage is somewhat similar. Some of the more interesting databases are described below.

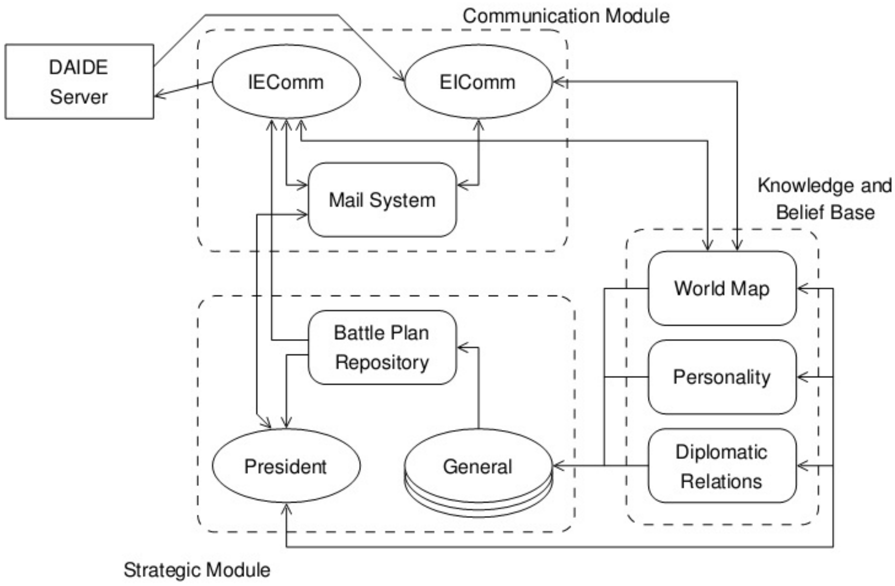

Figure 2.2: Darkblade's Arquitecture
[22]

2.3.1 Nine Men's Morris

A curious example, since the game is split into two phases. One in which both players decide the piece's placement, and the other where they actually play according to piece movement. The objective of the game is to leave the opponent with fewer than three piece or with no legal moves available. Each time a player creates a row of three of his pieces, called a mill, that player can remove a piece from the opponent which cannot be placed again. First the player will take turns in placing a single piece on the board represented in Figure 2.3. When each player has placed nine pieces on the board, regardless of how many actually remain on the board, the second phase begins. Then each player will move his pieces around in order to create mills or somehow corner the opponents pieces, fulfilling the game's objective.

A paper by Gasser shows that the game is drawn using a database to store the results of the last phase before using a traditional two-player game approach, alpha-beta pruning[25][36].

2.3.2 Chess Endgame Tablebases

An incomplete database, in that it's only partially solved. Complete endgame tablebases for up to 6 pieces exist and were completed in 2006 and 7 pieces tablebases have been estimated to be completed by 2016[26]. An early example of a 7 piece tablebase position is given in Figure 2.4.

These databases are computed with retrograde analysis, more commonly known as

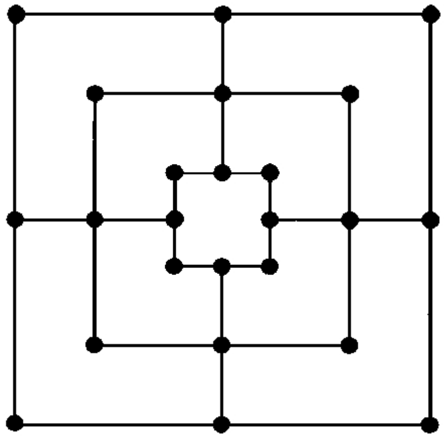

Figure 2.3: Nine Men's Morris Board

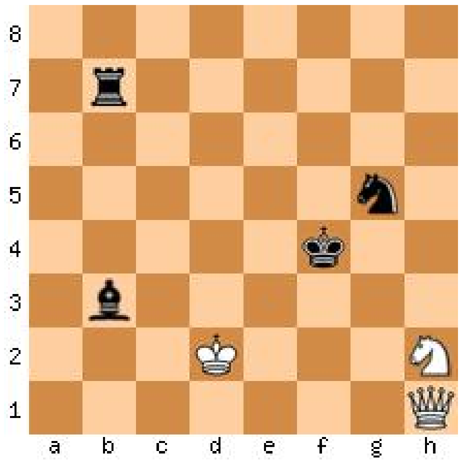

Figure 2.4: 7 Piece Tablebase Position: Black to Move, White mates in 517 moves [38]

---

backward induction, in which every possible unique mate position is obtained and then it maps out how other positions might arrive at the mate position. Any position in which a player may force the mating position to arise is a won position, every other is drawn.

Of note, is that human chess players have attempted to extract knowledge-based approaches from these databases[27].

2.3.3 Checkers Endgame Tablebases

Another example of an incomplete database, similar to the chess tablebases, of which complete tablebases exist for up to 10 pieces which were completed in 2005. Due to Chinook, most interest however fled to the 10x10 checkers variant, where tablebases exist for up to 8 pieces. They are obtained through a similar process to chess tablebases [37]. An example of

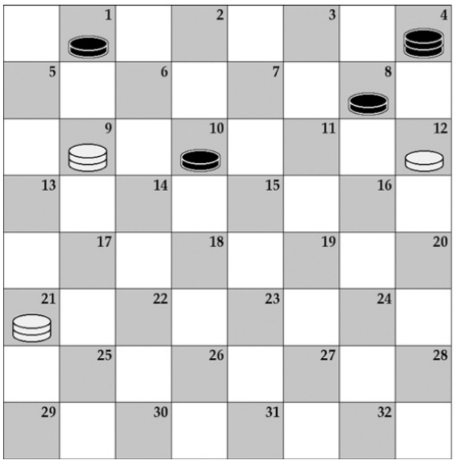

Figure 2.5: Longest 7 Piece Tablebase Position: Black to Move, wins in 253 plies [29]

a 7 piece tablebase position is given in Figure 2.5.

2.3.4 Checkers - Chinook

While slightly different from both examples given, in that it didn't attempt to strongly solve the game, Schaeffer showed that checkers is a draw[11]. The proof used both backward and forward search to establish the theoretical result, with a 10 pieces tablebase being generated for the backward search and a mixture of alpha-beta pruning and proof number search being used in the forward search.

Chapter 3

Blackmane's Architecture

The diplomat discussed over the course of this paper was named Blackmane after Ragnar Blackmane, a character in the Space Wolves novels by William King and Lee Lightner in an attempt to continue this department's tradition of naming diplomats after Warhammer characters.

Previous diplomats tried to cope with the staggering size of the game tree by dividing their decision making process amongst several agents, each controlling a single unit, and from there coordinating to reach a unified strategy[14][15][21][22]. While this has turned out to be a successful way of dealing with the problem, it doesn't blend well with the approach of having a database centered around conquering a single province. As such, our diplomat was instead centered around it's decision making object, the provinces themselves.

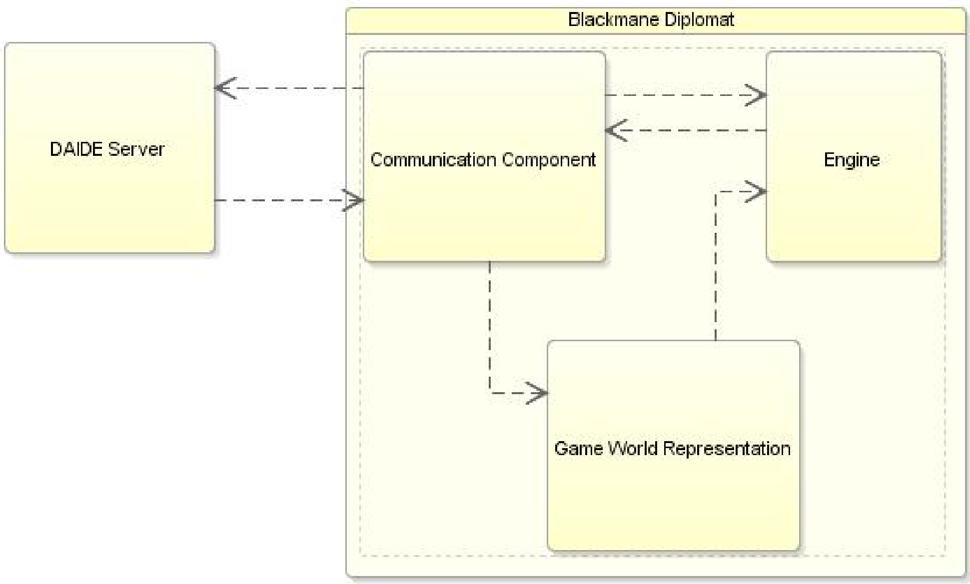

Figure 3.1: Blackmane's Arquitecture

The agent was divided into three components. A communication component that handles communication between the agent and the DAIDE Server, a world view component that contains information about the map, pieces and player arrangements and finally the engine itself, which uses the world view component to derive a strategy which will be sent to the communication component as can be seen in Figure 3.1.

