Improving user experience in StateCraft

THESIS

By Christoph Carlson and Mathias Hellevang

Department of Information Science and Media Studies
University of Bergen

Spring 2010

Abstract

User experience is an important aspect of game development, and techniques in Artificial Intelligence have been proved to increase the agent performance and have the potential to affect user experience in games. In this thesis, we aim to improve the user experience of the computer game known as StateCraft by extending the computer controlled opponent with the Emotion module and the Prisoner's Dilemma module. We conducted simulations and user tests to study the effects of these modules on the agent performance and user experience.

The Emotion module gives the autonomous agent in StateCraft emotions. Existing research shows that emotions affect the decision making of the agent and make it appear more human-like. Statistic analysis of the simulation data indicates that agents perform worse in general with emotions than without. Based on the study performed on the user test of StateCraft with and without the Emotion module, we are not able to make a clear conclusion on the user experience, but the results *indicate* that playing against emotional agents is more fun, hence increases the user experience.

The Prisoner's Dilemma module gives the autonomous agent in StateCraft an understanding of the concept of the Prisoner's Dilemma. StateCraft contains several situations similar to the ones expressed in the Prisoner's Dilemma, and we wished to study whether the agent would perform differently if it understood this concept. The module was implemented with two strategies for the Prisoner's Dilemma module, Tit-For-Tat and Freidman, which both modify the behaviour of the agent according to the original specification of the strategies. Statistic analysis of the simulation data shows that Tit-For-Tat and Freidman lead to a more balanced game when all the agents are using the same strategy. The results from the user test with and without the Prisoner's Dilemma module were ambiguous, and thus indicate that further research is required to study the effects of these strategies on user experience.

Acknowledgements

First and foremost we would like to extend our gratitude to our supervisor, Weiqin Chen, for her support and guidance. You are an excellent supervisor, and we consider ourselves lucky to have had the pleasure of working with you. Additionally, we would like to thank Aleksander Krzywinski for his help and guidance on the domain of StateCraft. Aleksander gave us several ideas for the development, and he refactored StateCraft extensively to make the debugging process easier. This could not have been done without you. Thank you!

We would also like to thank our co-students at room 638. Thor Møller, Tone Nordbø, Rune André Liland, Hermund Furu, Christer Nordberg and Hans Terje Møller - you guys rock! We could always count on you to deliver whenever we needed social input, guidance regarding the APA-style, fashion advice, someone to share a meal with, or simply company when we wanted to play around with RC helicopters. You were our family for two years, and we thank you for this.

Charlotte Lilleheil, Thomas Nordeide and Petter Brodin, you were never a part of room 638, but we would never the less like to extend our gratitude towards your kinship. We could always count on you to embarrass us publicly, share a meal and make excellent coffee (all things considered).

We would also like to thank our parents and close family for believing in us and providing us with financial support and hot meals when the going got tough.

Last but not least, we would like to thank all the participants for their help and input.

Contents

List of Figures

List of Algorithms

List of Tables

Chapter 1

Introduction

Ever since the 1970s, when computer games evolved into an industry, they have greatly influenced Artificial Intelligence (AI) as a research field. Game AI has come a long way since it took its first wobbling steps, and players have grown to expect more and more from the computer controlled opponents in a game. A computer controlled opponent is expected to make good choices, but not too good; the user should eventually be able to beat it, while at the same time appear *human like*. *User experience* is an important aspect of game development, and techniques in Artificial Intelligence have been proved to increase the agent performance and have the potential to affect user experience in games.

In their master project, Helgesen and Krzywinski implemented an electronic version of the multiplayer strategic board game Diplomacy. They named the game StateCraft. In StateCraft autonomous intelligent agents played as an opponent to human players. Improvements and further development have been carried out after Helgesen and Krzywinski. For example, Jensen and Nes (2008) implemented a Personality module for the agent and studied the effect of personality on the agent performance and player experience.

This master project is a further extension of StateCraft which aims at improving the performance and user experience. We implemented two different modules in the autonomous agent of the game. The modules are the Emotion module and the Prisoner's Dilemma module.

1.1 The Emotion module

The Emotion module aims to improve the user experience of the game by giving the agent internal emotional states based on an emotion model. The emotional states affect the decisions of the agent, and hopefully make it appear more human-like.

Emotions have shown to be an important part of the human intelligence, and that they play an important role in decision-making for human beings. In this thesis, we wish to study whether an agent equipped with emotions will enhance the user experience. Of course, many people are sceptical and believe agents with "feelings" to be a futuristic dream, and maybe they are right. The most complex artificial systems still lag behind the complexity of an animal such as a squirrel or a nest-building bird (Sloman, 1998). Some doubters might be driven by the fear of computers taking over the world, like in sci-fi novels, while others might just dislike the thought of machines having these abilities that we as humans value so highly. In this thesis, the focus is not on making computers able to feel consciousness, but rather on simulating emotions in an agent so that it *appears* more human-like, with the goal of increasing the player's game experience.

Earlier research has shown that emotions can be successfully implemented in agents. The results indicates that emotions can improve both the performance of an agent (Maria and Zitar, 2007), as well as the believability of the agent, where believability refers to the agent providing the illusion of life (Bates, 1994).

To decide how to design and implement the Emotion module, a player study was conducted. Seven players were gathered to play the board game, and four of them were interviewed about their emotions afterwards. The OCC-model (Ortony et al., 1988) combined with the information collected in the player study formed the foundation for the Emotion module.

To evaluate the Emotion module, we have defined the following research question:

RQ How does emotions affect the performance of the agent and player experience in StateCraft?

To answer the research question, we ran a total of 310 data simulations of the game from 1901 to 1911, and compared the performance of the agent with and without emotions, using statistical analysis. Additionally, we performed a series of user evaluations, where six participants were asked to play three games. In the first game they were asked to identify the agents' emotions. Then the participants played two more games, one against emotional agents and one against regular agents. Afterwards, they were asked to state their opinions on which game they preferred.

The intention of this module is to study whether or not giving the agent emotions will enhance the user experience; improve the performance of the agent.

1.2 The Prisoner's Dilemma module

The Prisoner's Dilemma (PD) module is implemented to give the agent an understanding of the classical game theory problem of the Prisoner's Dilemma. StateCraft contains

several situations similar to the ones expressed in the Prisoner's Dilemma, and because of this, we wish to study whether the agent will perform differently if it understands the concept of the Prisoner's Dilemma.

StateCraft in all its complexity contains several situations akin to those present in the Prisoner's Dilemma. The Prisoner's Dilemma is a non-zero-sum game used to describe the problem of cooperation versus defection among rational actors in an environment where the actors will not necessarily gain directly from cooperating with each other. In StateCraft the players are competing for world domination, where supply centers[1] are the resources all players are fighting over.

Earlier research has resulted in several strategies for maximizing the outcome of the Prisoner's Dilemma. In this thesis we have implemented two of these strategies, where one, Tit-For-Tat, has proven to be successful in a two-player Iterated Prisoner's Dilemma game[2] (Axelrod, 2006), while the other, Freidman, has shown great potential in a three-player Iterated Prisoner's Dilemma game (Matsushima and Ikegami, 1998). Consequently, we wished to study the effect of these strategies in n-player game such as StateCraft.

To evaluate the Prisoner's Dilemma module, we have defined the following research question:

RQ How does the PD strategies affect the performance of the agent and player experience in StateCraft?

There are several aspects of performance for the Prisoner's Dilemma module. The number of supply centers the agent is able to gain is maybe the most obvious, and can be interpreted as a direct indicator of the agent's performance in the game. Another aspect important for the Prisoner's Dilemma is the *score*, a quantitative unit of measurement used to evaluate how good a player performs in the Prisoner's Dilemma. The score will be further defined in chapter 2. Last but not least, there is the human interpretation of agent performance, that is, how users think the agent performs, and why.

To shed some light on these aspects, we have performed qualitative data collections by letting users play the game with and without the Prisoner's Dilemma module, which we then use to explore whether or not the Prisoner's Dilemma have any effect on the user experience of the game. We have also collected quantitative data from StateCraft by running 430 simulations of the game from 1901 to 1911. This data will be subject to statistical analysis with the goal of exploring whether the Prisoner's Dilemma module has any effect on the game in terms of supply centers and score.

[1] Supply centers are explained in detail in Section 3.2

[2] An Iterated Prisoner's Dilemma game is a Prisoner's Dilemma game with several iterations. The players will fight each other N times, where $N > 1$.

The intention of this module is to study whether or not giving the agent these capabilities will enhance the user experience; improve the performance of the agent; and last but not least, to see if classical strategies used for the two-player Prisoner's Dilemma will work in the n-player domain of StateCraft.

1.3 Organization of the thesis

The thesis is organized as follows: Chapter two presents a theoretical background for both emotions in Artificial Intelligence (AI) and the Prisoner's Dilemma, used as a basis for the development and evaluation of the modules. Chapter three contains general information about the board game Diplomacy and StateCraft, as well as information about lower-layer improvements we have made to StateCraft to facilitate the development of the Emotion and Prisoner's Dilemma modules. Chapter four gives a detailed walk-through of the design and implementation of the two modules, while the two modules are evaluated in chapter five and six. Chapter seven presents the conclusion of the thesis, as well as proposals for future work.

Chapter 2

Literature review

2.1 About this chapter

In this chapter we present artificial intelligence, emotions and different ways of representing them, as well as earlier research combining emotions and artificial intelligence. In addition to this, we present the Prisoner's Dilemma, discuss the difference between 2-player, 3-player and n-player Prisoner's Dilemma, and give an overview of different strategies used for the Prisoner's Dilemma.

2.2 Artificial intelligence

Most people have an idea of what artificial intelligence (AI) is, mainly because of the influence from popular culture. Movies such as "2001: A Space Odyssey" (1968), "Star Wars" (1977), "WarGames" (1983), "Terminator" (1984), "Artificial Intelligence: AI" (2001) and "I, Robot" (2004) all contain some sort of entities with artificial intelligence. The common denominator for these movies is that they all present complex entities with a vast intelligence, often in the form of humanoids (e.g. Terminator and I, Robot), and they all possess what can be characterized as a personality and, to some extent, emotions. Such movies might have created a public perception of AI that does not correspond with the harsh realities of AI research.

Artificial intelligence is quite new as a research field, but myths and stories tell us that mankind has been fascinated by the concept of autonomous "machines" for thousands of years. According to Greek myths, the God of Hephaestus made *Talos*, a man made of bronze. Talos was to patrol the beaches of Crete, protecting *Europa* from pirates and invaders (Russell and Norvig, 2003, p. 939). Looking past myths and mythology, Ktesibios

of Alexandria built the first known self-controlling machine about 250 years B.C. (Russell and Norvig, 2003, p. 15). This was a water clock, known as a clepsydra, designed to control the water flow at a constant pace by utilising a regulator. This represented a paradigm shift from looking at objects as static artifacts, to looking at them as dynamic entities able to modify their behaviour based on their environment.

However, that an artifact is able to respond to the environment is not necessarily enough for it to be characterised as an intelligent machine. Philosophers early assumed that to create intelligent machines, one first had to understand and identify human intelligence. Beginning as early as Ancient Greece, philosophers, such as Aristotle, have tried to describe human thinking as the mechanical manipulation of symbols (Russell and Norvig, 2003, p. 6). This way of thinking, combined with the idea of self-controlling machines, laid the foundation for the modern programmable computer and artificial intelligence.

Inspired by the programmable computer, artificial intelligence was founded as a research field at a conference at Dartmouth College in 1956 (Russell and Norvig, 2003, p. 17). The interest in AI rapidly grew and people quickly gained high expectations for the field.

There are numerous definitions of AI. McCarthy (2007), who originally coined the term artificial intelligence, defines it as "*the science and engineering of making intelligent machines*". Luger (2004, p. 1) describes it quite similarly as "[...] *the branch of computer science that is concerned with the automation of intelligent behavior*".

Due to the numerous applications of AI, it has over the years evolved into several sub-disciplines, each with a different focus or goal. Luger (2004, p. 20-28) lists "*Game Playing, Automated Reasoning and Theorem Proving, Expert Systems, Natural Language Understanding and Semantic Modelling, Modelling Human Performance, Planning and Robotics, and Machine Learning*" as examples of such disciplines. Game playing is the most relevant sub-discipline for this thesis, and will therefore be the main focus of this chapter.

2.3 AI in Games

Game AI took its first wobbling steps in the early 1950's, when Christopher Strachey wrote a checkers program for the Ferranti Mark I computer, at the Manchester Computing Machine Laboratory. By 1952, Strachey claimed that the program could "*play a complete game of draughts at a reasonable speed*" (Copeland, 2000). During the 1950's and 1960's, a few researchers attempted to create video games with or without AI, but games as an industry first developed in the 1970's.

In 1971, Nutting Associates released Computer Space, the worlds first commercially available arcade game. Although it is questionable whether the player's computer controlled opponent was intelligent enough to fit the description of "artificial intelligence", the commercialisation of video games created a new incentive for people to research game AI.

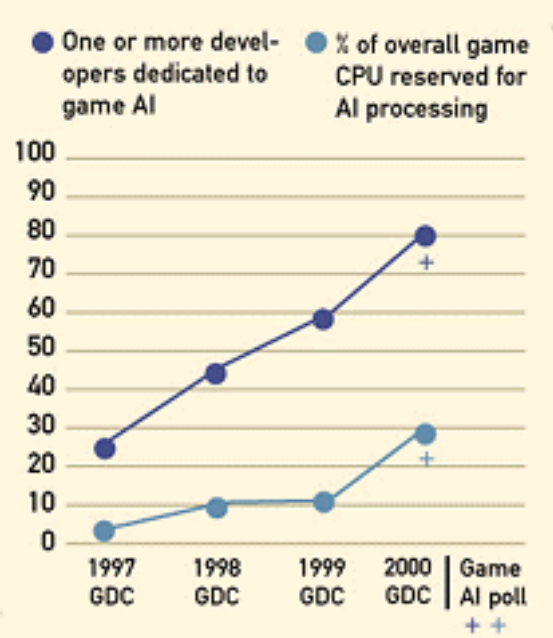


Figure 2.1: AI in computer games
(Woodcock, 2000)

By 1972, Pong was released by Atari. Primitive as it may have been, it was the first game gaining widespread success. The 70's and 80's created such titles as Space Invaders, Pacman, Zork, King's Quest and Donkey Kong.

Today, computer games generate big revenues. In 2001, the US computer game industry had a business volume larger than that of the US film industry (Kleiner, 2005). Artificial intelligence is becoming more and more important for video games, and human players are expecting the opponent to perform in a human-like manner. As a result of this, we can observe that the CPU resources used to process AI have grown considerably since the beginning, and the number of projects where one or more developers are dedicated to working on AI has increased significantly (see Figure2.1). Kleiner (2005) argues that the development in AI needs seen in later years might grow to be so large that it will enforce the development of specialised hardware for AI, as we have seen with for instance graphics accelerators or even physics accelerators. However, Tozour (2008) argue that such hardware will never surface, among other reasons because the problems faced when developing AI for games are too varied to make good use of such hardware.

In 1997, the chess-playing computer Deep Blue beat the reigning world champion Garry Kasparov in a game of Chess. What was remarkable about this is that Deep Blue derived most of its power by using brute force algorithms. This demonstrates how a computer,

albeit a supercomputer,[1] was able to beat the leading player at the time using a data driven approach to AI, analysing 10-12 levels ahead, calculating all possible moves, something which amounts to an incredible 1 000 000 000 000 000 000 different positions (Newborn, 2003).

In other board games like Checkers, Scrabble and Othello, computers also make some of the best players in the world. What's similar for all these games is that they are board games, with a finite number of potential moves in a round. This makes it theoretically possible to calculate all the possible moves with raw computational power, and for simple games like Tic-Tac-Toe, this has already happened.

2.4 Emotions

In this section, we will discuss what emotions are, if there are any basic emotions and why one would want to create emotional agents. We will also look at two models for emotion synthesis and other implementations of emotional agents. Last, we will discuss how emotions affect decision-making and behaviour.