3.1 Communication Component

The communication component was made using the JAC (Java AI Communication API) library by Daniel Yule and Henrik Bylund to handle communication from the DAIDE Server to the agent[30]. It breaks the server's messages into tokens, identifies the type of message and sends it to the proper place accordingly.

It's composed primarily of two objects:

- The server, which handles said communication between the server and the agent and message identification.
- The pressOffice, which handles communication between the server object and the engine.

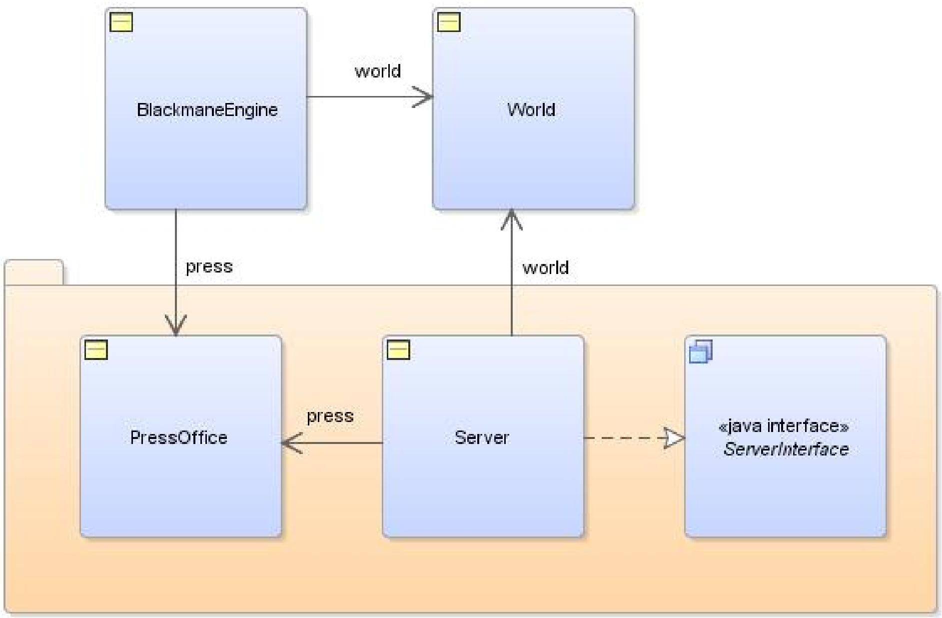

Figure 3.2: Architecture of the Communication Component - UML Class Diagram

As shown in Figure 3.2, the server implements JAC's ServerInterface, which creates a thread for each message received. Since our agent doesn't implement any press above level 0, i.e. it is a no press diplomat, it answers all messages with a level above that with a message that indicates it doesn't understand them, as detailed in the DAIDE Syntax document[28]. Before the game begins, it receives a message containing the map description which will then pass to the Game World Representation component for it to create the map representation. Every message from then on that doesn't detail the end of the game will go the pressOffice and from there to the engine. It also handles requests from the pressOffice by sending the appropriate messages to the DAIDE server.

The pressOffice object is a shared object between the server and the engine, which stores messages to and from the server and makes calls to the server in order for it to send messages to the DAIDE server.

3.2 Game World Representation

The agent represents the board by using a graph of nodes as detailed in Figure 3.3, and groups them into provinces. For example, a land or water province only have single nodes, whilst coastal provinces have at least two nodes, in order to represent the different possible movements in the province, which depend on the type of unit and even what individual node the unit is occupying. For instance, in the standard map, as detailed in Figure 1.1, Spain (Spa) would have 3 nodes. One for armies, and two for fleets with each representing the connectivity of each coast.

3.3 Engine

The engine is composed of several components as seen in Figure 3.4, amongst which the most important are the StrategicModule, the General and the DatabaseWrapper.

The General is the object responsible for the evaluation of each province, whose algorithm is described in this section. The DatabaseWrapper handles calls to the MovementDatabase or the Planner, depending on whether a previously calculated strategy exists. The StrategicModule calls upon both of these objects to extract information and creates a unifying strategy as detailed in Section 3.3.1.

The process of obtaining a strategy can be summarized in the following way:

- Evaluate each province.
- Obtain the first strategies that improve the occupation of the reachable provinces using a greedy algorithm.

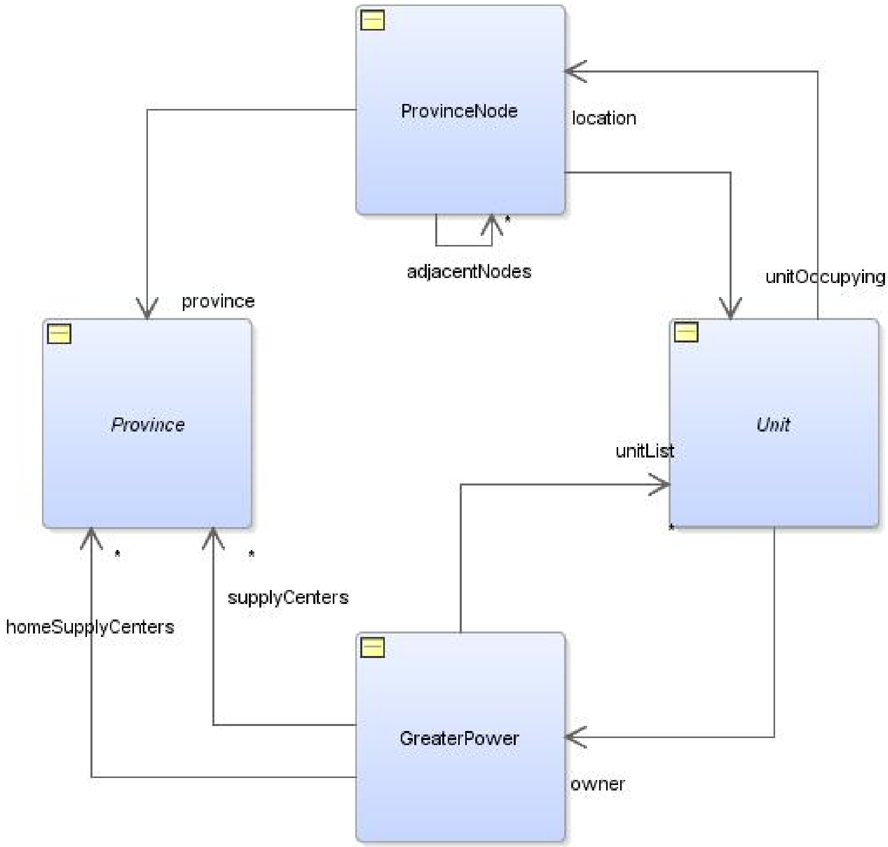

Figure 3.3: Game World Representation - UML Class Diagram

Initially, the way we evaluated provinces was identical to the way DumbBot province classification algorithm, which is:

$$Value = \begin{cases} P & \text{if it is our supply center} \\ N & \text{if it is not our supply center} \\ 0 & \text{if it is not a supply center} \end{cases} \tag{3.1}$$

with:
P = number of supply centers of the largest adjacent power in terms of supply centers
N = number of supply centers of the owning power

This way of classifying provinces also provides a way to see the usefulness of the planner and the database in a similar situation with which DumbBot deals. However, due to limitations in the planning algorithm, in particular our inability to look ahead more than one turn, and the fact that not all Supply Centers are adjacent with each other in the Standard Map as seen in Figure 1.1, meant we needed a way to differentiate non-supply centers too. So after

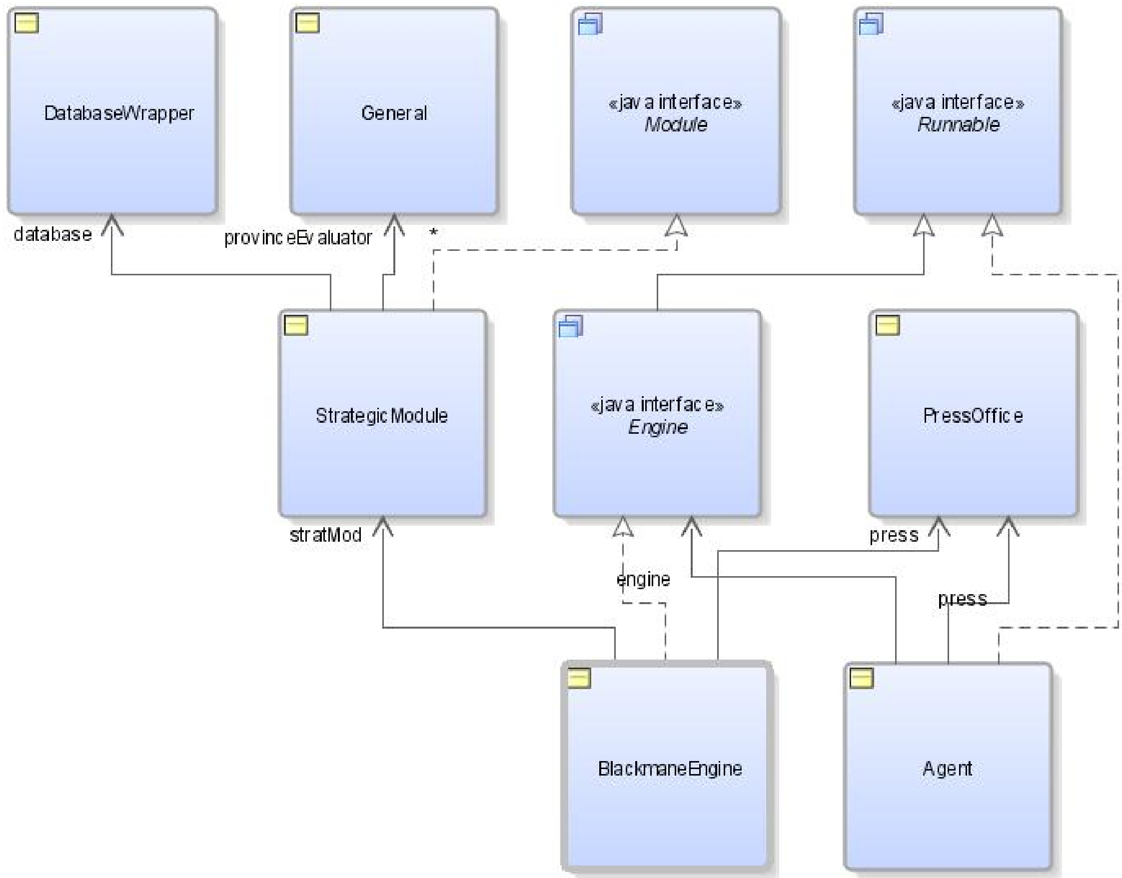

Figure 3.4: Blackmane's Engine - UML Class Diagram

---

by slightly modifying the above algorithm, we came up with:

$$Value = \begin{cases} 5 \times P & \text{if it is our supply center} \\ 5 \times N & \text{if it is not our supply center} \\ Q & \text{if it is not a supply center} \end{cases} \tag{3.2}$$

with:
Q = number of adjacent supply centers not owned by the player

3.3.1 Strategy

After evaluating each province and having a way to obtain a strategy to conquer each province, we need to see which strategies to choose. The outcomes mentioned below are obtained from the planner or the database, depending on if the sub-game has been previously calculated.

1. Obtain a list of reachable provinces, ordering them by value from highest to lowest.
2. For each province:
   (a) Obtain a list of possible sets of units in reachable radius of the target province that have not been used and order by them by number of units.
   (b) Obtain outcome if no owned unit is used.
   (c) For each set of units:
      i. Obtain strategy and outcome.
      ii. If the province is a Supply Center, check to see if there has been an improvement when compared to the outcome with no owned units. If so, identify the units used so they don't get used again, add the respective orders to the order list and move on to the next province on the list.
3. If there are units with no orders, order them to move to the most valuable adjacent province available or support a move there, provided here is a move to that province in the order list.

The last step was later revealed to be the most influential one in the agent's performance, and from here on out we'll call this last step the fall-back heuristic, for being the default strategy in case the we fail to find a strategy where that unit makes a difference.

Chapter 4

Movement Database and Planner

Because the planning algorithm is both slow, and not very scalable, it was considered storing the obtained results in a database. Thanks to this, not only can the agent obtain previously calculated strategies relatively fast from the database, but can also use these previously calculated results to calculate further into a position using an approach similar to memoization, considerably speeding up the whole process. Work by Rui Deyllot showed an example of such a database[31].

However, after careful analysis, this work revealed a serious flaw, which was the use of the minimax algorithm, used for sequential games. Because Diplomacy is a simultaneous game, using the minimax algorithm is the same as assuming that one of the players possesses knowledge of the other's strategy, a sort of oracle, which leads to strange, although interesting, situations and results in non-optimal strategies.

As an example, we can begin by analyzing another fairly well known simultaneous game, Rock-Paper-Scissors. The normal form is represented by the following matrix, with the row player's payoff being presented before the column player's payoff.

	Rock	Paper	Scissors
Rock	0 0	-1 1	1 -1
Paper	1 -1	0 0	-1 1
Scissors	-1 1	1 -1	0 0

Table 4.1: Rock-Paper-Scissors payoff matrix

The Nash Equilibrium of this game is a mixed strategy of all three pure strategies having a probability of one third of being chosen. But by thinking of this game as a sequential game, the first player to move always loses, and as such, every strategy available for him has equal value.

In this game, it might seem not to have much of an influence if we assume a single interaction between players. However, in an iterative game, if the first player chooses a different strategy other than the Nash Equilibrium, and if the second player uses previous history to model his responses, the first player can be in quite a serious disadvantage. Let's consider another simultaneous game, Rock-Paper-Scissors-Gum, which is basically the same as Rock-Paper-Scissors but with a new strategy available for both players. The normal form, similar to the one before is:

	Rock	Paper	Scissors	Gum
Rock	0 0	-1 1	1 -1	½ -½
Paper	1 -1	0 0	-1 1	½ -½
Scissors	-1 1	1 -1	0 0	½ -½
Gum	-½ ½	-½ ½	-½ ½	0 0

Table 4.2: Rock-Paper-Scissors-Gum payoff matrix

We can see that the Nash Equilibrium of this game is still the same, and that the new strategy, Gum, while not being dominated by any of the other pure strategies, is dominated by the Nash Equilibrium. However, if we look at the game as being a sequential one, this strategy turns into the first player's optimal strategy, since it minimizes his loss. In such case, where one player wrongly believes the game to be sequential, even if the encounter between both players is a single interaction, the first player will always come out worse.

To address this, a new planning algorithm was required, and was addressed in the Section 4.1.

4.1 Planner

The planning algorithm consists of two main phases:

1. A tree traversal to obtain the payoffs for each set of orders.

2. Finding the Nash Equilibrium of the matrix obtained in the previous steps.

In order not to traverse the entire game tree, which was shown to be impractical due to its size, we limit the depth of the tree to a fairly small value and prune the orders that will definitely not further the goal of controlling the target province. To do this we divided the sub-map into 3 areas:

- An inner zone, the area where the units will have influence in the following turns (in the trivial case, the target province). It can be described as a function of its distance to the target province, $2 \times N_{turns} - 1$. See areas A in Figure 4.2.