2.4.1 What are emotions?

What is an emotion? People use emotions to describe their internal states and a person can easily recognise emotions like *sadness*, but that does not mean that they know how it works in detail. It is difficult to define an emotion, and Sloman (1993) suggested that we must wait for deeper theories about the underlying mechanisms before we can hope to come up with a precise definition of the phenomena, just like we for example had to gain greater understanding about chemistry before we could form a good definition salt.

Kleinginna and Kleinginna (1981, p. 355) reviewed over 100 different definitions of emotions and proposed the following:

> *Emotion is a complex set of interactions among subjective and objective factors, mediated by neural˜hormonal systems, which can (a) give rise to affective experiences such as feelings of arousal, pleasure/displeasure; (b) generate cognitive processes such as emotionally relevant perceptual effects, appraisals, labeling processes; (c) activate widespread physiological adjustments to the arousing conditions; and (d) lead to behavior that is often, but not always, expressive, goaldirected, and adaptive.*

[1] At the time, the worlds 259th most powerful supercomputer

Their definition might be too wide for our use, so when we speak of emotions in this thesis, we refer to the definition from Ortony et al. (1988, p. 13):

> [...] *valenced reactions to events, agents, or objects, with their particular nature being determined by the way in which the eliciting situation is construed.*

2.4.2 Basic emotions

Most emotion theorists agree that some emotions are more basic than others, often called primary or fundamental emotions. In colour vision, which many theorists use as a metaphor for emotions, it was a great help to come up with the three basic irreducible colours; red, blue and green. Emotion theorists also try to find these basic, irreducible emotions. However, none of them seem to agree on which emotions are basic, why they are basic or how many basic emotions there are. Mowrer (1960) claims that the only two basic emotions are pleasure and pain, while other theorists, such as Frijda (1987) claim that there are as many as 18 basic emotions.

Some claim that, in order to be referred to as a basic emotion, an emotion should have its own distinct facial expression across cultures. However, there are many things we do not consider emotions that has its own facial expression across cultures (e.g. lifting a heavy object). Also, an extremely intense feeling of joy might lead to crying, which also would be the facial expression associated with sadness. Does this mean that sadness should not be considered an emotion? Ortony and Turner (1990) argue that perhaps the only basic emotions are those that are experienced by both humans and animals. For example, it is easy to see that fear and anger are being experienced by a monkey or dog, but how is it possible to know if a monkey or dog experiences emotions such as envy or shame? One could argue that fear and anger are *more basic* since it is more plausible that animals also experience these emotions.

The most frequently occurring basic emotions among emotion theorists are anger, happiness, sadness and fear. However, these emotions also seem to be the emotions that are most frequently referred to in Western culture (Conway and Bekerian, 1987), and this might bias some theorists into giving these emotions a special status. There is also a possibility that a phenomenon such as basic emotions does not exist. Ortony and Turner (1990, p. 329), after a long review on basic emotions, concluded that "[...] *the study of emotions is no more dependent on the existence of a nontrivial subset of basic emotions in terms of which all other emotions can be explained than is the study of language dependent on the existence of a small subset of elemental languages, or the study of animals on a set of basic animals*".

2.4.3 Why emotional agents?

Emotions have shown to be an important part of human intelligence which plays an important role in decision-making for most humans. Even though emotions and rationality/reasoning might seem contradictory, neurological evidence show that they are more connected than first thought (Damasio, 1994). A study performed on a patient called 'Elliot', who had a brain damage that made him unable of experiencing normal human emotions, shed new light on the link between emotion and decision-making. One would think that this brain damage would make him operate highly rational, but the study showed that he had difficulties performing rational tasks like getting dressed in the morning (Damasio, 1994). Damasio (1994) suggested that Elliot's cold-blooded reasoning might have prevented him from assigning values to the different options, which made his decision-making landscape hopelessly flat. In other words, emotion can be considered an important part of human intelligence:

> "*The question is not whether intelligent machines can have any emotions, but whether machines can be intelligent without any emotions*" (Minsky, 1988)

Many researchers think that emotions are part of the reason why humans are excellent at making decisions on small data sets in environments with much uncertainty, often called "acting on a gut-feeling". By assuming this, one could argue that emotions could be useful from an engineering perspective, with the main focus on increasing system performance. Some implementations have shown that emotions can help agents make better decisions, for example Maria and Zitar (2007) who showed that an agent with emotions performed better than a regular agent in a benchmark problem. Their research indicated that emotions can be performance enhancing if used correctly.

From a scientific perspective, implementations of emotional agents can be used as a test bed for theories about natural emotions in animals and humans. Emotional agents can also be used in the field of Human-Computer-Interaction as a tool to reason about how the user feels and thereby improve usability and acceptance, e.g. a personal agent should not ask the user the same question many times if it feels that it annoys the user. But most relevant to this thesis, emotional agents can be used in computer games, to create a more human-like opponent, making it more fun to play against.

Animators have known for a long time that one of the keys to create a believable character is to give the character the ability to express emotion in a clear manner with appropriate timing. Two of Disney's core animators, Thomas and Johnson, stated that: "*From the earliest days, it has been the portrayal of emotions that has given the Disney characters the illusion of life*" (Thomas and Johnson in Bates, 1994, p. 2).

AI researchers, on the other hand, have been focusing more on reasoning, problem solving and other characteristics which we associate with intelligence, on their path to solve

the riddle of human-like agents. The reason for this might be that such characteristics are highly valued by scientists and research communities. Bates (1994) argues that while scientists may have been able to recreate the scientist, animation artists have come closest to understanding and capturing the essence of humanity that many AI researchers seek.

2.4.4 Synthesising emotions

For an agent to "have" emotions, it needs the ability to synthesise or generate them. There are several theories on how emotions are generated, but the OCC-model (Ortony et al., 1988) and the three-layered architecture by Sloman (1993) were considered most relevant since they were made with computation in mind.

2.4.4.1 The Ortony Clore Collins-model (OCC)

The OCC model (Ortony et al., 1988) focuses on what contribution cognition makes to emotion, and devotes less time to other important aspects such as facial expressions or behavioural components. Nor does it focus on how different emotions interact with each other. To this day, it is one of the most popular models for synthesising emotions, despite the fact that it was never intended for that purpose. Ortony et al. (1988) thought it was important for a machine to reason about emotions, not to have them. They did not try to describe every different emotion, but work at a higher level, called *emotional clusters* or *emotional types*. For example, the emotion joy in the OCC-model also includes emotions like happiness, cheerfulness and bliss.

The OCC-model puts emotions into 22 categories, as seen in Figure 2.2, depending on if they are appraisals to events, actions of agents or objects, giving these emotions different emotion words. Ortony et al. (1988, p. 21) specify that these words are just chosen as suggestive labels for entries for that position in the structure and that their structure are not intended to be used to define these words.

In the OCC-model, emotion was described as a valenced reaction to an event, agent or an object. Valenced means that the emotion has to get a positive or negative reaction, excluding neutral emotions such as *surprise*. The emotion's particular nature is being determined by the construal of the eliciting situation. For example, take two people watching a soccer game when one of the teams scores. The first person cheers for the scoring team, while the other person cheers for the other team. The first person will probably feel *joy*, since his team just scored. However, the other person may feel *distress*, since he was unappeased with the event. The event is the exact same for both persons, but their construals of the event is different (Ortony et al., 1988, p. 4).

Events are considered things that happen, and the agent's reaction depends on his goals. Given that the main goal of the agent in StateCraft is to gain 18 provinces, losing a

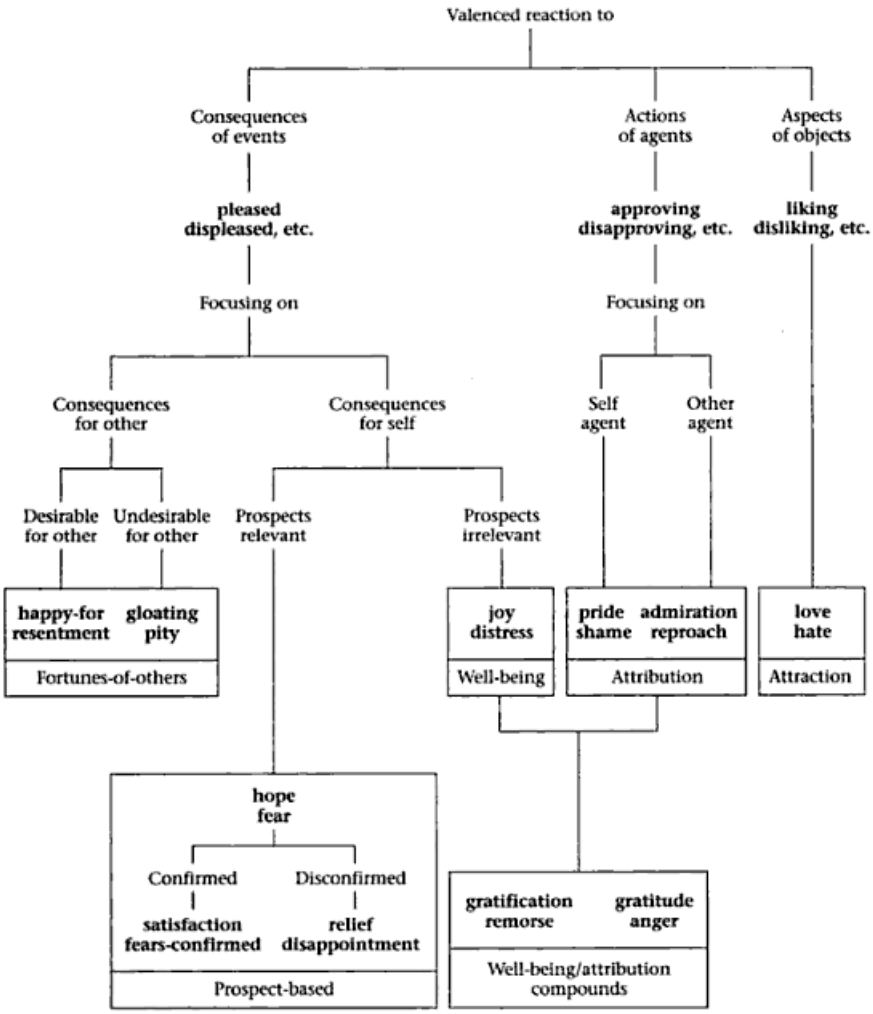


Figure 2.2: The original OCC-model
(Picard, 2000)

province would be displeasing to him, and thereby cause him distress. The prospect of future events can give rise to the prospect-based emotions *hope* and *fear*, and the intensity also depends on the goals. If the agent thought it was likely that a neighbour would attack it and conquer its provinces (i.e. a displeasing event), he might experience *fear*. The intensity of the emotion also depends on the likelihood of him attacking.

Actions of agents can be approved or disapproved depending on the agent's set of standards. If it is another agent who performs the action, it can give rise to the emotions admiration and reproach. If it is the agent itself who carries out the action, the emotion *pride* or *shame* might occur. Standards represent both moral and beliefs, for example "One shall not lie to other people". When an agent lies to another player (e.g. about supporting someone in an attack on Burgundy) it will disapprove of its own action, and therefore feel *shame* if the intensity of the emotion is strong enough.

Objects can be disliked or liked based on the agent's attitudes towards the object. An object can also be another agent. An attitude can be "I don't like people who wear gold chains". The intensity of these emotions are mostly based on the grade of "dislike" or "like" towards the object.

Last but not least, there are compound emotions consisting of two other emotions. For example an agent will feel *reproach* towards another agent who attacked him, and *distress* since he lost a province. This combination will lead to *anger* towards the other agent, which is stronger in intensity than the two others and thereby will overwhelm them.

Let's look at a simple example of how the OCC-model's implementable rules can easily be used to synthesise the emotion *joy*.

```
Rule 1:
IF DESIRE (p, e, t) > 0
    THEN set JOY-POTENTIAL (p, e, t) =
    f_j(| DESIRE (p, e, t) |, I_g (p, e, t))
```

DESIRE (p, e, t) is a function that returns the desirability that a person p assigns to some event e at a time t. If it returns a positive number, it is a desirable event for the person p, and therefore will cause *joy*. However, in the case of an undesirable event, it will return a negative number and might also cause *distress*. I_g is a function that returns the combined effects of all the global intensity variables. $F_j$is a function specific to *joy*, that takes the desirability and the effects of global intensity as parameters (Ortony et al., 1988, p.183).

A crucial component that is presented in the OCC-model is that an emotion's intensity has to be above a certain threshold value. If the value is below the threshold value, the emotion will not be experienced by the person. Therefore we will need rules such as Rule 2:

```
Rule 2:
IF JOY-POTENTIAL (p, e, t) > JOY-THRESHOLD (p, t)
    THEN set JOY-INTENSITY (p, e, t) =
        JOY-POTENTIAL (p, e, t) - JOY-THRESHOLD (p, t)
    ELSE set JOY-INTENSITY (p, e, t) = 0
```

Example: Frank is watching a funny movie, but he is not feeling joyful. The setting may imply that he should be feeling joyful, but the JOY-POTENTIAL does not exceed the JOY-THRESHOLD, meaning that he feels no joy.

2.4.4.2 Three-Layered Architecture

In 1998, Aaron Sloman proposed a three-layer architecture for human emotions, where the layers can be categorised by their evolutionary age, from oldest to newest. The first layer consists of purely reactive emotions, also known as primary emotions, and is considered the oldest layer. These emotions are found in all animals, even insects. However, animals with only a reactive layer will probably have a very predictable behaviour, for example running every time it sees light. Emotions found in this layer are *terrified* and *disgust* (Sloman, 1998).

The second layer consists of the more recently evolved deliberative emotions found in some animals. Emotions such as *relief*, *apprehension* and *anxiousness* belong to this layer. The deliberative layer performs tasks such as planning, decision-making, success detection and resource allocation. An example can be a person who discovers he passed his driver's license. After being nervous for some time, he will detect his success and most likely transition into the feeling of *relief* (Sloman, 1998).

The third layer is called the meta-management or self-monitoring layer and is the most recently evolved layer. It is likely that this layer only exists in primates, and very young human infants do not seem to have it fully developed. Examples of emotions found in this layer are *shame*, *grief*, *infatuation* and *embarassment*. One of the interesting points about this layer is perturbance, meaning that an emotion makes one lose control over ones thoughts, and thoughts unwillingly interrupt one's attention. For example, if a father has lost his son in a car accident, the *grief* will probably be so intense that he will have problems focusing his mind on anything else (Sloman, 1998).

This architecture lacks implementation details, but sheds light on the need for a layered architecture, which includes low-level primitive emotions as well as high-level cognitive ones. But above all, it shows that there is a need for a layer for meta-management or self-control, so that the system is able to manage its emotions. According to Picard (2000), this is not only important, but also necessary for a system whose purpose is to develop the skills of emotional intelligence for regulating and using its emotions wisely.

2.4.5 Emotion's affect on behaviour and decision-making

When the emotions have been generated internally in the agent, it needs to act on its emotions. Since facial expressions are not relevant to this thesis, we will focus on how emotions influence behaviour and decision-making.

Emotion plays two interesting roles in decision-making: expected emotion and immediate emotion. *Expected emotion* consists of the predictions about emotional consequences of decisions. For example, a person might think through all the different possible outcomes and assign an expected emotion to each outcome, based on prior experiences. *Immediate emotions* are the emotions felt at the time of the decision-making. "*Immediate emotions can influence decisions indirectly by altering the decision maker's perceptions of probabilities or outcomes or by altering the quality and quantity of processing of decision-relevant cues*" (Loewenstein and Lerner, 2003, p. 620).