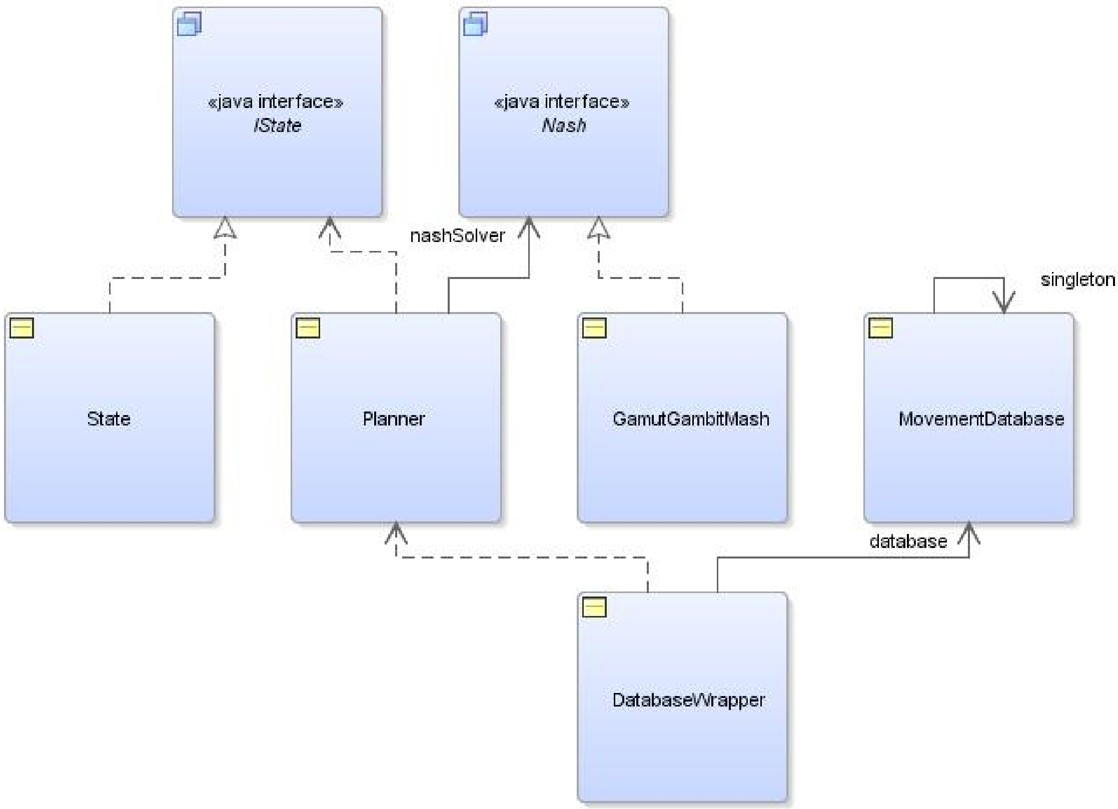

Figure 4.1: Database and Planner Architecture - UML Class Diagram

- A middle zone, which can be defined as the area not contained by the inner zone where units can influence what's happening in the inner zone (in the trivial case, the neighbor provinces of the goal province). Its function in relation to the distance to the target province is $2 \times N_{turns}$. See areas B in Figure 4.2.

- And an outer zone, the area not contained by either the inner or the middle zone, and where units can influence what's happening in the middle zone (in the trivial case, the neighboring provinces of the middle zone's provinces which do not neighbor the target province). Its function in relation to the distance to the target province is $2 \times N_{turns} + 1$. See areas C in Figure 4.2.

Units outside of these zones have no influence in the final outcome of the sub-game, which is to control, or lack thereof, of the target province. As an example, see areas D in Figure 4.2. In the trivial case of looking ahead one turn, the inner zone consists solely of the target province.

For each of these zones we can describe a behavior for the units inside them. For simplicity we included hold orders for all units, although if needed, a more aggressive pruning strategy

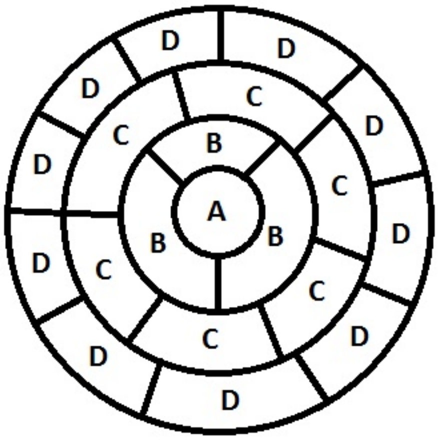

Figure 4.2: Example of the zone division in the Planner

could be applied.

For the outer zone this is:

- Move to a province in the middle zone occupied by enemy units, with the intent of interrupting a potential support.

For the middle zone this is:

- Move to a province in the inner zone, hoping to capture it or prevent its capture by enemy units.
- Support a movement by an allied unit to a province in the inner zone.
- Move to a province in the middle zone occupied by enemy units, with the intent of interrupting a potential support.

For the inner zone this is:

- Move to another province in the inner zone.
- Supporting a movement by an allied unit to any province in the inner zone.

Every order that is not a Hold or one of the orders defined by these guidelines is pruned.

After this pruning, we create combinations of every move and then we prune sets of orders that don't make sense, such as:

- Supporting void (supporting an action that doesn't exist).
- Conflicting moves (two friendly units trying to move into the same province).

- Conflicting move/hold (unit trying to move into another province which has another friendly unit holding).

In our implementation, the convoy order was not implemented when checking for possible orders. We assumed that lacking this order only had significant influence in England in the experiments, and because the other powers seem to very rarely use this order, it was also assumed that the overall results did not significantly deviate from a situation where that order would have been implemented.

The payoff of a node is calculated by checking the occupation of the target province. To encourage the continuous occupation of the target province, it was decided to calculate the payoff based not only on the leaf nodes, but also on the previous nodes, as given by:

$$N_{root} = \begin{cases} 1 & \text{if occupied by a friendly unit} \\ 0 & \text{if not occupied} \\ -1 & \text{if occupied by an enemy unit} \end{cases} \tag{4.1}$$

$$N_{lvl} = \begin{cases} \frac{F\times 1+N_{lvl-1}}{F+1} & \text{if occupied by a friendly unit} \\ \frac{F\times 0+N_{lvl-1}}{F+1} & \text{if not occupied} \\ \frac{F\times -1+N_{lvl-1}}{F+1} & \text{if occupied by an enemy unit} \end{cases} \tag{4.2}$$

Where F is a factor expresses the relative value of the importance between a node and its parent node. Our implementation gave F a value of 2.

After the tree traversal, we use linear programming to find a Nash Equilibrium. Initially, we tried to use Karmarkar's algorithm[33], but because of implementation issues and memory requirements, we decided to use the linear programming tool that is made available by Gambit[35], a library of software and tools with the purpose of analyzing games in either extensive or normal form, which uses the Lemke's algorithm[32][34].

Figure 4.1 exposes the architecture of the Movement Database and the Planner. The DatabaseWrapper checks the MovementDatabase to see if the required strategy was already calculated. If so, it simply returns it. If not, it uses the Planner to transverse the game tree, obtaining the payoffs for each set of orders, and then passes on this payoff matrix to the GamutGambitMash, which wraps Gambit's linear programming tool, to obtain a Nash Equilibrium. It then returns this strategy to the DatabaseWrapper which will store it in the MovementDatabase, before returning it to the requesting agent.

Strategy		
PK	province:	CHAR(3)
PK	turns:	INTEGER
PK	units:	CHAR(1024)
	solution:	CHAR(1024)
	payoff:	SINGLE

Figure 4.3: Table used for storage

4.2 Storage

Due to its small size and the ease of implementation, we decided to store the results in a SQLite database, even though there wasn't a need to use a relational database. Additionally, since retrieval of a strategy from this storage was far faster than the calculation of one, it suited our needs and we didn't find it necessary to implement a more specialized database.

A single table was created, as seen in Figure 4.3, whose primary key was a composite key formed by the target province, the considered number of turns and the units present on the sub-map. The associated data was the strategy obtained from those parameters via the planning algorithm, which consists of a description of the pure strategies used, followed by the probability that each pure strategy should be chosen and the payoff of the strategy.

The units are sorted lexicographically before being stored so as to avoid duplication of positions in the database with the only difference being the order by which the units are referred to.

Chapter 5

Experiments

5.1 Blackmane

Because of our objective was to test the feasibility of the planner/database instead of our agents performance, whose weak playing strength didn't help, instead of a victory/draw/loss table we found more instructive to present other relevant data to show how the planner and database performs, instead of focusing on the agent. Two separate experiments were setup. In the first, 30 games with equal number of instances of both Blackmane and DumbBot were played, with the Blackmane diplomat using the planner and the database. In the second, the setup was the same except the Blackmane diplomat didn't use neither the planner nor the database and just used the fall-back heuristic. The objective of this was to see the influence of the planner on the agent. We captured the logs made by the server and extracted the result of the game, IE. victory, draw or defeat, the maximum number of provinces held, and finally the number of provinces held at the time the game ended. Besides the data before mentioned, in the first experiment, we also collected the number of units that had orders given from the database and the owned number of units in every movement phase. Figure 5.1 shows the percentage of units that get their orders from the database, in relation to the movement turn number. A set detailing the percentage for each individual power is shown in Appendix A. We decided to use DumbBot because of the resemblance between its province evaluation heuristic and that of our agent's.

Before starting one parameter had to be decided, the number of turns the planner looked ahead, also known as the tree depth. A sample game was played with a two move look ahead between 6 DumbBot diplomats (which send their orders almost immediately, and so don't interfere) and a single Blackmane diplomat. Since the Spring 1902 turn was taking over 24 hours to complete, confirming the staggering growth of the game tree, we decided to terminate the game and proceed testing with a depth of one, effectively seeing just a turn ahead.

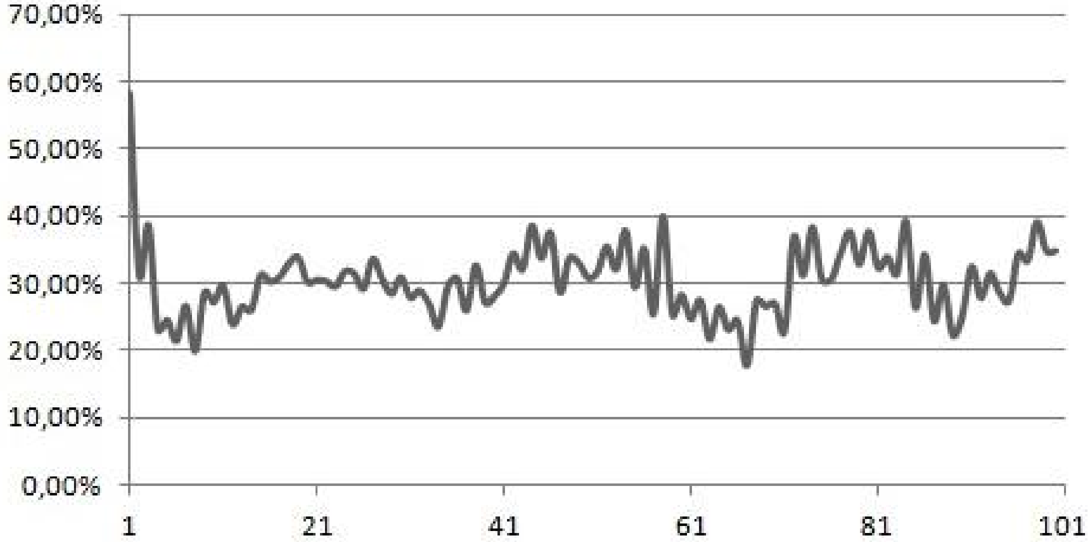

Figure 5.1: Units using moves from Planner/Database

Tables 5.1, 5.2, 5.3 and 5.4 show the diplomat's average maximum number of provinces, IE. their highest province count, and the average number of provinces they had at the end of the game.

Figures 5.2, 5.3, 5.4 and 5.5 show in more detail the differences in performance between the diplomats, when the Blackmane diplomat uses the planner and when it doesn't. Blue represents cases of not using the planner having higher averages and red represents cases where of using the planner having higher averages.

	ENG	FRA	RUS	ITA	TUR	GER	AUS	TOTAL
Using no Planner	3,44	2,08	2,44	1,94	1,92	1,50	3,29	2,36
Using Planner	1,93	2,37	2,00	1,50	2,71	1,07	1,23	1,89
Difference	1,51	-0,29	0,44	0,44	-0,78	0,43	2,05	0,48

Table 5.1: Blackmane's Average Final Number of Provinces

	ENG	FRA	RUS	ITA	TUR	GER	AUS	TOTAL
Using no Planner	6,00	5,00	5,00	3,44	4,85	5,50	6.05	5,09
Using Planner	4,60	6,47	5,00	4,83	5,00	5,73	4,85	5,28
Difference	1,40	-1,47	0,00	-1,39	-0,15	-0,23	1,20	-0,19

Table 5.2: Blackmane's Average Maximum Number of Provinces

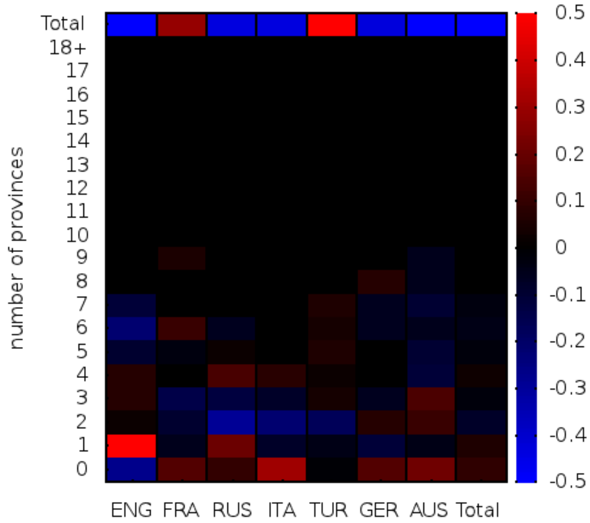

Figure 5.2: Final number of provinces held by Blackmane; Difference between using and not using the planner

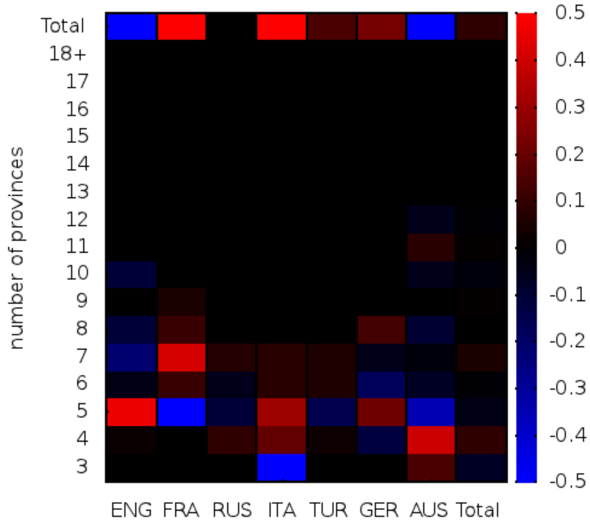

Figure 5.3: Maximum number of provinces held by Blackmane; Difference between using and not using the planner

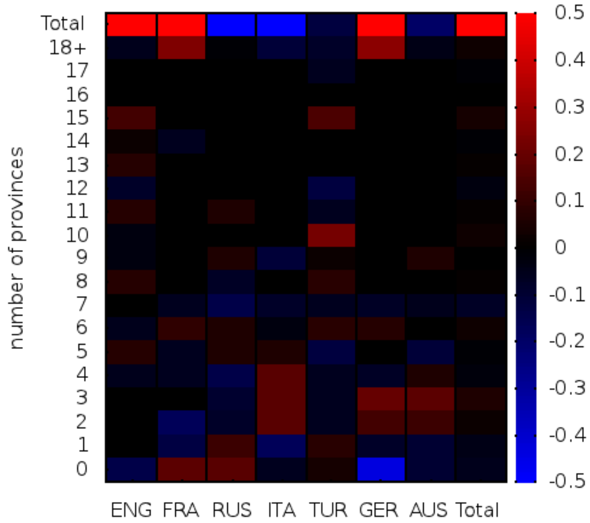

Figure 5.4: Final number of provinces held by DumbBot; Difference between games with Blackmane using and not using the planner

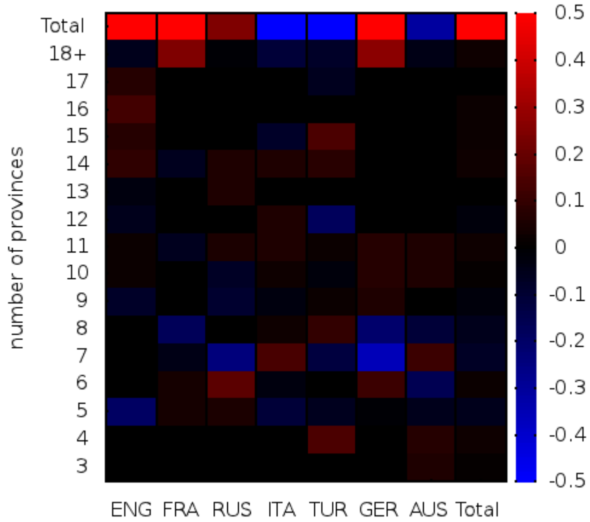

Figure 5.5: Maximum number of provinces held by DumbBot; Difference between games with Blackmane using and not using the planner

	ENG	FRA	RUS	ITA	TUR	GER	AUS	TOTAL
Games with no Planner	11,52	9,22	4,43	8,75	9,35	1,14	7,56	7,79
Games with Planner	13,47	12,09	3,75	6,61	9,23	6,33	7,35	8,13
Difference	-1,94	-2,87	0,68	2,14	0,12	-5,19	0,20	-0,34

Table 5.3: DumbBot's Average Final Number of Provinces

	ENG	FRA	RUS	ITA	TUR	GER	AUS	TOTAL
Games with no Planner	12,57	11,89	8,50	11,42	11,88	6,86	9,78	10,67
Games with Planner	15,00	13,73	8,75	10,72	11,08	10,07	9,47	11,10
Difference	-2,43	-1,84	-0,25	0,69	0,81	-3,21	0,31	-0,43

Table 5.4: DumbBot's Average Maximum Number of Provinces

Observing Figure 5.3 and Table 5.2 we can see that the planner seems to help on average in the expansion of the diplomat, however a number of conflicting observations can be gathered from these results. While France and Italy seemed to have improved significantly with their expansion thanks to the planner, England and Austria seemed to have worsened their expansion.