Several studies have been performed on the effect on decision-making of specific emotions such as anger, guilt and fear. Lerner and Keltner (2001) found that angry and happy decision-makers make optimistic risk estimates and more risk-seeking choices, while fearful people express more pessimistic risk-estimates and tend to make the safe choice. Another well documented effect of anger is aggression (Berkowitz, 1990), but studies have shown that it is hard to focus it towards the person one is angry at (Lerner and Tiedens, 2006). On the other hand, Harmon-Jones et al. (2003) found that anger makes people more eager to act if there is something that can be done to resolve the negative situation. Freedman et al. (1967) discovered that people who are feeling guilty are easier to manipulate into complying, at least when they have the possibility of avoiding a confrontation with the one towards which they are feeling guilty.

2.4.6 Implementations of emotional agents

In this subsection, we will look at some implementations of emotional agents and how the emotions were used to affect facial expressions and decisions.

2.4.6.1 The Orphanage-Care project

Maria and Zitar (2007) used a simplified version of the OCC-model to create an emotional agent. The emotions are generated by a symbolic approach which utilises a rule-based system with interactions with the environment and an internal "thinking" mechanism. They also made an agent without emotions, and compared the two agents by letting them solve a benchmark problem called the "The Orphanage Care Project". The purpose was to show that systems with a proper model of emotions can, in many cases, perform better than the same system without the emotion model, and their test results showed that this was the case.

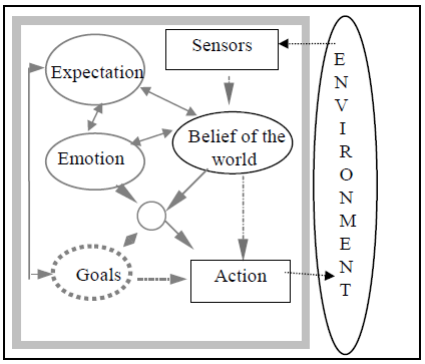


Figure 2.3: An Extended Intelligent Agent Framework
(El-Nasr and Skubic, 1998)

2.4.6.2 A Fuzzy Emotional Mobile Robot

El-Nasr and Skubic (1998) researched the utility value of emotions for decision-making in a mobile robot. They used a fuzzy logic model to represent the intensity of the three emotions; anger, fear and pain. The reason for choosing a fuzzy logic model is that an emotion is not either on or off, it can vary in intensity. For example one can feel *mild anger* or one can feel *wild rage*. The framework in use is an expansion of the Intelligent Agent (IA) framework presented by Russell and Norvig (2003).

As shown in Figure 2.3, expectations are important for calculating the intensity of the emotions. If a person believes he will receive an A on his test, but receives an F, he will probably feel sad and disappointed. If he feels that the teacher is to blame, he may also feel anger towards the teacher. The person's goals also affect the emotion generation. If his goal was to get a bad grade, he would not feel the same emotions. Another interesting aspect of their algorithm is that emotions are suppressed if their intensity is below a certain threshold. The threshold depends on for example the number of people nearby. If the robot is surrounded by many people, it will suppress feelings of mild anger.

2.4.6.3 The Oz project

Bates (1994) and his research partner Scott Neal Reilly were interested in making believable agents, so they created a fictional world, called "The Oz Project". It consists of several fictional characters that do not look like any real creatures, but are built to influence their audience as they were. The key to making believable characters were, inspired by Disney animators, to create characters with clearly defined emotional states. The characters generate emotions by using cognitive appraisals to the world with a system called Em. Em is based on the OCC model, with two extra emotions: *startle* and *frustration*. Each emotion has an intensity threshold, which the intensity has to exceed in

order to affect the characters outward behaviour. Each emotion's intensity also descends towards zero as time goes by. The mood of the character is defined as the sum of all negative and positive emotions. If the intensity of the positive emotions is larger than the intensity of all negative emotions, the character is in a good mood, and vice versa. In addition, the creatures are equipped with a personality, which also affect their outward behaviour.

2.4.6.4 Poker-playing agent with facial expressions

Koda and Maes (1996) developed a personified poker-playing agent with facial expressions. The agent uses a subset of the emotions presented in the OCC model, extended with the emotion *surprise*, for emotion synthesis. These emotions are then used to affect the facial expressions of the agent, for example, if the agent feels *sad*, they will show the image where the player has a sad facial expression. They concluded that people found the poker agents with facial expressions more likable, engaging and comfortable than the ones without. However, the users did not perceive the agents with facial expressions as more intelligent than those without. To be perceived as intelligent, it seems that the agent's actual skill at poker is the major factor.

2.5 The Prisoner's Dilemma

2.5.1 What is the Prisoner's dilemma?

Prisoner's Dilemma (PD) is a well-known problem in the field of game theory. It's a non-zero-sum game used to describe the problem of cooperation vs. defection among rational actors in an environment where the actors will not necessarily gain directly from cooperating with each other. Axelrod (2006, p. 9) describes it as an "*abstract formulation of some very common and very interesting situations in which what is best for each person individually leads to mutual defections, whereas everyone would have been better off with mutual cooperation*". It is defined as follows:

> *In the Prisoner's Dilemma game, two individuals can each either cooperate or defect. The payoff to a player is in terms of the effect on its fitness (survival and fecundity). No matter what the other does, the selfish choice of defection yields a higher payoff than cooperation. But if both defect, both do worse than if both had cooperated (Axelrod and Hamilton, 1981, p. 1391).*

As the name implies, PD is formalised as a scenario set to a prison of some sort. Two prisoners (the *players*) are arrested for some crime, e.g. a bank robbery, and put into

Table 2.1: Prisoner's Dilemma payoff matrix

	Cooperate	Defect
Cooperate	R = 3, R = 3	S = 0, T = 5
Defect	T = 5, S = 0	P = 1, P = 1

Where T stands for Temptation to defect, R for Reward for mutual cooperation, P for Punishment for mutual defection and S for Sucker's payoff. To be defined as a Prisoner's Dilemma, the following inequalities must hold: $T > R > P > S$.

interrogation rooms. The police do not have sufficient evidence to tie the prisoners to the robbery, but strongly suspect both prisoners to be guilty. What they do have is enough evidence to send the prisoners to jail for a short period of time for some other crime (this is, after all, rather ruthless criminals), for example petty theft, i.e. a lesser crime. The police do not want to jail them for this lesser crime, instead, they want to jail them for the big crime. So they try to get the prisoners to talk, stating that if one of the prisoners P1 give enough information to jail the other prisoners P2, P1 will go free, while P2 will receive a maximum sentence. If both the prisoners give information about each other, they will both receive a medium sentence. If none of the prisoners give any information to the police, they will both receive the minimum sentence for whatever other crime they have committed (the petty theft).

The act of talking or giving information about the other prisoner to the police is defined as **defecting**, while the act of remaining silent is defined as **cooperating** (with the other prisoner). None of the prisoners care about their accomplice, they only care about their own personal gain, and as such the choice of defecting by testifying against their accomplice is tempting. However, they risk serving a very long sentence if they both testify against each other, but - not as long as the sentence the loosing part serves if only one of them testifies against the other. Axelrod (2006, p. 9) argues that the best choice for a player P1 is to defect if he thinks the other player P2 will cooperate, and that the best choice for a player P1 is to defect if he thinks the other player P2 will defect. So, the best choice for a player P1 is to defect if P2 will cooperate **and** if P2 will defect. However, this is true for P2 as well, so P2 should defect no matter what the first player does. The dilemma is derived from the fact that if both of them defects, they will both receive the medium sentence, which is considerably worse than the minimum sentence they could receive if they both cooperated!

See Table 2.1 for a more structured, weighted description of the dilemma.

The Prisoner's Dilemma is not meant to be interpreted as a dilemma between two prisoners. The dilemma they face, albeit most likely very important for the prisoners, probably

isn't very interesting from a scientific point of view[2]. Axelrod applies the dilemma to such cases as the "live-and-let-live" system that emerged during World War I, where the two fighting forces (the Allies and the Central Powers, respectively) had soldiers stationed in trenches along the Western Front for long periods of time. During the course of the war, the soldiers of both sides developed a system of mutual cooperation, where the soldiers on the allied side would refrain from inflicting damage to the soldiers fighting for the central powers, and vice versa, the soldiers fighting for the central powers would be hesitant to administer what they considered needless violence to the allied soldiers (Axelrod, 2006, p. 73-74).

> *The real reason for the quietness of some sections of the line was that neither side had any intention of advancing in that particular district... If the British shelled the Germans, the Germans replied, and the damage was equal: if the Germans bombed an advanced piece of trench and killed five Englishmen, an answering fusillade killed five Germans.* (Cobb in Axelrod, 2006, p. 76)

The Prisoner's Dilemma can be identified in several other aspects of life, from politics, biological systems, economics and so on.

2.5.2 The Iterated Prisoner's Dilemma

In the classic Prisoner's Dilemma the players only play one round. In the Iterated Prisoner's Dilemma (IPD) the game is played repeatedly, with the number of rounds being either a fixed number, an infinite number or a random number. This makes it possible for players to keep a history of how their opponent plays, and introduce the opportunity to punish the player's opponent for not cooperating. Axelrod (2006, p. 12) argues that it is primarily under these conditions that cooperation will emerge, because "the choices made today not only determine the outcome of this move, but can also influence the later choices of the players". The fact that the players' paths can cross again (i.e. new round in IPD), both with a presumably clear memory of their opponent's action in the previous game, gives them an incentive to act cooperative towards their opponent.

The extension to an iterated game introduces the concept of *strategies*. In game theory and, therefore, the Prisoner's Dilemma, a strategy is a recipe a player follows to gain a goal, in this case the maximum score (or, to keep the thread to the metaphor of prison, the minimum penalty). A strategy specifies what the player should do in every case, so that if the opponent does A, the player should perform action B, or if the opponent does C, the player should perform action D. There are several strategies for the IPD, ranging from

[2]Criminologists might disagree.

the simple ALL-D strategy which always defects, to complex strategies using Bayesian inference to select the best choice for the long run (Axelrod, 2006, p. 14).

In the one round PD, defection is always the best strategy (Axelrod, 2006, p. 10). The same applies for IPD games where the game is played exactly N times. There is no incentive to cooperate on the very last move of an N-round game because the players will never meet again, and thus have no reason to fear that their actions will affect their score in later rounds. This is also true for the next-to-last round, as both players can anticipate defection from each other in the last round (i.e., the next round). If we apply this to every round from $N-1$ all the way back to the first round, we can observe that there isn't any reason for the players to cooperate at all in a IPD game with a fixed number of rounds. (Luce and Raiffa in Axelrod, 2006, p. 10)

In an IPD game played with an unknown or infinitive number of rounds, however, assuming that the players do not know how many iterations the game is composed of, defection isn't always the best move, and the players should rather cooperate than defect to avoid being punished by their opponents later in the game. However, if a player is playing against a player who always defect, the safest strategy is to defect.

Another variant of the Iterated Prisoner's Dilemma is the *Peace War Game*. In this game, the two decisions (i.e. defect, cooperate) are replaced with **Peace** and **War**. The game is used in academic circles to study the possible strategies for cooperation and aggression, but the metaphor is changed from being a prison-scenario to being a metaphor of countries' actions in the world, i.e. making peace or war.

2.5.3 Axelrod's Tournament

In 1980, Robert Axelrod held a tournament of various strategies for the Prisoner's dilemma. Game theorists were encouraged to submit strategies to the tournament, and the strategies would be tested in an Iterated Prisoner's Dilemma game. Strategies were submitted by game theorists from several different fields of science, and a total of 14 strategies were submitted. These 14 strategies were paired with each other plus a random strategy (i.e. a strategy choosing either cooperate or defect randomly) in a round robin tournament, with a game lenght of 200 moves (Axelrod and Hamilton, 1981). What Axelrod discovered in his tournament was that the greedy strategies tended to do poorly (i.e., the ones who defected), while the altruistic strategies performed noticeably better. The tournament showed that the overall best strategy was the simplest strategy submitted, the *Tit-For-Tat* strategy (TFT). TFT had the property of being nice, that is, it was never the first to defect, something which proved to be a key property of all the high-scoring strategies.

Some of the strategies submitted to Axelrod's Tournament will be discussed in the next section.

2.5.4 Strategies for Prisoner's Dilemma

2.5.4.1 2p IPD strategies

Always-Defect might be the simplest strategy for IPD. This strategy will always defect, no matter what its opponent does.

Always-Cooperate is competing with ALL-D for simplest strategy. This strategy will, as the name implies, always cooperate, no matter what its opponent does.

Tit-For-Tat is, as mentioned in section 2.5.3, the winning strategy from Axelrod's Tournament. It is composed of two parts:

1. Cooperate on the first move
2. On the next move, do whatever your opponent did on the previous move.

TFT, created by Professor Anatol Rapoport, is by far the most well known strategy for the IPD today. Even though it is a very simple strategy, it has a key property which separates it from the low-scoring strategies in the IPD: the property of never being the first to defect. That is, a player using the TFT will only defect once someone else have defected against the player first. Axelrod (2006) describes a strategy which never is the first to defect as being *nice*. TFT is also a very *robust* strategy, meaning a strategy which can grow or reproduce in an environment filled with other or similar strategies. It also has high *evolutionary stability* against the ALL-D strategy, where evolutionary stability refers to a dominant strategy's ability to hold it's position as a dominant strategy in an environment. This means that if you have an environment where all the players are playing TFT, and a mutant playing ALL-D shows up, the players using TFT will outperform ALL-D (Axelrod and Hamilton, 1981).

Friedman is another "nice" strategy for IPD, but of all the nice strategies submitted to Axelrod's Tournament, Friedman scored the worst. Friedman is a very unforgiving strategy. It starts out with a cooperative move, as it must to be considered nice, and then continues to cooperate as long as its opponent cooperates. If the opponent defects, however, Friedman will defect immediately, and continue to defect for the remainder of the game (Axelrod, 2006).

Downing is a rather complex strategy in the sense that it tries to mimic the actions of human subjects in a Prisoner's Dilemma laboratory experiments (Axelrod, 2006, p. 34). It tries to "understand the other player and then to make the choice that will yield the best long-term score based upon this understanding" (Axelrod, 2006, p. 34). It does this by estimating a probability of the other playing cooperating after it either cooperates or defects, and if the opponent seems unresponsive, it deliberately tries to maximise its own winning by defecting against the opponent

Axelrod (2006) categorises it as a "fairly sophisticated decision rule", but argues that its poor performance in Axelrod's Tournament was based on an implementation flaw that assumed that the other player initially would be unresponsive. This led Downing to defect on the first two moves, causing it to score poorly when the other players defected in revenge.

Joss is another interesting strategy which tries to increase its score by performing a random defection now and then. It is basically a modified TFT-strategy that randomly defects about 10% of the time. It performed poorly in Axelrod's Tournament, as any defection against TFT would cause TFT to defect in response, thus leading Joss to defect in response to TFT's defection, and an endless cycle of mutual defection was started (Axelrod, 2006).

2.5.4.2 3p IPD strategies

Just as there is a difference between a single-iteration Prisoner's Dilemma and a Iterated Prisoner's Dilemma, there is a difference between IPDs with two, three or more players. Therefore its necessary to examine these differences before we study strategies for the different game types. In a *2-player* IPD (2p IPD), there are always two and only two players facing each other in a given dilemma. Although there can be several players playing at the same time in the same environment, all battles are fought between two players. In a *3-player* IPD (3p IPD) there are always three players facing each other in a given dilemma (Matsushima and Ikegami, 1998). If we defined Diplomacy as a 3p IPD, all battles would be fought between three players at the time, but, given the rules of Diplomacy, this is not always the case. In Diplomacy, all battles can be fought between two and more players, where the maximum number of players are equal to the number of units there is room for adjacent to a province. As Diplomacy can be played with 7 players, we could say that the IPD aspect of Diplomacy is a 7p IPD, but the more general term N-player IPD is more fitting. In an N-player IPD, any number of players can play against each other for a given dilemma.

The problem with strategies such as Tit-For-Tat is that while they perform good in a two-player IPD setting, the performance drops if noise is introduced (i.e. noisy 2p IPD).