When checking on Figure 5.2 and Table 5.1 however, it seems the planner had the opposite effect on the diplomat, worsening its chances for survival. One could argue that a fast expansion, triggered a more aggressive response from the other diplomats. A counter-argument to this though, would be the situation when playing with Austria, where its score without the planner is still considerably better its score with the planner, in spite of the first having a larger expansion.

Regarding DumbBot, there is a notable observation in that it seems to have a very large performance difference when playing with Germany, depending on our use, or lack thereof, of the database as can be seen reading Tables 5.3 and 5.4, and Figures 5.3 and 5.4, which we had difficulty finding an explanation.

5.2 Database and Planner

After running 30 games with the planner, our database grew to about 60000 entries. This led to the initial turns of the game being played rather fast when compared a non-storage approach. Also noticeable were draw situations where after a few turns, the diplomat started to use only the database to get its orders. To test how much the database accelerated our diplomat, we decided to run three games populated solely with Blackmane diplomats. One of the games had them using the planner without the database, the others had them use the planner with the database; initially with the database obtained from the previous 30 games, and then again stored results from the database using Blackmane diplomats.

In the period of 1 minute:

- The non database using Blackmane diplomats played until Fall 1904.
- The initial set of database using Blackmane diplomats played until Fall 1906.
- The last set of diplomats played until Fall 1908.

Hence, we see the database enabling a significant speed up.

Chapter 6

Conclusion

6.1 Comments

As the game progresses, the planner seems to stop being used, in favor of the fall-back heuristic, which means that the planner finds fewer plans that have a positive outcome, ie. plans that succeed in controlling a province that wasn't in our control, and fewer plans that even have influence on the outcome. This may have to do with the subgames not being independent of each other and/or the diplomat simply having a bad position. The data detailed in Appendix A does not seem to support this last explanation, with possession of large number of units seemingly correlating to fewer moves taken from the database. In the end, this makes the fall-back heuristic have a considerable influence on the playing strength of the diplomat.

Based on this, we conclude that this approach, based on a pure usage of this approximation of a sub-game perfect equilibrium doesn't seem to be viable.

Regarding the use of a database, apart from the beginning of the game and in the few instances where the game reached endgame-like character positions, only about a third of the units used movements taken from the database. This suggests however two different uses for the database than those intended may be viable. One is as an opening database, which seems to be common in most board game engines. The other, is as a endgame database, detecting forced draws through draw lines in a position or standard stalemate configurations.

On a more practical note, trying to find Nash Equilibrium was found to be a slow process, even in sub-games with some pruning and a limited number of units.

6.2 Future Work

In order to increase the amount of plans the diplomat from the database that are used, we suggest the following:

- The planning algorithm be modified to reach a closer approximation of sub-game perfect equilibrium. As it is, every enemy unit of the most numerous opponent participates in the sub-game, some of which may take distinct roles in other sub-games, while influential units of a less numerous opponent do not participate in the sub-game.
- The planning algorithm be modified to have additional parameters (ex. payoffs per province, cost per movement, alliances, etc.), with the idea of merging some of the more dependent sub-games.
- Increase the performance of the planner to reach a search depth to at least two in a feasible amount of time. Since there are no provinces in the standard map that have a distance of more than two to the nearest supply center, this would enable the planner to focus on conquering just supply center provinces.

Other uses for the database might also be interesting to explore, in particular for endgame positions, which doesn't seem to have attracted as much attention as other aspects of the game.

Appendix A

Movement Information

The following tables describe the first 50 movement turns of the games where the Blackmane diplomat used the planner and the database. The first column indicates the turn number, 1 being Spring 1901, 2 being Fall 1901, etc. The second column indicates the number of games the diplomat survived to reach that turn. The third and fourth column indicate the number of units that got their movement that turn from the planner or the database and the total number of units, each of these on average per game reaching that turn. The final column indicates the percentage of units that got orders from the planner or the database that turn.

As an example, a table entry containing:

Turn	Number of games	Moves from database	Number of Units	Database/Total
27	7	2,00	3,00	66,67%

Means that 7 games were played that the diplomat reached turn 27, which is Spring 1915, where upon they had three units on the board on average, and gave orders to two of those from strategies taken from the planner or the database. This means that about 66.67% of those units had orders taken from the planner or the database while the remaining used the fall-back heuristic.

Turn	Number of games	Moves from database	Number of Units	Database/Total
1	13	2,00	3,00	66,67%
2	13	0,54	3,00	17,95%
3	13	1,38	3,77	36,73%
4	13	0,31	3,77	8,16%
5	13	0,38	3,00	12,82%
6	13	0,23	2,92	7,89%
7	13	0,15	2,69	5,71%
8	13	0,23	2,46	9,37%
9	13	0,54	2,00	26,92%
10	13	0,23	2,00	11,54%
11	13	0,38	1,92	20,00%
12	13	0,08	1,92	4,00%
13	10	0,50	2,40	20,83%
14	10	0,60	2,30	26,09%
15	9	1,11	2,67	41,67%
16	9	0,78	2,56	30,43%
17	8	1,00	3,00	33,33%
18	8	1,38	3,00	45,83%
19	8	1,13	3,13	36,00%
20	8	1,13	2,88	39,13%
21	8	0,75	2,38	31,58%
22	8	1,00	2,25	44,44%
23	8	1,00	2,00	50,00%
24	8	0,75	2,00	37,50%
25	8	0,75	1,88	40,00%
26	8	0,75	1,75	42,86%
27	8	0,88	1,88	46,67%
28	8	0,63	1,75	35,71%
29	7	0,71	1,86	38,46%
30	7	0,43	1,86	23,08%
31	7	0,71	2,00	35,71%
32	7	0,43	2,00	21,43%
33	7	0,71	2,14	33,33%
34	7	0,29	2,14	13,33%
35	6	0,83	2,50	33,33%
36	6	0,83	2,50	33,33%
37	5	1,20	3,20	37,50%
38	5	1,20	3,20	37,50%
39	5	1,40	3,40	41,18%
40	5	1,20	3,20	37,50%
41	5	1,40	3,20	43,75%
42	5	1,20	3,00	40,00%
43	5	1,60	3,20	50,00%
44	5	1,40	3,20	43,75%
45	5	1,60	3,20	50,00%
46	5	1,20	3,20	37,50%
47	5	1,00	3,60	27,78%
48	5	0,80	3,60	22,22%
49	5	1,60	3,60	44,44%
50	5	1,00	3,60	27,78%

Table A.1: Austria's Movement Information

Turn	Number of games	Moves from database	Number of moves	Database/Total
1	15	3,00	3,00	100,00%
2	15	1,80	3,00	60,00%
3	15	3,20	3,80	84,21%
4	15	1,87	3,80	49,12%
5	15	1,73	3,60	48,15%
6	15	1,67	3,53	47,17%
7	15	2,00	3,40	58,82%
8	15	1,47	3,33	44,00%
9	15	1,80	3,27	55,10%
10	15	1,80	3,27	55,10%
11	15	1,67	3,00	55,56%
12	15	1,47	3,00	48,89%
13	15	1,73	2,60	66,67%
14	15	1,47	2,60	56,41%
15	14	1,93	2,64	72,97%
16	14	1,50	2,64	56,76%
17	12	1,83	2,83	64,71%
18	12	1,67	2,83	58,82%
19	10	2,30	3,10	74,19%
20	10	2,00	3,10	64,52%
21	10	2,40	3,60	66,67%
22	10	2,40	3,60	66,67%
23	10	2,10	3,80	55,26%
24	10	2,50	3,80	65,79%
25	10	2,10	3,50	60,00%
26	10	2,30	3,50	65,71%
27	10	1,90	3,40	55,88%
28	10	1,90	3,40	55,88%
29	9	1,67	3,44	48,39%
30	9	1,89	3,44	54,84%
31	9	1,33	3,00	44,44%
32	9	1,89	3,00	62,96%
33	9	1,56	3,00	51,85%
34	9	1,78	3,00	59,26%
35	7	2,00	3,57	56,00%
36	7	2,14	3,57	60,00%
37	6	1,50	3,83	39,13%
38	6	1,50	3,83	39,13%
39	5	1,60	3,60	44,44%
40	5	1,60	3,60	44,44%
41	5	2,00	3,60	55,56%
42	5	1,80	3,60	50,00%
43	3	2,33	3,67	63,64%
44	3	3,33	3,67	90,91%
45	3	2,67	3,00	88,89%
46	3	3,00	3,00	100,00%
47	3	1,33	2,33	57,14%
48	3	1,67	2,33	71,43%
49	3	2,00	2,67	75,00%
50	3	2,67	2,67	100,00%

Table A.2: England's Movement Information

Turn	Number of games	Moves from database	Number of moves	Database/Total
1	19	3,00	3,00	100,00%
2	19	2,11	3,00	70,18%
3	19	2,79	4,79	58,24%
4	19	1,53	4,79	31,87%
5	19	1,37	5,05	27,08%
6	19	0,74	5,05	14,58%
7	19	1,47	4,89	30,11%
8	19	0,68	4,79	14,29%
9	19	1,26	5,00	25,26%
10	19	1,32	4,95	26,60%
11	19	1,16	4,89	23,66%
12	19	1,00	4,79	20,88%
13	19	0,74	4,74	15,56%
14	19	0,84	4,53	18,60%
15	18	1,06	4,22	25,00%
16	18	0,83	4,17	20,00%
17	18	1,06	4,22	25,00%
18	18	0,78	4,17	18,67%
19	17	1,00	3,65	27,42%
20	17	0,76	3,59	21,31%
21	17	0,88	3,71	23,81%
22	17	0,59	3,71	15,87%
23	17	0,76	3,71	20,63%
24	17	0,71	3,65	19,35%
25	17	0,71	3,71	19,05%
26	17	0,47	3,65	12,90%
27	17	0,94	3,53	26,67%
28	17	0,82	3,47	23,73%
29	16	0,75	3,63	20,69%
30	16	0,69	3,44	20,00%
31	15	0,87	3,67	23,64%
32	15	0,60	3,60	16,67%
33	15	0,67	3,67	18,18%
34	15	0,67	3,67	18,18%
35	14	0,57	3,43	16,67%
36	14	0,64	3,36	19,15%
37	14	0,64	3,29	19,57%
38	14	0,71	3,29	21,74%
39	13	0,54	3,08	17,50%
40	13	0,62	3,08	20,00%
41	13	0,38	2,69	14,29%
42	13	0,46	2,62	17,65%
43	11	0,55	2,45	22,22%
44	11	0,55	2,45	22,22%
45	10	0,50	2,70	18,52%
46	10	0,40	2,70	14,81%
47	10	0,50	2,70	18,52%
48	10	0,70	2,70	25,93%
49	9	0,44	2,33	19,05%
50	9	0,56	2,33	23,81%

Table A.3: France Movement's Information

Turn	Number of games	Moves from database	Number of moves	Database/Total
1	15	2,00	3,00	66,67%
2	15	1,07	3,00	35,56%
3	15	1,60	4,87	32,88%
4	15	0,80	4,87	16,44%
5	15	1,00	5,00	20,00%
6	15	1,33	5,00	26,67%
7	15	1,07	3,80	28,07%
8	15	0,87	3,73	23,21%
9	15	1,00	3,40	29,41%
10	15	0,93	3,27	28,57%
11	15	0,87	2,87	30,23%
12	15	0,87	2,80	30,95%
13	15	0,67	2,53	26,32%
14	15	0,53	2,40	22,22%
15	14	0,43	2,43	17,65%
16	14	0,57	2,43	23,53%
17	12	0,50	2,58	19,35%
18	12	0,33	2,50	13,33%
19	9	0,22	2,67	8,33%
20	8	0,13	2,88	4,35%
21	6	0,67	3,50	19,05%
22	6	0,67	3,33	20,00%
23	6	0,33	2,67	12,50%
24	6	0,17	2,67	6,25%
25	6	0,17	2,33	7,14%
26	6	0,17	2,33	7,14%
27	6	0,17	1,83	9,09%
28	6	0,33	1,83	18,18%
29	4	0,25	2,00	12,50%
30	4	0,00	1,75	0,00%
31	3	0,33	2,00	16,67%
32	3	0,00	2,00	0,00%
33	3	0,00	2,00	0,00%
34	3	0,33	2,00	16,67%
35	3	0,00	2,00	0,00%
36	3	0,33	2,00	16,67%
37	3	0,00	1,67	0,00%
38	3	0,33	1,67	20,00%
39	3	0,00	1,67	0,00%
40	3	0,33	1,67	20,00%
41	3	0,00	1,67	0,00%
42	3	0,33	1,67	20,00%
43	2	0,00	2,00	0,00%
44	2	0,50	2,00	25,00%
45	2	0,00	2,00	0,00%
46	2	0,50	2,00	25,00%
47	2	0,00	2,00	0,00%
48	2	0,50	2,00	25,00%
49	2	0,00	2,00	0,00%
50	2	0,50	2,00	25,00%

Table A.4: Germany's Movement Information

Turn	Number of games	Moves from database	Number of moves	Database/Total
1	12	0,00	3,00	0,00%
2	12	0,00	3,00	0,00%
3	12	0,00	4,00	0,00%
4	12	0,83	3,92	21,28%
5	12	0,33	3,83	8,70%
6	12	0,67	3,83	17,39%
7	12	0,58	4,00	14,58%
8	12	0,58	4,00	14,58%
9	12	0,75	3,83	19,57%
10	12	1,08	3,75	28,89%
11	12	1,00	3,67	27,27%
12	12	0,75	3,67	20,45%
13	12	0,67	3,33	20,00%
14	12	0,92	3,33	27,50%
15	12	1,17	3,33	35,00%
16	12	1,25	3,25	38,46%
17	11	0,73	3,27	22,22%
18	11	1,36	3,27	41,67%
19	10	1,20	3,20	37,50%
20	10	1,10	3,20	34,38%
21	10	0,90	3,10	29,03%
22	10	1,40	3,10	45,16%
23	10	1,50	3,30	45,45%
24	10	1,40	3,30	42,42%
25	10	1,50	3,10	48,39%
26	10	1,20	3,10	38,71%
27	10	1,10	2,90	37,93%
28	10	1,30	2,90	44,83%
29	9	1,11	2,56	43,48%
30	9	1,44	2,56	56,52%
31	7	1,00	2,86	35,00%
32	7	1,71	2,86	60,00%
33	7	0,86	2,43	35,29%
34	7	0,57	2,43	23,53%
35	7	1,14	2,57	44,44%
36	7	1,00	2,57	38,89%
37	6	0,67	1,83	36,36%
38	6	1,17	1,83	63,64%
39	6	0,67	1,83	36,36%
40	6	0,50	1,83	27,27%
41	6	0,50	1,83	27,27%
42	6	0,50	1,83	27,27%
43	5	0,20	1,40	14,29%
44	4	0,25	1,50	16,67%
45	3	0,33	2,00	16,67%
46	3	0,33	2,00	16,67%
47	3	0,33	2,00	16,67%
48	3	0,33	2,00	16,67%
49	3	0,33	2,00	16,67%
50	3	0,33	2,00	16,67%

Table A.5: Italy's Movement Information

Turn	Number of games	Moves from database	Number of moves	Database/Total
1	14	0,00	4,00	0,00%
2	14	0,14	4,00	3,57%
3	14	0,00	4,57	0,00%
4	14	0,71	4,14	17,24%
5	14	0,43	4,00	10,71%
6	14	0,43	4,00	10,71%
7	14	0,50	4,14	12,07%
8	14	0,50	4,07	12,28%
9	14	0,50	3,93	12,73%
10	14	0,64	3,86	16,67%
11	14	0,71	3,86	18,52%
12	14	0,50	3,71	13,46%
13	14	0,57	3,36	17,02%
14	14	0,50	3,29	15,22%
15	13	0,62	3,38	18,18%
16	13	0,77	3,38	22,73%
17	12	1,08	3,92	27,66%
18	12	0,83	3,67	22,73%
19	10	0,80	3,80	21,05%
20	10	0,70	3,80	18,42%
21	10	0,40	3,50	11,43%
22	10	0,70	3,50	20,00%
23	10	0,20	3,30	6,06%
24	10	0,50	3,20	15,63%
25	10	0,50	2,90	17,24%
26	10	0,50	2,80	17,86%
27	10	0,70	2,90	24,14%
28	10	0,20	2,90	6,90%
29	9	0,67	3,22	20,69%
30	9	0,56	3,22	17,24%
31	9	0,67	3,33	20,00%
32	9	0,56	3,33	16,67%
33	9	0,44	3,22	13,79%
34	9	0,33	3,22	10,34%
35	7	0,57	2,71	21,05%
36	7	0,43	2,71	15,79%
37	6	0,50	2,67	18,75%
38	5	0,40	2,80	14,29%
39	6	0,50	2,33	21,43%
40	6	0,50	2,33	21,43%
41	6	0,50	2,17	23,08%
42	6	0,67	2,17	30,77%
43	6	0,50	2,00	25,00%
44	6	0,83	1,83	45,45%
45	5	0,80	2,00	40,00%
46	5	0,80	2,00	40,00%
47	5	0,40	2,00	20,00%
48	5	0,60	2,00	30,00%
49	4	0,50	2,25	22,22%
50	4	0,50	2,25	22,22%