The same is true if the noise in the 2p IPD is replaced with a third player, making it a 3p IPD (Matsushima and Ikegami, 1998). Matsushima and Ikegami (1998, p. 54) writes the following about noise in the IPD:

> *The role of noise in the iterated prisoner|s dilemma game is to destabilise mutual cooperating states by creating a high level of distrust between the players. For example, two TFTs will alternatively defect each other. Because of the resulting distrust, TFT will be replaced by strategies which are more tolerant of defections, but at the same time more complex strategies will evolve.*

What we can see from this is that noise in a IPD makes the players distrust each other, and makes the possibility of the players making a wrong decision based on inaccurate input more probable. Matsushima and Ikegami (1998) states that many-person games are complex in the sense that the effectiveness of retaliations is not assured. Even if the players can recognise and punish the defector, because of simultaneous playing, they cannot be certain that they will not punish another player, or that their attack will punish the defector at all.

Neither Diplomacy nor StateCraft in their simplest forms are noisy[3], but both games have the aspect of being N-player games. This is, assuming that we look at the IPD-aspect of the games, somewhat equivalent to a noisy 2p IPD in the sense that the many players make the complexity of the game far greater, and that an attack on a player might affect several other players because several players can share the same dilemma provinces (e.g. player P1 tries to defect against P2 by attacking a dilemma province, but ends up attacking players P3 and P4 in addition to P2, because the attacked province is a dilemma for all four players).

Matsushima and Ikegami (1998) ran simulations using an evolutionary approach to find strategies which performed good in a 3p IPD. The strategies that proved to dominate the population of strategies in the simulations were named *Strategy-X* and *Strategy-Y*.

Strategy-X will start out cooperating. It will continue to cooperate if both the opponents either defect or cooperate (i.e. CC or DD), but will change to a defective state if one opponent defects and one opponent cooperates (i.e. CD, see arch marked CD from C to D in Figure 2.4). Once in the defective state, it will continue to defect until both opponents cooperate at the same time (i.e. CC), at which time it will return to the cooperative state.

[3]Although one could argue that the social aspect of the games could make them "noisy", in the sense that if a player attacks you, you cannot be sure that he is not performing the attack as a good-will favour to some other player

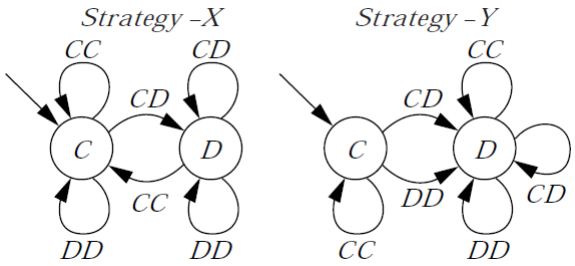


Diagram of Strategy X and Strategy Y
(Matsushima and Ikegami, 1998)

Figure 2.4: Strategy-X and Strategy-Y

Strategy-Y will start out cooperating, and will continue to do so as long as both its opponents cooperates. If one or both of his opponents defects (i.e. CD or DD), he will switch to a defective state, and will continue to defect no matter what his opponents does for the rest of the game (i.e., he will become ALL-D). Strategy-Y is 3p IPD version of the 2p IPD Friedman strategy (see section 2.5.4.1)

2.5.5 Prisoner's dilemma in StateCraft

In the case of StateCraft or, for that matter, Diplomacy, the Prisoner's Dilemma can be observed when two players want to occupy the same province at the same time. To exemplify, say we have two players, P1 and P2, and a very valuable province, L1. Both players occupy and have one or more units situated in provinces adjacent to L1, and both of them wants to occupy province L1. In Figure 2.5, we can see P1 as Russia, occupying among other provinces, Warsaw. P2 is played by Turkey, occupying Sevastopol, among other provinces. Moscow is the valuable province L1. Depending on the choices the players make, the situation have three different outcomes:

1. One of the players, P1, chooses to defect by moving to L1, while the other player, P2, chooses to cooperate with P1 by not moving for L1. The outcome is that P1 gain everything while his opponent, P2, gain nothing. This could, of course, have been played out the other way, with P1 cooperating and P2 defecting, but the outcome would have been the same. $P1 = 5$, $P2 = 0$.

2. Both of the players choose to defect by moving for L1[4]. None of them will gain control of the province, as only one player can occupy one province at a time, and none of the units have support from other units, thus resulting in a standoff where

[4]This is the outcome demonstrated in Figure 2.5

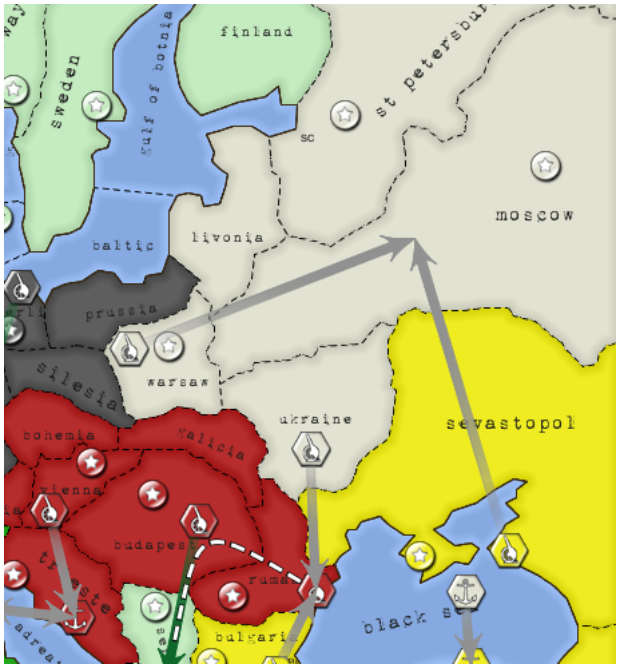


Figure 2.5: Prisoner's Dilemma in Diplomacy

none of the players are more powerful than the other. Both of the players will have to move back to the province from which they came, and both of the players will "loose", as they just threw away a perfectly good move by moving to a province at the same time as another player. $P1 = 1$, $P2 = 1$.

3. None of the players chooses to move to L1, which translates to cooperating in the Prisoner's Dilemma. They will both experience some gain, although not as great a gain as they would have experienced if they actually occupied the province. The reason that this is regarded as some gain for both of them is that their opponent didn't occupy the province, and because of this they are both free to occupy the province later on. In addition they (ideally) didn't waste their move by holding the unit's position but rather focused on occupying something else, or helping another unit or player. This roughly translates to a situation where none of the prisoner's chooses to defect (or both cooperates), and is in game theory regarded as the safest strategy. $P1 = 3$, $P2 = 3$.

Chapter 3

Diplomacy & StateCraft

3.1 About this chapter

This chapter contains information about the Diplomacy board game, its rules and history. It also includes information about StateCraft, the Java implementation of the board game used as the foundation of our thesis.

In addition to information about the original StateCraft implementation, it also contains information about the TacticTree, an addition to the tactical layer of StateCraft. The TacticTree is created by Christoph Carlson and Mathias Hellevang, with the goal of improving the performance of tactic generation by utilising a tree-based depth-first approach to generate tactics for the Caeneus Agent. The reason for implementing the TacticTree is based on a problem identified in the original StateCraft implementation. The problem occurred when the agents reached a certain number of units, making the number of legal tactics too big to be processed by an ordinary computer in an acceptable time frame.

3.2 Diplomacy

Diplomacy is a strategic social multiplayer board game that focuses on diplomacy as a tactic for reaching the goal state of world domination. Created in 1954 by Allan B. Calhamer, Diplomacy has been popular ever since.

The setting of Diplomacy is a Europe in war, more specific Europe during the First World War. Seven nations (Russia, The United Kingdom, France, Germany, Italy, The Ottoman Empire and Austria-Hungary), play for the sovereignty of Europe. The board is a map of Europe divided into 75 provinces, and each nation starts out with a specific

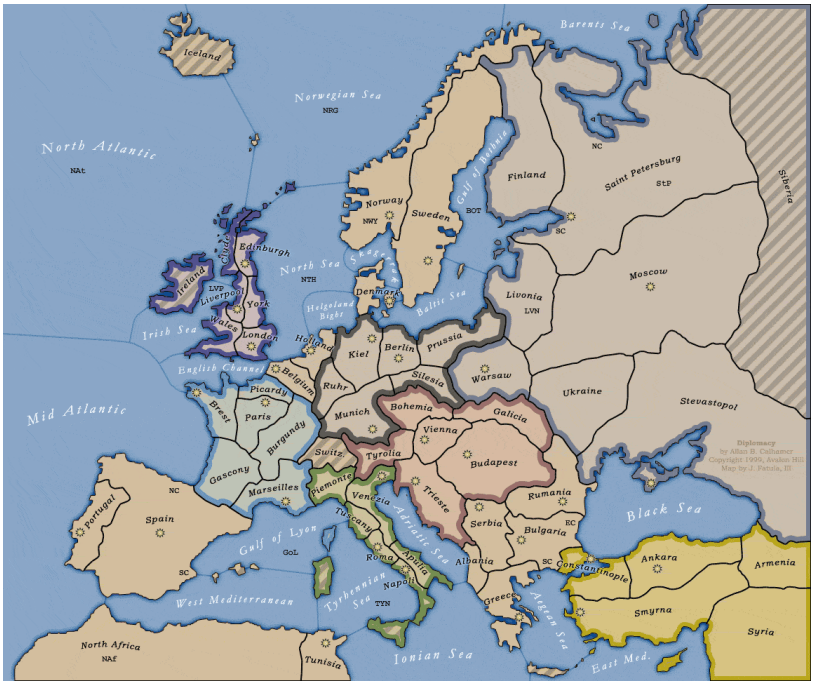


Figure 3.1: Map of Diplomacy

set of provinces (e.g., The United Kingdom has the British isles), which are considered the nation's home provinces. The provinces are land, open waters and coastal areas.

The players can build and commandeer units of two types: armies and fleets. Armies are land-based units and are used to occupy territory inland. Fleets are sea-based units used to occupy open waters and coastal areas. Both armies and fleets can occupy coastal areas. Armies can be given orders to move or hold position. Fleets can be ordered to move, hold position, or the fleet-specific order of convoying armies from one province to another province crossing a sea area. Both armies and fleets can assist other units, either the player's own units or the units of an opponent.

Some of the provinces have supply centres. The supply centres have the property of mapping 1:1 to a unit, so by occupying a new supply centre, the player is given the opportunity to build a new unit. The player cannot have more units than supply centres, so if he wants to build extra units he needs to occupy new supply centres. If the player loses a supply centre to an enemy after he has built a new unit, he cannot support his current units, and has to scale down his army by destroying one of his units. Only the provinces with supply centres in the player's starting area can build units. Provinces the player occupies during the game cannot be used to build new units.

Given that the supply centres are a requirement for expansion of the players' armies, they are necessarily an important aspect of the game. Additionally, the winning state of the

game can only be reached by controlling 18 supply centres.

A province can only hold one unit at a time. To occupy a new province, there are 5 different scenarios:

S1 Unit U1 moves into province P1 which is unoccupied, with no other attackers trying to occupy it. U1 occupies the province.

S2 Unit U1 moves into Province P1 which is occupied by unit U2. This leads to a standoff, and U1 has to retreat to the province from which he came. U2 keeps province P1.

S3 Unit U1 moves into Province P1, with support from unit U2. Province P1 is occupied by unit U3, but since U1 and U2 outnumber U3, U1 occupies the province while U3 has to retreat to an empty, adjacent province, or is disbanded.

S4 Unit U1 moves into Province P1, with support from unit U2. Province P1 is occupied by unit U3, with support from unit U4. This leads to a standoff like in S2, and U1 has to retreat to his province from which he came.

S5 Unit U1 moves into Province P1, which is unoccupied. Unit U2 also moves into P1 from another province. This leads to a standoff where none of the players outnumber each other, with the result that both U1 and U2 must retreat to the provinces from which they came.

As the name implies, social interaction makes up an essential part of Diplomacy. To win a player needs to control a majority (18) of the supply centres, a control that can be hard or even impossible to reach unless the player chooses to form alliances during the course of the game.

The game is divided into different rounds, where each round represents a season. Spring and autumn are known as *action rounds*, and gives the player the ability to perform actions affecting the position of the units (e.g. unit relocation, supporting other units and convoying other units). Spring and autumn are preceded with a *negotiation phase*, where the players negotiate and form alliances between each other. For example, before an action round, England might ask France to support an attack on a german province, while France might ask something of England in return, e.g. to "give" him a province France considers strategically important. The orders given by the players in the action rounds are interpreted in the summer or winter round, making the outcome of the preceding round visible for the players. This means that if a player promises another player to support him in some action, while in reality chooses to break his alliance and rather attack the one he promised to support, this will not be visible for the players until the following season.

Another aspect of Diplomacy is that the game's combat system has no element of randomness, and all randomness incorporated in the game is the randomness given by players when choosing a move. This makes it very interesting in a context of AI, as any randomness will have to be implemented in the autonomous agents.

3.3 StateCraft and The Caeneus Architecture

In 2006, Arne S. Helgesen and Aleksander Krzywinski published their master thesis titled "The Caeneus Architecture – An Agent Architecture for Social Board Games". The thesis presented "*a general architecture for any kind of social board game using Diplomacy as a testbed*" (Helgesen and Krzywinski, 2006). The Caeneus Architecture was used to implement a game the authors named "StateCraft", a social multiplayer game based on the game Diplomacy. The result of the thesis consists of the Caeneus Architecture, a game engine (StateCraft), an agent (the Caeneus Agent) that operates within the world described in the game engine and a graphical game interface which makes it possible for human players to interact with the agent.

3.3.1 The Caeneus Architecture

Helgesen and Krzywinski (2006) describe the The Caeneus Architecture as a "*one pass, three layered hybrid architecture*". Three layered here means the architecture is divided into three modules, each having separate responsibilities. One pass refers to the fact that the agents using the architecture receives the input (the game state) in one end and sends it through the system from A to B, until a tactic is chosen. The architecture is inspired by the Subsumption Architecture, first presented by Brooks (1990). The Caeneus Architecture differs from the Subsumption architecture by being proactive in addition to reactive, because, as Helgesen and Krzywinski (2006) claim, an agent in the domain of Diplomacy would be useless without being proactive. Proactive behaviour means the ability to act in advance of future situations, rather than just reacting to the current game state, as a reactive architecture would. Wooldridge (2002) referred to an architecture with both of these capabilities as a hybrid architecture.

Helgesen and Krzywinski (2006) describe the three layers as:

> *The **Operational layer** focuses on single pieces and their opportunities*
>
> *The **Tactical layer** focuses on how all the pieces can combine their efforts*
>
> *The **Strategic layer** focuses on diplomatic negotiation and long-term planning*

The layers operate concurrently, and have their own internal computation mechanism for processing received input (Helgesen and Krzywinski, 2006).

3.3.1.1 The Operational layer

The Operational layer of StateCraft is purely reactive. It registers the current game state and generates all the legal operations for every unit. By all legal operations we refer to every move, support, hold or convoy operation a unit can perform without breaking the rules of the game (Helgesen and Krzywinski, 2006).

3.3.1.2 The Tactical layer

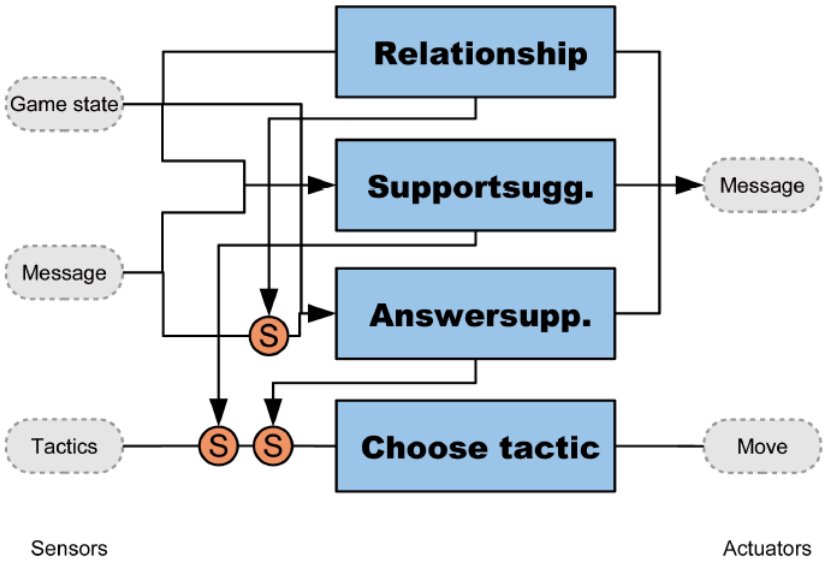
The Tactical layer combines the operations created in the operational layer to generate tactics. A tactic is a combination of operations for every unit for a country. The tactical layer is also responsible for ranking the tactics by a value, describing how "good" a tactic is. The tactics that are considered good are given a high value, while poor tactics are given a low value. This value is based upon several heuristics. For instance a tactic containing a move operation which has the potential of giving the agent control of a supply centre is in most cases more valuable than a tactic containing only hold operations (Helgesen and Krzywinski, 2006).