Table A.6: Russia's Movement Information

Turn	Number of games	Moves from database	Number of moves	Database/Total
1	17	2,00	3,00	66,67%
2	17	0,65	3,00	21,57%
3	17	1,82	4,00	45,59%
4	17	0,53	4,00	13,24%
5	17	1,35	3,71	36,51%
6	17	0,88	3,71	23,81%
7	17	1,00	3,59	27,87%
8	17	0,76	3,53	21,67%
9	17	1,06	3,12	33,96%
10	17	0,53	3,06	17,31%
11	17	1,06	3,00	35,29%
12	17	0,71	2,94	24,00%
13	16	1,06	3,44	30,91%
14	16	0,88	3,38	25,93%
15	15	0,67	3,13	21,28%
16	15	0,93	3,07	30,43%
17	15	1,00	3,13	31,91%
18	15	1,40	3,07	45,65%
19	15	1,13	3,20	35,42%
20	15	1,07	3,13	34,04%
21	15	1,00	3,20	31,25%
22	15	0,60	3,20	18,75%
23	14	0,86	3,50	24,49%
24	14	1,07	3,50	30,61%
25	14	1,14	3,93	29,09%
26	14	1,00	3,79	26,42%
27	14	1,14	3,64	31,37%
28	14	1,00	3,57	28,00%
29	13	0,77	3,54	21,74%
30	13	1,00	3,31	30,23%
31	12	0,83	3,50	23,81%
32	12	0,75	3,33	22,50%
33	12	1,00	3,42	29,27%
34	12	0,67	3,25	20,51%
35	11	0,91	3,27	27,78%
36	11	1,00	3,27	30,56%
37	9	0,67	2,89	23,08%
38	9	1,22	2,89	42,31%
39	9	0,78	3,11	25,00%
40	9	0,89	3,11	28,57%
41	9	1,11	3,22	34,48%
42	9	1,56	3,22	48,28%
43	7	1,14	3,71	30,77%
44	7	1,29	3,71	34,62%
45	7	1,00	3,71	26,92%
46	7	1,57	3,43	45,83%
47	7	1,57	3,71	42,31%
48	7	1,71	3,71	46,15%
49	7	1,29	3,57	36,00%
50	7	0,86	3,57	24,00%

Table A.7: Turkey Movement Information

Bibliography

[1] Avalon Hill, *Diplomacy*, available at http://www.wizards.com/default.asp?x=ah/prod/diplomacy

[2] Allan Calhamer, *The Invention of Diplomacy.* Available at http://www.diplom.org/~diparch/resources/calhamer/invention.htm

[3] European Diplomacy Association, *World Diplomacy Database.* available at http://eurodip.nuxit.net/php/rencontre/index.php?lang=Ang

[4] Wikipedia, *International prize list of Diplomacy.* available at http://en.wikipedia.org/wiki/International_prize_list_of_Diplomacy

[5] *The Diplomacy Pouch.* available at http://www.diplom.org/index.py

[6] William Vickrey *Public Economics: Selected Papers by William Vickrey.* Cambridge University Press, 1997.

[7] Kenneth J. Arrow, *Social Choice and Individual Values.* Yale University Press, 1951.

[8] John Maynard Smith and David Harper, *Animal signals.* Oxford University Press, 2003.

[9] Loeb, Selhammar, Green and Hand, *How Many Openings Are There?.* Available at http://www.diplom.org/Zine/S1995R/Misc/Blast.html

[10] Jamie Dreier, *The Endgame.* available at http://www.diplom.org/Zine/F1995M/Dreier/Endgame.html

[11] Jonathan Schaeffer, et al., *Checkers is Solved.* Science, 2007.

[12] Dylan Loeb McClain, *Man Vs. Computer Match Ends in Victory for . . . Man, but With a Catch.* The New York Times, 2008. available at http://gambit.blogs.nytimes.com/2008/10/01/man-vs-computer-match-ends-in-victory-for-man-but-with-a-catch/

[13] H. Jaap van den Herik, Jos W.H.M. Uiterwijk and Jack van Rijswijck, *Games solved: Now and in the Future.* Artificial Intelligence, 2002.

[14] Sarit Kraus and Daniel Lehmann, *DIPLOMAT, An agent in a Multi Agent Environment: An Overview.* In Proc. of the Seventh Annual IEEE Phoenix Conference on Computers and Communications, pp. 434–438, Arizona, 1988.

[15] Sarit Kraus and Daniel Lehmann, *Designing and building a negotiating automated agent.* Computational Intelligence, 1995.

[16] Michael R. Hall and Daniel E. Loeb *Thoughts on Programming a Diplomat.* 1992.

[17] John Newbury, *Tournaments for AI in the Game of Diplomacy: Tournament #9.* 15 May 2006, available at `http://johnnewbury.co.cc/diplomacy/tournaments/tournament9.htm`

[18] John Newbury, *AI in the Game of Diplomacy: the BlabBot Bot.* 19 October 2008, available at `http://johnnewbury.co.cc/diplomacy/blabbot`

[19] Fredrik Haard, *Multi-Agent Diplomacy: Tactical Planning using Cooperative Distributed Problem Solving.* Master Thesis, August, 2004.

[20] Fredrik Haard and Stefan J. Johansson, *Multi-Agent Tactics in Diplomacy.* Second European Workshop on Multi-Agent Systems, Barcelona, Spain, 2004.

[21] Stefan J. Johansson, *On using Multi-agent Systems in Playing Board Games.* ACM, 2006.

[22] João Ribeiro, *Darkblade - An Agent for Diplomacy.* Master's Thesis, Universidade de Aveiro, 2008.

[23] João Ribeiro, Pedro Mariano and Luís Seabra Lopes, *DarkBlade: a program that plays Diplomacy.* Progress in Artificial Intelligence: Lecture Notes in Computer Science, Springer, 2009.

[24] Jason van Hal *Albert.* available at `http://sites.google.com/site/diplomacyai/albert`

[25] Ralph Gasser, *Solving Nine Men's Morris.* Games of No Chance, MSRI Publications, Volume 29, 1996.

[26] Joe Hurd and Guy Haworth, *Data Assurance in Opaque Computations.* Advances in Computer Games, 12th International Conference, 2009.

[27] John Nunn, *Secret of Rook Endings.* Gambit Publications, 2nd edition, 1999.

[28] *Diplomacy AI Development Environment Message Syntax* available at http://www.ellought.demon.co.uk/dipai/daide_syntax.pdf

[29] Edward Trice and Gilbert Dodgen, *The 7-piece Perfect Play Lookup Database for the Game of Checkers.* In van den Herik et al., 2003.

[30] Daniel Yule and Henrik Bylund, *Java AI Communication API* available at http://web.unbc.ca/~yuled/dip/

[31] Rui Deyllot, *Diplomacy – Base de Dados de Movimentos para Controlar Províncias.* Master's Thesis, Universidade de Aveiro, 2010.

[32] Daphne Koller, Nimrod Megiddo, and Bernhard von Stengel, *Efficient computation of equilibria for extensive two-person games.* Games and Economic Behavior , 14, 1996.

[33] Ilan Adler , Narendra Karmarkar , Mauricio G. C. Resende , Geraldo Veiga, *An Implementation Of Karmarkar's Algorithm For Linear Programming.* 1986.

[34] Carlton Lemke, *Bimatrix Equilibrium Points and Mathematical Programming.* Management Science, 11, 1965.

[35] McKelvey, Richard D., McLennan, Andrew M. and Turocy, Theodore L., *Gambit: Software Tools for Game Theory.* 2007.

[36] Edwards, D.J. and Hart, T.P., *The Alpha-Beta Heuristic.* 1961.

[37] Yngvi Bjornsson, Jonathan Schaeffer, and Nathan R. Sturtevant, *Partial Information Endgame Databases.* Advances in Computer Games 11, 2005.

[38] Tim Krabbé, *OPEN CHESS DIARY 301-320.* available at http://timkr.home.xs4all.nl/chess2/diary_16.htm