The Valuator class calculates a tactic's *potential value* and a tactic's *factual value*. The potential value ignores the opponents units and valuates the tactics based on what it **potentially** can achieve. The factual value takes probability of success into consideration, something which is influenced by e.g. the opponent's units. If a tactic has 100% certainty of succeeding, the potential value will be equal to the factual value. Both the potential and factual value are calculated by using several heuristics created for this purpose.

3.3.1.3 The Strategic layer

The Strategic layer is the most complex layer because it has to decide which tasks to do and how to communicate with the other players. The implementation originally consisted of 4 modules (see Figure 3.2):

- The **ChooseTactic** module, which selects the tactic the agent performs, based on its factual values and a little dash of randomness.
- The **SupportSuggestor** module, which analyses the game state, looking for situations where an opponent could contribute with support. It is also responsible for sending messages asking for support.



Inhibitors are not depicted in this picture, as they were not used for any of the modules in the original version of StateCraft

Figure 3.2: Architecture of Strategic layer
(Helgesen and Krzywinski, 2006)

- The **AnswerSupportRequest** module receives support suggestions from the opponents and decides whether the agent should answer yes or no, based on criteria such as relationship to the message sender and randomness.
- The **Relationship** module keeps track of the relationships to the other countries and adjusts this based on the opponents' actions, such as supports and attacks. Relations are represented as states in a Final State Machine, and can either be of state Friend, Neutral or War.
- The **Planner**, which evaluates the agent's position in the game and selects long- and short-term goals.

"*The strength of the Subsumption architecture lies in its ability to accept new modules (behaviours) as more complex behaviours are needed*" (Helgesen and Krzywinski, 2006, p. 52). This has proved to be advantegeous for us, as the main parts of our contributions to the game have fitted seamlessly as modules into the Strategic layer.

Modules in the Strategic layer, like the Subsumption architecture, receive input through sensors and express their output through actuators. Modules are able to override input to other modules by acting as suppressors (see orange S-symbols in Figure 3.2), or override output from modules by acting as inhibitors.

3.4 TacticTree

The TacticTree is a data structure used to generate and represent all the tactics for a given agent. A tactic[1] is a combination of operations[2] for all the units of an agent. An example of a tactic can be:

[F hol -> bel(638) - A mun (supp) kie -> ber(851) - A kie -> ber(670)]

This syntax describes the operation for each unit the agent controls. *F hol -> bel* means that the fleet in Holland moves to Belgium. *A mun(supp) kie -> ber* means that the army in Munich supports the move from Kiel to Berlin. The numbers (638, 851 and 670) represents the *potential value* of each operation in the tactic, i.e. the move operation [F hol -> bel(638)], moving the fleet from Holland to Belgium, has a potential value of 638. The potential value, along with the more conservative *factual value* is created by an object known as the *Valuator*, using several heuristics to calculate how "important" a given operation is.

An agent can have several tactics, where the number of tactics is determined by the number of legal permutations of operations. An example is a country with 10 units, where each unit has 10 possible operations. This leads to a total of 10^10 = 10 000 000 000 tactics. In the original version of StateCraft each tactic was represented as a list of operations, making generation of tactics a repetitive task where each tactic was cloned and changed marginally to accommodate the small change that separates one tactic from another, i.e. the last operation. This made the generation of tactics a very time- and memory-consuming task. Therefore, we introduced the TacticTree as a means of tackling these issues. This allows us to represent the tactics as a tree structure rather than as a flat list-based structure, which we argue[3] reduces the number of clone operations performed during creation of the tactics, as well as the memory footprint of the agent.

The TacticTree (see Figure 3.3) is a tree structure consisting of OperationNodes as its nodes / vertices. An OperationNode is a structure containing a reference to an Operation-object. Each OperationNode has a reference to a parent node, either another Operation-Node or the root. The reference to the parent node makes it possible to traverse the tree from leaf to root, using the path formed by the child-parent references to navigate from a node, starting at the leaf, via its parent, its parent's parent (and so on), until we have

[1] A tactic is represented by the class Tactic

[2] An operation is represented by the class Operation

[3] The increase in performance and reduction of memory consumption for the agent has not been measured as the TacticTree was implemented as a means to combat problems with the original program, rather than as a goal in itself. Although we do not have any quantitative data as a result of benchmark testing, we have observed that the agent is capable of handling more units in an acceptable time frame using the tree structure.

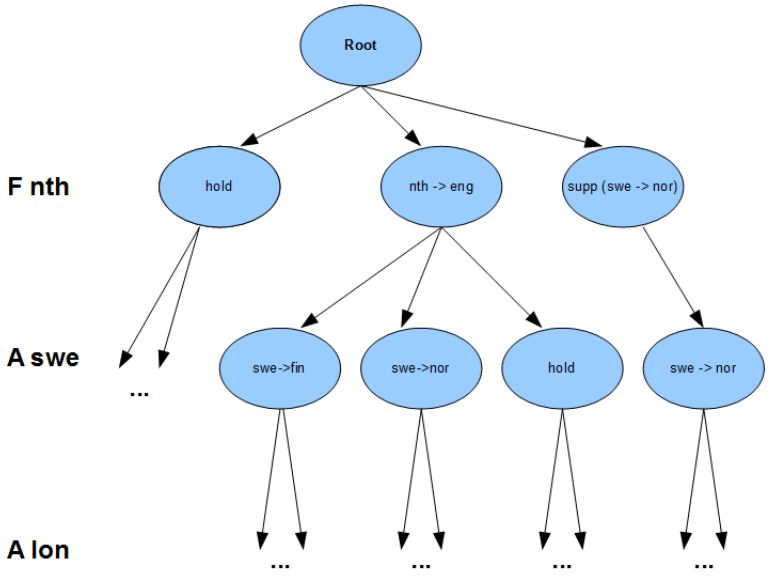


Figure 3.3: TacticTree

Algorithm 3.1 Traversing TacticTree

```
OperationNode currentNode = this;
while(!currentNode.isRoot()) {

    // perform some operation
    currentNode = currentNode.getParent();

}
```

reached the root node (see Algorithm 3.1 for traversal algorithm). There is no reference from parent to child, as this is deemed unnecessary for the tasks the tree performs. A path in the tree is semantically the equivalent to a list-based tactic, where each element in the list is the same as an OperationNode in the TacticTree.

3.4.1 Generating the TacticTree

The class Tactics is given the responsibility of generating the TacticTree. This was the class used to generate the collection of Tactics for the Caeneus Agent in the original version of StateCraft, but the class has been re-factored to accommodate the requirements of the TacticTree. The tree is generated recursively, starting at the top, building in a depth-first order, similar to pre-order tree traversal. The height of the tree is equal to the number of

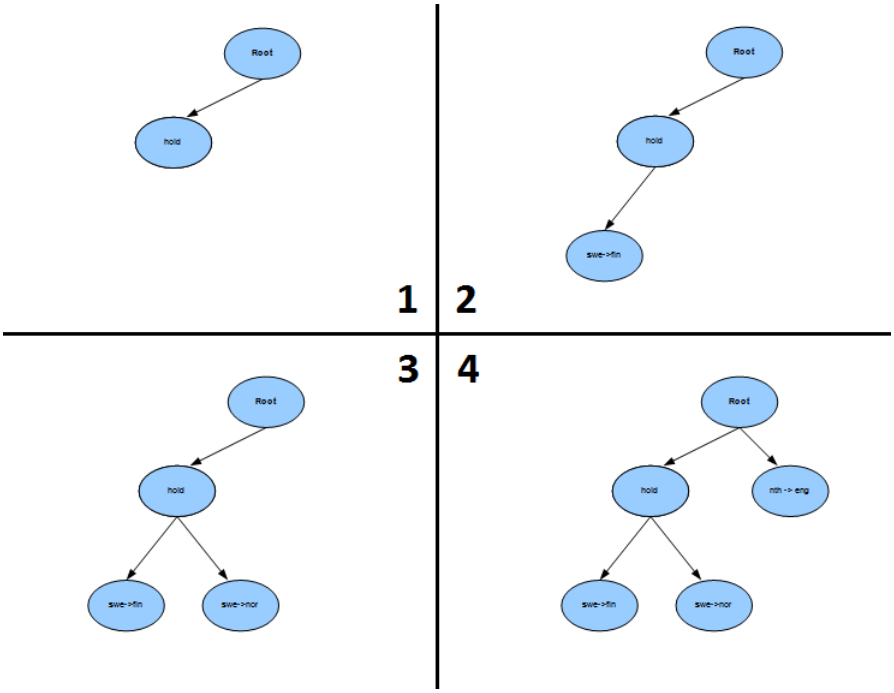


Figure 3.4: Generating the TacticTree

units the agent has, plus one for the root node. Each level in the tree consists of all the operations a given unit has, and no order from another unit will ever exist on the same level.

As visualised in Figure 3.4, a tactic is created by starting at the root node, adding the first operation for the first unit processed as the child of the root node. The algorithm will then continue by performing recursive calls, travelling deeper into the tree, thereby adding an operation for the next unit as the child of the current node. When it reaches a leaf node, the node is used to create an instance of the TreeTactic-class, thus representing a complete tactic. The algorithm then backtracks to the parent node, and continues to add children to the current node until it reaches another leaf node. This cycle will continue until the tree is filled.

Algorithm 3.2 gives a pseudo-code representation of the algorithm used to generate the TacticTree. This is a simplistic overview of the implementation, lacking features such as tactic validation and pruning, but never the less gives a brief walk through of the main parts of the algorithm.

Algorithm 3.2 Generating the TacticTree

```
createTacticTree(currentNode, n)

    if isLeafNode(currentNode)
        addCompleteTactic(currentNode);
        return;
    for each operation for unit n
        nextNode = new OperationNode(operation)
        nextNode.setParent(currentNode);
        createTacticTree(nextNode, n+1);
```

3.4.2 Pruning of the TacticTree

In order to keep the number of tactics to a manageable size, a number of pruning techniques have been introduced. Pruning is the act of removing branches from the tree to reduce its size. By removing potential tactics early, we avoid wasting system resources otherwise used to create illegal, stupid or unnecessary tactics. Pruning in StateCraft is performed by the following algorithms.

3.4.2.1 Mandatory operations

Mandatory operations is a structure used to keep record of any mandatory operation a unit must perform. This structure is necessary to maintain consistency between collaborating units, e.g. a unit supporting or convoying another unit, and the unit being supported or convoyed. When an operation containing a support operation is identified, the unit that performs the supported operation is put into a HashMap supporting key-value look-ups, with unit as **key** and the operation as the **value**, reserving the unit for this particular operation.

To give an example, assume that the first unit (i.e. the first level of the tree) contains a support order, where the supported unit has yet to be processed by the algorithm. The algorithm will then put the reserved unit and the operation the unit is reserved for into the HashMap, thereby making it possible to perform a look-up for reserved operations when the unit is processed. When the unit is reserved for an operation, it cannot perform tasks differing from the one defined in the HashMap.

As seen in Figure 3.5, the grey operations are pruned because the operation *swe -> nor* is the mandatory operation for the army in Sweden.

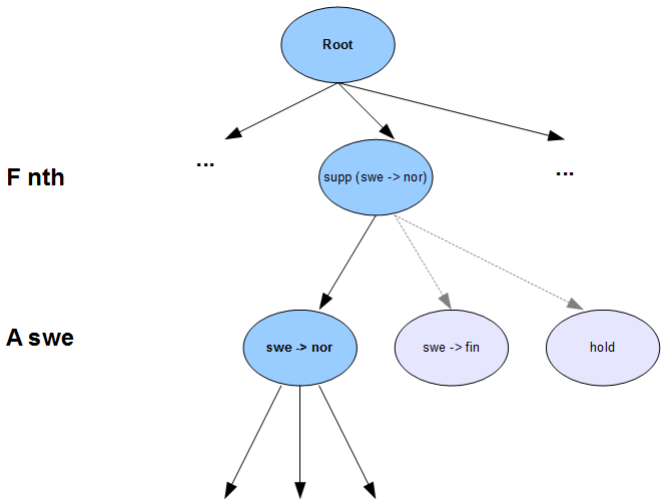


Figure 3.5: Mandatory operations

3.4.2.2 Rules

The original implementation of StateCraft was highly dependent upon rules to remove stupid, illegal or unnecessary tactics. However, after implementing the TacticTree, the original rules became useless, as the TacticTree works differently than the old structure. Therefore, we have re-written all the rules used by the class Tactics to accommodate the new data structure. These are named as following:

TreeCircleRule prunes the tactics where the units are moving in circles.

TreeCollisionRule removes the tactics where two or more units are trying to move into the same province.

TreeConsistencyRule removes the tactics where unit A tries to convoy unit B, but unit B does not perform the move order specified for the convoy.

TreeDoubleBookedConvoiRule prunes the tactics where several units are trying to utilise the same fleet for convoy.

TreeIllegalSupportRule removes the tactics where unit A supports unit B, where unit B does not perform the move order specified for the support order.

TreeUnecessaryConvoiRule prunes the tactics where unit A is convoyed by unit B, although unit A can reach the target without crossing the water.

The common denominator for these rules is that they remove tactics containing operations which are illegal per the rules of the game. In addition to these, three more rules have been added, where the focus have changed from removing illegal or impossible tactics to removing tactics containing operations which, from a human point of view, most likely would be deemed poor or stupid, but not necessarily illegal or impossible to accomplish.

TreeChainedConvoiRule seeks to prune the number of tactics by removing tactics containing convoy-chains considered too complex to be feasible in a game scenario. That is, if a tactic contains a move operation using several convoys to get from one place to another, e.g. from Norway to North Africa, it will often be impossible to accomplish without any enemy interruptions, and is therefore considered unnecessary and will be removed. This is a somewhat special rule as it might be characterised as a *heuristic*, rather than a rule, because a chained convoy is not technically against the rules of the game.

TreeUselessHoldRule seeks to prune the number of tactics by removing tactics containing stupid or useless hold operations. Specifically, it will prune tactics where a unit tries to hold, even though there are no enemies adjacent to the unit. The reason for this rule is that while holding may be feasible in several situations, the agent most likely will have several other good alternative operations for this unit. This is, like TreeChainedConvoiRule, also technically a heuristic.

TreeUselessSupportRule is a rule implemented to prune the number of tactics by removing tactics containing useless support operations. An example is when a unit tries to support another unit moving into a province where the province isn't adjacent to any provinces containing enemy units. That is, the rule removes all tactics containing a support operation and a move operation, where the unit moving is located such that it will never need the support operation to perform the move (i.e. the unit moving is located far away from any enemies).

The new rules are, as already mentioned, heuristics rather than rules, but are implemented never the less because they are helpful in the task of pruning the rapidly (exponentially!) growing TacticTree.

3.4.2.3 Roulette pruning

Roulette pruning is an algorithm used to calculate the value and survival odds of a tactic. The methods works by simulating a roulette wheel from the classical casino game of Roulette. Basically it uses a Java Random-object to generate a double $D1$ between 0.0 and 1.0, and checks this number against a number $D2$ created by algorithms calculating the survival odds of a tactic based on variables such as the maximum potential value of

the unit, a pruning factor calculated by the depth of the unit in the tree, as well as the potential value of the particular operation being processed. If $D1 > D2$, the tactic will "survive", i.e. be stored for later use, and if $D1 < D2$, the tactic will be discarded.

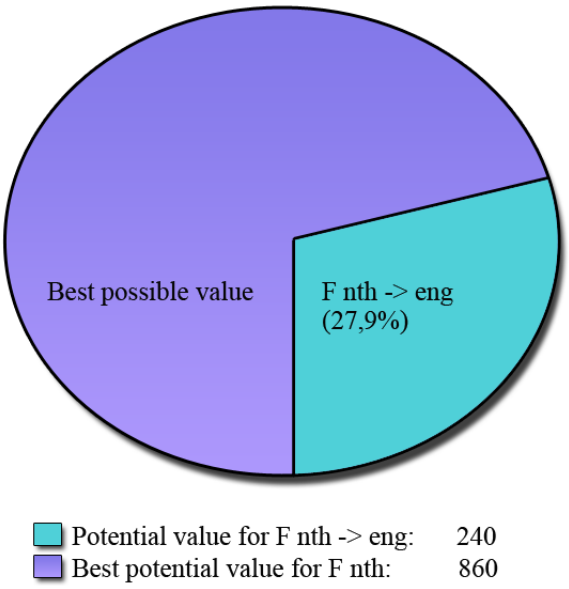


Figure 3.6: Chance of survival

Figure 3.6 visualises the roulette wheel of the algorithm, where an operation has $27,9\%$ chance of survival, based on the potential value of the operation versus the best potential value of the unit. In other words, it is a $27,9\%$ chance that $D1 > D2$ for this particular operation.

3.4.2.4 isRetreating()

isRetreating() is a method that checks whether a unit is moving away from an enemy or not. There are very few good reasons for a player to retreat back into his empire in this game. Unless the player wishes to occupy or protect a supply centre from an advancing enemy, or he has made some sort of alliance with his opponents (forcing him to stay put), he most likely would want to advance into unoccupied territory or enemy provinces.

Therefore, we have chosen to dispose of all tactics where a unit is moving away from the enemy (i.e. retreating). An example of this is when a unit is located at a province two moves away from the closest enemy, but tries to move to a province three moves away from the closest enemy, without having any good reasons for doing so (i.e. moving closer

to an unoccupied supply centre). The method works by performing a breadth-first search for any move operations, where the start position is the province the unit is located at, and the stop position is the closest province containing an enemy. It utilises a HashMap to store the result of a search, with the goal of minimising redundant or duplicate searches.

Chapter 4

Design and development

4.1 About this chapter

This chapter describes in detail the additions that have been implemented in StateCraft as part of our master thesis. All the additions were made to the Caeneus Agent (the StateCraft agent, see section 3.3), originally designed by Helgesen and Krzywinski (2006). The three different modules are:

- The Emotion module
- The Prisoner's Dilemma module
- The TacticTree

The Emotion Module is designed by Christoph Carlson to research the effects of emotional agents in the domain of social board games such as StateCraft.

The Prisoners Dilemma Module is designed by Mathias Hellevang to measure the effectiveness of popular and "good" strategies for the classical game theory problem known as the Prisoners Dilemma, in an n-player strategical board game such as StateCraft.

All modules were implemented using Java programming language version 1.6. Subversion was used for versioning control, and Eclipse was chosen as IDE because of prior experience with the application, as well as the availability of plug-ins such as Subclipse.

Most of the development has been based on agile methodologies, that is, development phases were divided into short increments (usually day-to-day or week-to-week) with minimal planning, with a focus on working code rather than extensive documentation. Sketches and rough drafts were made on post-it notes, large changes were accommodated as they surfaced, and pair-programming have been used whenever possible, where the roles of **driver** and **navigator** were filled in a turn based manner.

4.2 Emotion module

The Emotion module is a module created for the Strategic layer in Statecraft, with the purpose of researching whether emotions will affect the agent's performance or improve the user's game experience.

4.2.1 Choosing emotion model

To successfully synthesise emotions in the agent, we first needed to select an emotion model to base the implementation on. The two alternatives considered were the OCC-model (Ortony et al., 1988) and the three-layered architecture Sloman (1998). The OCC-model was selected as the best choice because simplified versions of the OCC-model have been implemented into several projects, some of them discussed earlier (see Section 2.4.6), and the model has proven to be a good choice for synthesising emotions in agents. In addition, the three-layered architecture lacks implementation-specific details, and it was hard to find research were the model actually was applied.

4.2.2 Identifying emotions in Diplomacy

In order to help decide which emotions would be most relevant to implement for this project, two criteria were set:

1. The emotion must be experienced during a game of Diplomacy
2. The emotion must influence the decision-making to some extent

In order to collect data about which emotions were experienced and how they influenced decision-making, we conducted a player study. 7 players were invited to play the board game. Afterwards, 4 of them were interviewed individually in a semi-structured interview (Grønmo, 2007). They were asked to describe their emotions during the course of the game, and how the emotions affected their choices.

Based on the responses, the most frequently occurring emotions were joy, loyalty, guilt, fear, anger, shame, relief and disappointment. Relief and disappointment seems to lead to the same outcome as joy and anger, and is therefore not included in Table 4.2. Shame was also excluded since it overlaps with guilt.

We, thus, concluded that the most relevant emotions were joy, loyalty, guilt, fear and anger. Joy, fear and anger are already included in the OCC-model.

Emotion	In which situations did the emotion occur?	How did the emotion influence your decisions?
Joy	People tend to feel joy and happiness when things go their way (e.g. gaining a province or beating someone you consider a peer). "When things went my way, I felt happiness", said P1.	The participants seemed to feel like they were on a roll, and maybe not think through their actions thoroughly enough.
Loyalty	P2 stated that he felt more loyalty towards P1, because P1 was the most experienced player, and therefore most likely the best player. P3 stated that he felt loyalty towards P5, since P5 helped him defeat P4 in Munich.	All of the participants answered that loyalty made them keep their promises towards the player they were loyal to, except from P1 who thought it was a stupid thing to do from a cynical perspective.
Guilt	The participants seem to agree that guilt occurred when one lies to another player about supporting him but attacks him instead. P1 also added that a player might feel guilt when he "exterminates" another player very early in the game.	"The guilt might have made me be more nice to Sondre", P3 claimed, while P1 said that guilt had caused him to keep players with few units artificially alive. P2 and P4 did not think that the guilt affected them in any significant way.
Fear	Three of the participants answered that they had experienced some sort of fear while playing Diplomacy, and it seemed to occur when they knew that an opponent could hurt them if he wanted to.	According to the participants, fears leads to a more defencive and cautious approach, thus taking fewer risks.
Anger	Two of our participants experienced anger while playing Diplomacy. The emotion was caused by a negative event (e.g. losing a province) when they had an opponent to blame for the province loss.	P1 claimed that the anger drove him on a seek for revenge. P4 had the same opinion and said that he threw all his resources into getting his province back.

P1 through P7 represents the different participants.

Table 4.2: Summary of interview

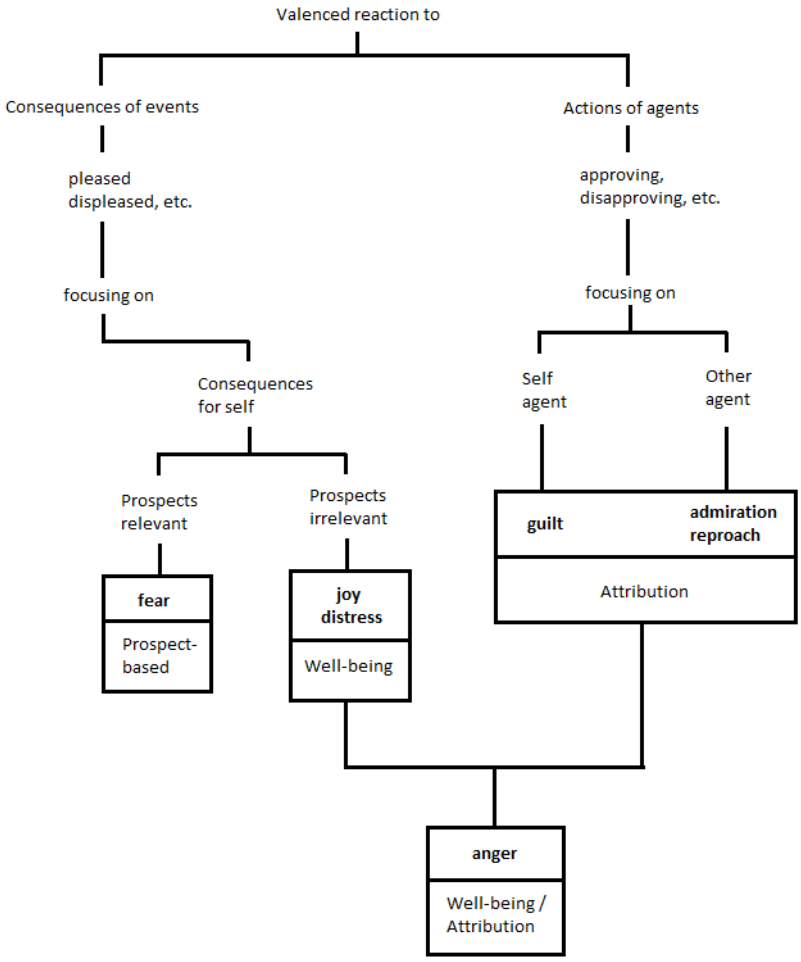


Figure 4.1: Structure used to synthesise emotions in StateCraft (based on the OCC model)

Joy is considered as the reaction to an event which was construed as desirable by the person.

Fear is the prospect of a situation which would be undesirable for that person.

Anger is the combination of **distress** because of an undesirable event and the feeling of **reproach** towards a person who did something one does not approve of.

Loyalty is not defined in the OCC-model, and it is questionable that Ortony et al. (1988) would even categorise it as an emotion, as it is not a valenced reaction to events, agents or objects. However, the situations where the participants reported that they experienced loyalty seem to be when they approved of another player's actions. In the OCC-model, admiration occurs when one approves another agent's actions. Ortony et al. (1988) specified that the meaning of the words used to describe emotions in the OCC-model were not necessarily equivalent to the meaning of the words in spoken English. Therefore, we might say that what the participants actually meant was the emotion that is structured as *admiration* in the OCC-model, since an effect of admiration often will be loyalty to the admirer.

Admiration is the reaction to another agent's action which one approves of.

Guilt is not categorised by the OCC-model either, but for the sake of simplicity, we claim that guilt is a sub-category of the emotion *shame*. Guilt involves disapproving of one's own action towards another person, while also feeling sorry for that other person. This could possibly have been structured as a compound emotion in the OCC-model, but even though it is not included in the model, we choose to model guilt in our implementation because we feel that it plays an important part in the domain of social board games.

Guilt is the disapproving reaction to one's own action combined with feeling sorry for the agent(s) which the action affected in an undesirable way.

4.2.3 Emotion intensity

The intensity of an emotion is represented by a numeric value between 0 and 100, where all emotions start at a default value of 0. The alternative to a numeric value was to implement the emotion as a binary value, meaning that it's either switched on or off. Since emotions differ in intensity (e.g. from mild irritation to furious rage) we considered the numeric value to be the best choice for this implementation. For an emotion to affect the agent's decisions it needs to exceed the threshold set for the particular emotion.

Algorithm 4.1 Anger calculation

```
IF joyValue < 0 AND admirationValue < 0
    IF |joyValue| > joyThreshold AND |admirationValue| >
    admirationThreshold
        anger = sqrt(joyValue * admirationValue)
```

All emotions have a different intensity directed towards each player, except from joy, which only has a general intensity. The emotions joy and admiration also have the possibility of negative values down to -100. A joy value of -100 represents the opposite of joy, which in the OCC-model is *distress*. A negative admiration value represents *reproach*. Alone, distress or reproach does not affect the decisions made by the agent, but together they form the emotion *anger*. The anger's intensity is calculated by calculating the square root of the absolute value of *distress* multiplied with *reproach*, provided that both of these emotions exceed the negative threshold. Pseudo code can be seen in Algorithm 4.1.

To ensure that the agent has a clearly defined emotional state, the agent can only have one emotion towards each country at the same time. The strongest emotion will suppress the other emotions. Take for instance when Germany feels very angry and a little afraid of Austria: since anger is the greatest in intensity, it will suppress the fear. The exception is joy, since joy is general and not directed towards a particular country. However, if fear or guilt is the strongest emotion towards a country, the intensity of joy will decrease. Also, it is impossible to experience joy and anger at the same time, since anger depends on a negative joy value.

4.2.4 Deciding emotion intensity

4.2.4.1 Joy

Joy is the positive reaction to events as specified by the OCC-model, and is therefore naturally affected by the events. The event that increases the joy are:

- The agent gains a province. The intensity depends on the desirability of gaining the province. Gaining a province containing a supply centre will be more desirable for the agent, and therefore increase the joy more than gaining a province without a supply centre.

In the emotion module, distress is modelled as the joy emotion with a negative intensity, because joy and distress are mutually exclusive. The events that decrease joy are:

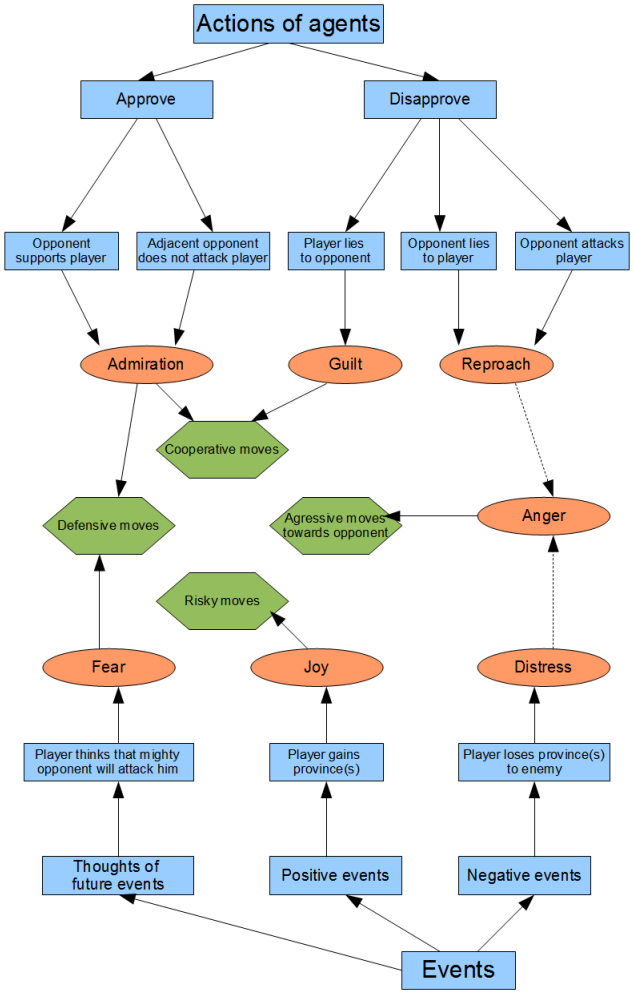


Figure 4.2: Overview of emotion generation

- The agent loses a province. Like when gaining a province, the desirability of keeping the province influences the intensity.
- The agent asks for support. The opponent agrees to perform the support, but for some reason avoids or neglects to perform the support.
- An opponent asks the agent for support and the agent performs the support order, but the opponent does not perform the move order of which they asked support.
- The agent's strongest feeling towards another player is guilt or fear.
- The agent is outnumbered by its opponents.

4.2.4.2 Admiration

Admiration is the approving reaction to actions by other agents, while reproach is the disapproving reaction. Actions made by other agents which the agent approve of:

- An opponent agrees to support the agent and keeps the deal by supporting the agent's attack.
- An opponent with adjacent units does not attack the agent.

Reproach is represented as admiration with a negative value in the emotion module. The actions which decrease admiration are:

- The agent asks an opponent for support and the opponent accepts. Despite this, the opponent does not perform the support move.
- The agent is attacked by an opponent. The intensity of the emotion depends on what the agent expects of the opponent. If the opponent and the agent has a friendly relationship, the admiration towards the opponent will decrease more than, for example, if they were at war.

4.2.4.3 Anger

Anger is the combination of reproach and distress, meaning that when admiration and joy decreases, anger increases. The situations that decrease admiration and joy cause anger to increase (see 4.2.4.1 and 4.2.4.2).

4.2.4.4 Fear

Fear is the negative expectation to an upcoming event. The negative events which the agents are likely to fear in StateCraft are to be attacked by an agent, and to be lied to by someone who agreed to support the agent. For the sake of simplicity, we have chosen to focus on the fear of being attacked by a mighty neighbour. The intensity is calculated based on two factors:

Probability The more adjacent provinces the opponent has to the agent's provinces, the more likely it is that the opponent will attack.

Damage A more powerful opponent can do more damage to the agent than a less powerful opponent.

4.2.4.5 Guilt

Guilt is the disapproving reaction to one's own action combined with feeling sorry for the agent(s) which the action affected in an undesirable way. The events which increase guilt are:

- The agent attacks an opponent it has a friendly or neutral relationship to.
- The agent agrees to support another player, but avoids performing the support operation. The intensity is also dependent on the relationship to the other player.

If the agent performs a support towards a player which it feels guilty towards, the guilt intensity will be reset to zero.

4.2.5 Affecting the agent's decisions

4.2.5.1 Joy

Joy will make the agent perform more risky moves, since it will give it the feeling of "being on a roll".

4.2.5.2 Admiration

Admiration towards opponents will decrease the chance of the agent attacking them. In addition, the agent will more likely perform the support moves it has promised towards the opponents it admire.

4.2.5.3 Anger

Anger towards opponents will increase the chance of the agent attacking the opponents. It will also decrease the chance of the agent supporting them.

4.2.5.4 Fear

Fear towards opponents will decrease the chance of the agent attacking them. It will make the agent more defencive.

4.2.5.5 Guilt

Guilt towards opponents will decrease the chance of the agent attacking them, and make the agent less reluctant to support them.

4.2.6 Implementation details

Given that the Emotion module is an addition to the current agent in StateCraft, the whole module has been implemented in the *emotions* package in the Strategic layer of the agent in StateCraft. Since the Strategic layer uses an architecture similar to the Subsumption system, using sensors to look for changes in the environment and actuators to act on the changes from the sensors, the Emotion module receives input through an input line from GameStateSensor, the sensor listening for new game states from the server, and MessageSensor, the sensor listening for new SupportRequestMessages and AnswerSupportRequestMessages. Then it performs its actions by suppressing the input to ChooseTactic, the module responsible for choosing tactics. Additionally, it inhibits the output from the AnswerSupport module. Figure 4.3 depicts the Emotion module as part of StateCraft's Strategic layer.

4.2.6.1 EmotionSynthesizer

is the main class in the emotions package, used to tie all the different pieces together. It is implemented as a module in the Strategic layer of StateCraft, which means that it has a receive()-method, making it possible to couple the class with the output lines of different sensors in the StateCraft subsumption-like environment, in this case the GameStateSensor and MessageSensor. It also implements the Suppressor interface (see Figure 4.4), which gives it the suppress()-method used to suppress the input to other modules, as well as the Inhibitor-interface, which gives it the ability to inhibit the output of other modules.

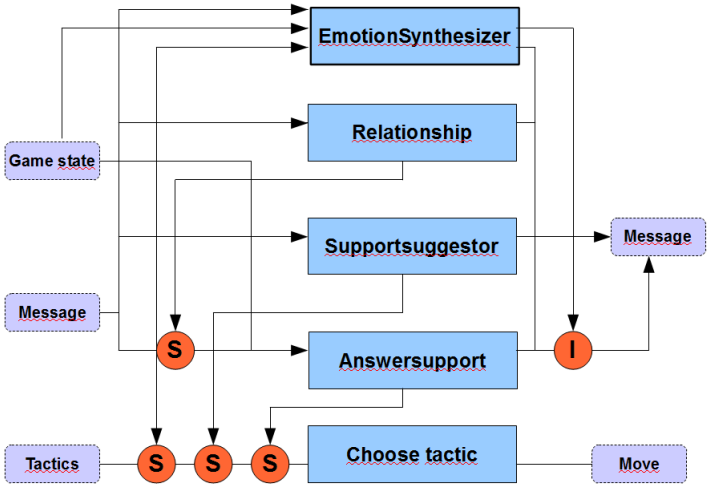


Figure 4.3: Emotion module in the Strategic layer

receive() is the method that receives the GameState and the diplomatic messages, through EmotionSynthesizer's input line that is added to the gameStateSensor in the Strategic class. This ensures that the EmotionSynthesizer receive the GameState object each action round, and thereby passes it to all of the emotions in the emotion list. Each emotion considers the last rounds orders and outcomes and affects the emotion intensity based on it. Each SupportRequestMessage and AnswerSupportRequestMessage is also passed through the receive()-method of EmotionSynthesizer, and it keeps track of the deals made in the preceding round.

suppress() is the method which suppresses the TacticList from the ChooseTactic module and allows the module to remove tactics or change their values. This method calls all the affectChoices()-methods in all the subclasses of Emotion.

inhibit() is the method which inhibits the outgoing AnswerSupportRequestMessage. The message is stored until next round, when the agent will check to see if it performed the support operations it promised.

4.2.6.2 Emotion

is an interface. All classes implementation Emotion is required to implement the methods affectChoices(), affectEmotion() and getValueFor().

affectEmotion() is the method used to affect the emotion intensity based on rules described in each emotion. Each subclass of Emotion implements this corresponding with the rules defined in Section 4.2.4.

affectChoices(TacticList) changes the TacticList based on the emotions' intensities, before it eventually returns it. This is where the emotions influence the agent's decisions according to the rules specified in Section 4.2.5.

getValueFor(Country) gets the emotion's intensity towards the specified country

4.2.6.3 Joy

Joy is a class that implements the Emotion interface. As opposed to all the other emotions, the joy emotion is not directed towards the other agents. It is general, and therefore the intensity is stored in a integer variable and the increase- and decrease-methods ensures that the value can not exceed 100 or go below -100. The intensity of the emotion is changed in affectEmotions() based on the events from the last round. The orders for the next round is changed in affectChoices().

affectEmotion() increases the joy intensity with a medium-sized random value for each supply centre the agent gained in the previous round, and decreases it with a medium-sized random value for each supply centre it loses. If the province contains a supply centre it increase or decrease the value additionally with a large random value.

It also looks through the strongest emotions towards each opponent each round, and for each opponent the agent feels guilty or fear towards, the joy intensity will decrease with a small random value.

In addition, it has a private method that counts how many supply centres the average country own. For each supply centre that the agent has less than $average - 1$, joy will decrease by a small random value.

affectChoices() suppresses the TacticList and increases the factual value closer to the potential value based on the intensity of joy. If the intensity is 100, the factual value will become similar to the potential value. This makes the agent do more risky moves.

getValueFor(Country) gets the general intensity for joy even if a country is specified, since joy is general.

4.2.6.4 Admiration

Admiration is a class which implements the Emotion interface. The agent has an admiration value for each of the country, and therefore the admiration values are stored in a HashMap, where they key is the country and the value is an integer between -100 and 100.

affectEmotion() increases admiration towards the adjacent countries who did not attack the agent last round. The admiration value increases by a small random value for each adjacent province the country has. If the opponent attacks the agent, the agent's admiration will decrease based on how good the relationship to the attacking opponent is.

The method also decrease the admiration value with a medium-sized random value towards opponents that took a province without a supply centre from the agent and a large random value to those that took a province with a supply centre.

affectChoices() increases the value of tactics where the agent attacks

4.2.6.5 Anger

Anger is a class which implements the Emotion interface. The class also contain a reference to Joy and Admiration, since the intensity of anger is calculated based on these two values.

affectChoices() increases the factual value of tactics containing attack orders with supports towards opponents the agent is angry at. It also removes the supports towards countries that the agent is angry at. If the anger intensity is 80, the tactic will be removed 80% of the times.

affectEmotionBasedOnDiplomacy() increases anger towards an opponent with a large random value when the opponents accepts a support request but does not perform the support.

affectEmotionBasedOnPromisedSupportOrders() checks if an opponent which we gave support to performed the move which was promised. If he did not, the anger increases with a large random value.

4.2.6.6 Fear

Fear is a class which implements the Emotion interface. The agent has a fear value between 0 and 100 towards each country, which is calculated from scratch each round.

affectEmotion() This method updates the intensity of fear against each country each round based on how close they are and how many supply centres they have.

affectChoices() This method decreases the factual value for each operation that involves moving into or supporting attacks on countries that the agent is afraid of.

4.2.6.7 Guilt

Guilt is a class which implements the Emotion interface. It contains a guilt value between 0 and 100 towards each of the other countries.

affectEmotion() increases guilt intensity towards the countries the agent attacked the previous round if they had a good relationship, but resets guilt to 0 towards an opponent if the agent supported him last round. It also has a method for increasing guilt if the agent has accepted a support order, but kept from performing it.

affectChoices() increases the factual value of a tactic with x%, where x is the intensity of guilt, if the tactic contains support order where the supportee is someone the agent admires. It also decreases the chance of attacking opponents the agent feels guilty towards.

4.3 Prisoner's Dilemma module

The Prisoner's Dilemma module is a module created for the Strategic layer in StateCraft (see Section 3.3.1.3 for information about the Strategic layer) with the intention of researching whether strategies for the Prisoner's Dilemma will affect the performance of an agent, and the game experience for a user.

4.3.1 Dilemmas in StateCraft

The Prisoner's Dilemma is, as described earlier, a non-zero-sum game used to describe the problem of cooperation vs. defection among rational actors in an environment where

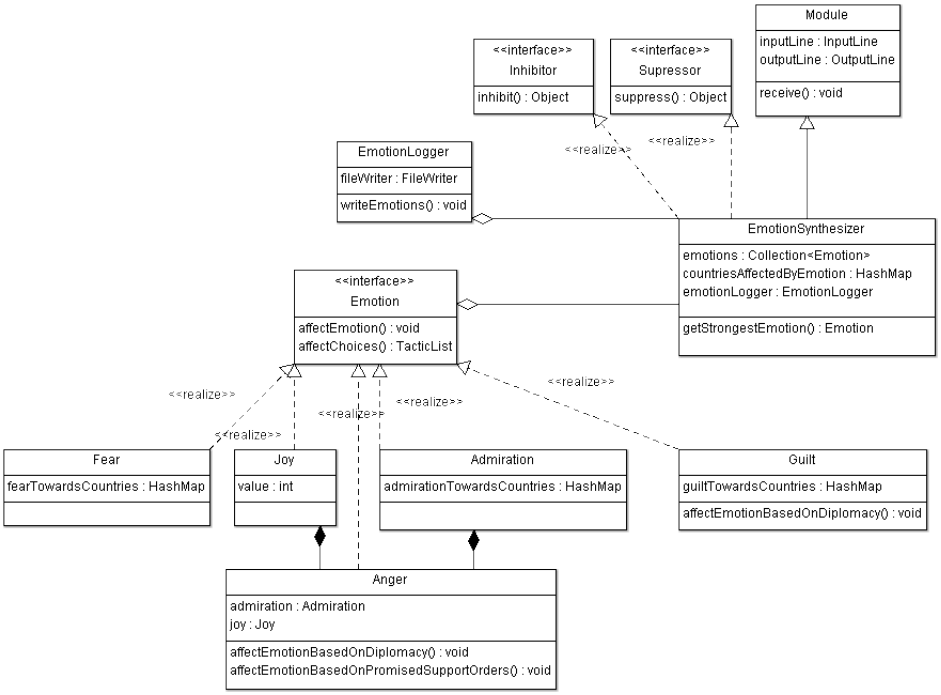


Figure 4.4: A simplified UML Class diagram for the Emotion module

the actors will not necessarily gain directly from cooperating with each other. StateCraft contains several of these situations, but to better confine the problem at hand, dilemmas have been defined as an empty province P1 adjacent to a province P2 containing a supply centre, where two or more agents have units located adjacent to P1. The requirements can also be summarised in list form. A dilemma is:

Prisoner's Dilemma requirements

1. An empty province P1
2. Where P1 is adjacent to a province P2
3. Where P2 contains a supply centre
4. Where two and more agents have units located adjacent to P1

In Figure 4.5 we see several provinces that fits our definition of a dilemma.

- **Silesia** is an empty province without a supply centre, located adjacent to the supply centres in Berlin, Warsaw and Munich. Germany (dark grey player) has a unit

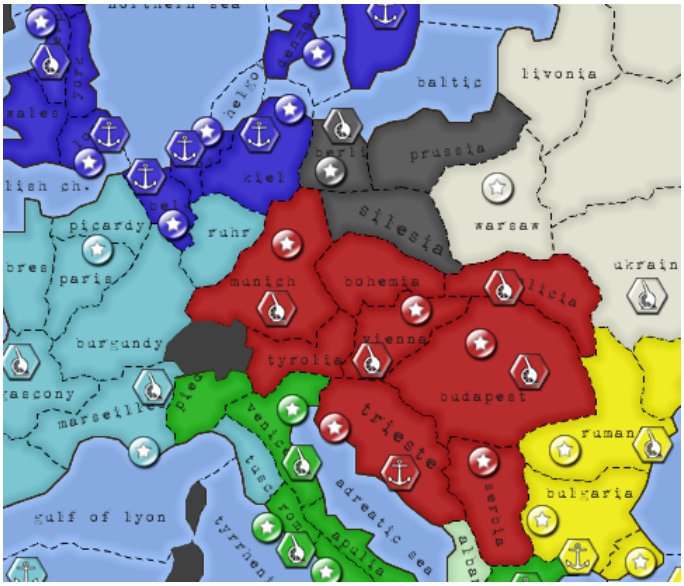


Figure 4.5: Dilemmas in StateCraft

located in Berlin, while Austria-Hungary (red player) has a unit located in Munich.

- **Burgundy** is an empty province without a supply centre, located adjacent to the supply centre in Marseilles, Paris, Belgium and Munich. Austria-Hungary has a unit located in Munich, while France (light blue player) have units located in Marseilles and Gascony.

In addition to these, there are several other provinces which satisfy the basic requirement of being an empty province adjacent to a supply centre (1 through 3 in the definitions for dilemmas), but still should not be treated as a dilemma. For instance, **Ruhr** is an empty province without a supply centre (satisfies requirement 1), located adjacent to the supply centres in Belgium, Holland, Munich and Kiel (satisfies requirement 2 and 3). Austria-Hungary has a unit located in Munich, and England (blue player) have units located in Belgium, Holland and Kiel (satisfies requirement 4). All of England's units are of type fleet, so they are actually unable to move to Ruhr, thus posing no treat to Austria-Hungary's unit in Munich. This means that the province should not be treated as a prisoner's dilemma, but the current version of the Prisoner's Dilemma module unfortunately does not identify these cases.

4.3.2 Implementation overview

Given that the the Prisoner's Dilemma module is an addition to the current agent in StateCraft, the whole module have been implemented in a single package (containing

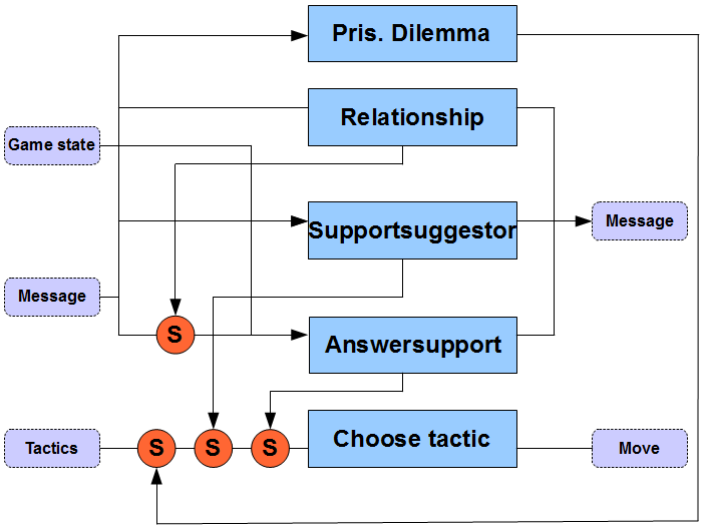


Figure 4.6: Prisoner's Dilemma in the Strategic layer

sub-packages) in the Strategic layer of the agent in StateCraft. Since the Strategic layer uses an architecture similar to the Subsumption system, using sensors to look for changes in the environment and actuators to act on the changes from the sensors, the Prisoner's Dilemma module receives input via an input line from GameStateSensor, the sensor listening for new game states from the server, and performs its actions by suppressing the input to ChooseTactic, the module responsible for choosing tactics. Figure 4.6 depicts the Prisoner's Dilemma module as part of StateCraft's Strategic layer.

4.3.3 Implementation details : Prisoner's Dilemma module

The main Prisoner's Dilemma package contains all classes relevant for the identification of provinces representing potential Prisoner's Dilemma-situations (dilemmas), registering and logging the agent's actions (defection or cooperation), as well as two strategies for the Prisoner's Dilemma, namely the Tit-For-Tat and Friedman strategies. The implementation details of these strategies will be presented in section 4.3.4. A UML Class diagram for the Prisoner's Dilemma module is shown in Figure 4.7.

4.3.3.1 PrisonersDilemma

PrisonersDilemma is the main class of the module. It is implemented as a module in the Strategic layer of StateCraft, which means that it has a receive-method, making it

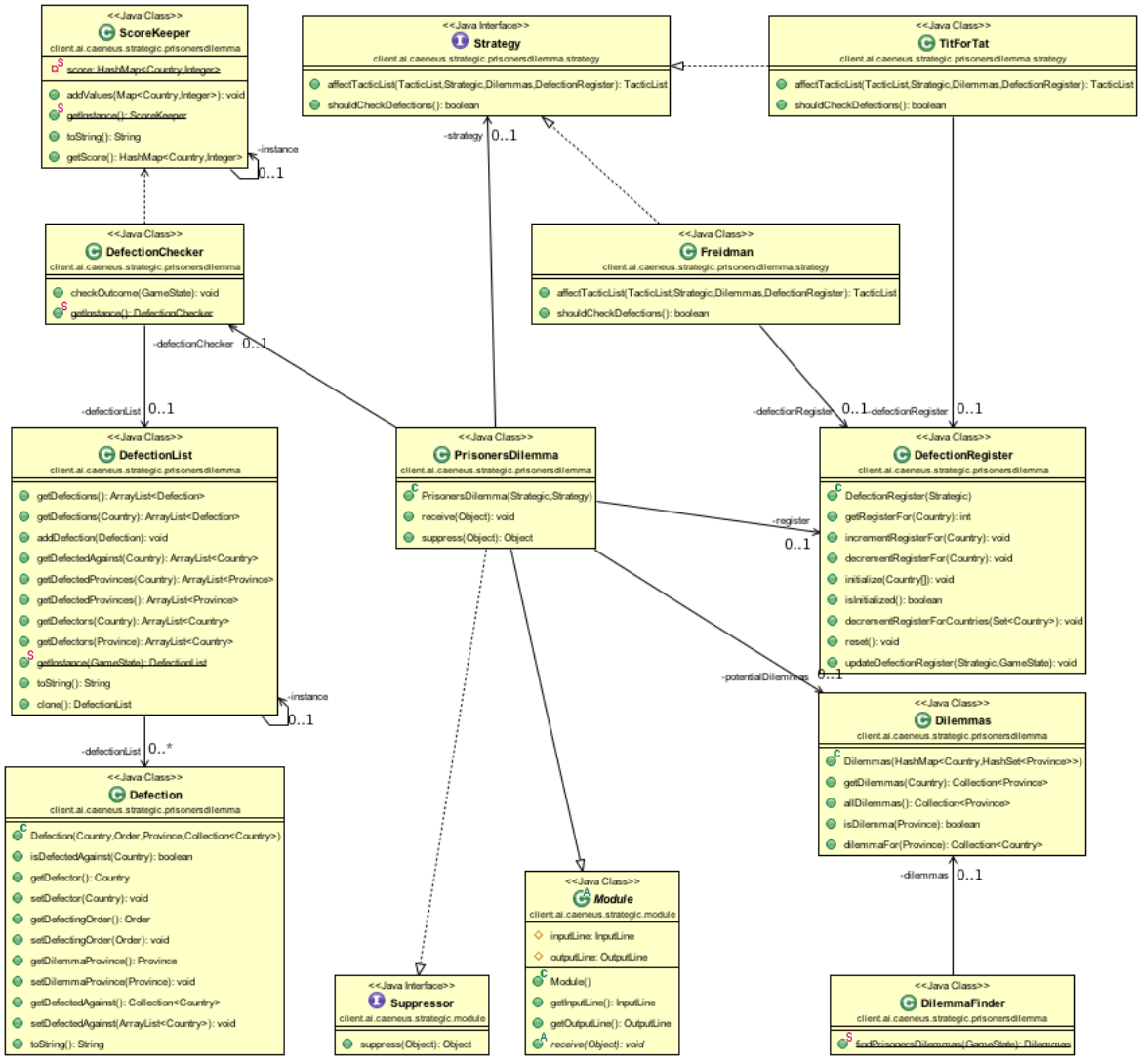


Figure 4.7: UML Class diagram for Prisoner's Dilemma module

possible to couple the class with the output lines of different sensors in the environment, in this case the GameStateSensor.

In addition to being a module, it is also a suppressor, meaning that it can be attached to other modules in the system. Prisoner's Dilemma acts as a suppressor to the ChooseTactic module, which is responsible for choosing which tactic the agent should use. This means that whenever some module is trying to send a TacticList to the ChooseTactic module, Prisoner's Dilemma will catch the input and perform whatever action it deems necessary (e.g. removing tactics or boosting the value of tactics).

4.3.3.2 Defection

Defection is a model used to represent a defection in the Prisoner's Dilemma module. Instances of this class will contain data about which country performed the defection, which operation the defection appears in, which province it represents a defection for, as well as a list containing information about which agents consider this province a dilemma (and, by this, which agents this object represents a defection against).

4.3.3.3 DefectionChecker

DefectionChecker is responsible for registering defections by analysing the outcome of action rounds, and checking the outcome of dilemmas by analysing the actions of the agent the dilemma was relevant for. It creates a DefectionList containing objects of type Defection, representing defections done by the different agents.

The class is accessed by all the agents, but contains conditions limiting the number of calls to the methods in the class to once per season for a given year. The data created by the DefectionChecker is valid for all agents, so when one agent has used the DefectionChecker in a given round, all other agents accessing the DefectionChecker will simply be given the data from the first call.

checkDefections() is the method responsible for registering defections performed in the previous action round. It is called every summer and winter, and uses the current GameState as well as the dilemmas defined for the previous spring or autumn to check, based on the orders from the orderlist, which of the agents that defected. Based on its findings, it creates objects of type Defection, and put them into the static DefectionList.

checkOutcome() is the method responsible for registering the outcome of dilemmas. It is called every spring and autumn. The method is mostly used for logging of the data each round provides us with, and not of real importance for the agents, as they keep their

own internal register of who has defected against them. The method accesses an instance of the DefectionList created by checkDefection in the previous round, and loops through all the Defections in the list, using the current GameState and the Dilemmas for the previous action round to see which agents that cooperated, which agents that defected, and what the outcome of this situation is. It then writes the outcome of the situation as well as a score to a log (see Section 2.5.1 for detailed information about score).

4.3.3.4 DefectionList

DefectionList is a data structure used to represent a collection of defections for a given round. The class is implemented using a variant of the Singleton design pattern, restricting the instantiation of the class to one object. This makes it possible for all the agents to use the same list of defections when analysing what their opponents did in the previous round. This list is created and re-created by the DefecionChecker each summer and winter round.

4.3.3.5 DefectionRegister

DefectionList is a class used by the agents to keep a record of their opponents actions, that is, it serves as the agent's memory of previous defections performed by enemies. Each agent using the Prisoner's Dilemma module will have an instance of this object, and this register will be changed based on the content of the DefectionList which the agents analyse using the strategy they are configured to use (i.e. Tit-For-Tat or Friedman). The necessity for such a register stem from the fact that in Diplomacy, unlike the theoretical IPD-game, each round does not necessarily contain a Prisoner's Dilemma, and as such, if an agent is defected against in round 1, he might not be able to perform whatever action he wants to perform in round 2 (e.g. Tit-For-Tat will defect against his opponent in round 2 if defected against in round 1). Therefore, to make the agent able to perform actions the next time a dilemma occurs instead of only in the next round, this register is implemented.

The register is represented as a HashMap containing a country as a key and an integer as a value. It has several methods for increasing and decreasing a value mapped to a country, so when e.g. England defects against France, France will increase the value mapped to England by one. When France again meets England in a dilemma, France can consult the DefectionRegister for France and, based on the value contained and depending on which strategy France uses, perform some action.

The value mapped to a country can also be decreased by one. For example will an agent using the Tit-For-Tat strategy decrease the value for an opponent by one if the agent has defected against the opponent, i.e. if the agent has performed an act of revenge using defection.

4.3.3.6 DilemmaFinder

DilemmaFinder is a helper class used to generate the Dilemmas-object used by the agents. It loops through all provinces relevant for the Prisoner's Dilemma, identifies which provinces are dilemmas in the current round and puts them in a Dilemmas-object. Like the DefectionList this is also accessible to all the agents, but instead of using a Singleton pattern it is based on a static method (findPrisonersDilemmas()), returning either a new or a stored instance of Dilemmas to the agent calling the method.

4.3.3.7 Dilemmas

Dilemmas is a data structure used to represent all potential dilemmas for a given round. In addition to containing all provinces and all countries affected by every dilemma, it contains several methods to extract information regarding the dilemma (e.g. if a given Province is a dilemma province, which countries a given province represents a dilemma for etc.).

4.3.3.8 ScoreKeeper

ScoreKeeper is a class used to store the score the different players are awarded after the outcome of a dilemma is summarised. Each player involved in the dilemma is given a score depending on whether they defected or cooperated, and if they defected, whether anybody else defected at the same time. See Table 2.1 for values. The score differs from the ones in the table on one issue: if two players P1 and P2 defects, but P1 also receives support for his move either from his own units or from the units of an opponent, P1 will win the province, and is awarded 5 points, while P2 who also defected receives 0 points, as he was not able to win the province according to the rules of StateCraft. This class is of no importance to the agents, and is only used for logging purposes.

4.3.4 Implementation details : strategies

Two strategies for the Prisoner's Dilemma have been implemented, Tit-For-Tat and Friedman. Tit-For-Tat has proven to perform well in a 2p IPD, while Friedman has been found to perform good in a 3p IPD. StateCraft is an N-player IPD[1], and because of these differences, these strategies was selected as interesting candidates for StateCraft. Both strategies are stored in sub-packages of the package **strategy**, located as a sub-package of the Prisoner's Dilemma module.

[1]The number of players in a IPD in StateCraft varies from game to game.

The strategies have one key method in common, affectTacticList(), which performs some action on the list of tactics it receives, based on the state of the DefectionRegister.

4.3.4.1 Friedman

Friedman is implemented in the class with the same name, located in the strategy package. It receives a TacticList, an instance of Dilemmas, and the DefectionRegister. In short, it iterates through all the operations in all the Tactics present in the TacticList, and if an operation is either a support or move with the target of the operation being a dilemma province, Friedman will check if it's been defected against or not. If somebody has defected against the agent in the previous action round, the Friedman strategy will proceed to **boost**[2] all tactics containing operations which targets dilemma provinces. On the other hand, if nobody has defected against the agent, Friedman will **remove** all tactics containing moves to potential dilemma provinces.

The strategy is implemented in accordance with the original specification of Friedman, stating that it should never be the first to defect, but once Friedman is defected against, it will continue to defect against everyone until the end of the game.

4.3.4.2 Tit-For-Tat

Tit-For-Tat is implemented in the class with the same name, located in the strategy-package. It receives a TacticList, an instance of Dilemmas, and the DefectionRegister. Like Friedman, it iterates through all the operations in all the Tactics present in the TacticList, and if an operation is either a support or move with the target of the operation being a dilemma province, Tit-For-Tat will check if it's been defected against or not. If an opponent have defected against the agent in a previous action round, the Tit-For-Tat strategy will proceed to **boost** all tactics containing operations which target dilemma provinces the defector and the agent have in common. However, it will not target dilemma provinces which are shared with both the defector and a third part, unless said third party also have defected against the agent. On the other hand, if nobody have defected against the agent, Tit-For-Tat will **remove** all tactics containing moves to potential dilemma provinces.

The Tit-For-Tat implementation used in the Prisoner's Dilemma module is modified to better accommodate the requirements given in the StateCraft environment. In contrast to the original Tit-For-Tat implementation by Anatol Rapport (Axelrod, 2006), which stated that Tit-For-Tat should cooperate on the first move, but retaliate in the next round when

[2]To boost a Tactic in StateCraft refers to the act of increasing the probability that the selected Tactic will be chosen by the TacticChooser by increasing the Tactic's factual value (which is used to describe how "good" a Tactic is).

defected against, the implementation in the Prisoner's Dilemma module keeps a record of defections over several rounds using the DefectionRegister.

The reason for this is the way a Prisoner's Dilemma province is defined in the Prisoner's Dilemma module. There are only a finite number of relevant provinces in each round, depending on the placement of the players' units and provinces, thus, when a player P1 is defected against, it is not certain that he will be able to retaliate against the defector in the next round. Also, a Prisoner's Dilemma province for player P1 might not be a Prisoner's Dilemma province for player P2 in the next round. This means that player P2 might defect against P1 several times, increasing P1's defection-count against P2. When P1 has the possibility to do so, he will retaliate against P2 by invading P2's Prisoner's Dilemma provinces, and P1 might do so several times spanning several rounds, given that his defection count against P2 is sufficiently high (the count will increase or decrease by 1 for every defection or retaliation).

Chapter 5

Evaluation of Emotion module

5.1 About this chapter

In this chapter we will evaluate the Emotion module. In the introduction of this thesis we described a research question for the Emotion module:

RQ How does emotions affect the performance of the agent and player experience in StateCraft?

This was concentrated into three more specific research questions:

RQ1 How do emotions affect the performance of the agent in terms of supply centres?

RQ2a Can players make a distinction between agents in StateCraft with and without the Emotion module initialised and do they manage to identify what the agent is feeling?

RQ2b Do players think it is more fun to play StateCraft against agents with emotions than against agents without emotions?

RQ1 is explored and tested in Section 5.1.1. Statistical analysis of data collected by running simulations of StateCraft using various configurations of the Emotion module is used to evaluate the questions. All the data have been tested for normality, by using a Shapiro-Wilk test (Shapiro and Wilk, 1965). For the normally distributed data, we used paired sample t-test (Wohlin et al., 2000). For the data that are not normally distributed, we used Wilcoxon signed-rank test (Wohlin et al., 2000). In all the t-tests, we use 95% confident interval.

RQ2a and RQ2b is explored and tested in Section 5.1.3, where user testing is used as the basis of the evaluation.

Table 5.1: Emotion simulation configurations

Configuration	Turkey	Austria	Italy	Germany	Russia	France	England	Amount
All emo	Emo	Emo	Emo	Emo	Emo	Emo	Emo	50
No emo								50
E101	Emo							30
E102		Emo						30
E103			Emo					30
E104				Emo				30
E105					Emo			30
E106						Emo		30
E107							Emo	30

5.1.1 Simulations

To study how emotions affect the agents in terms of performance, we decided to run simulations of the game from 1901 to 1911, where all 7 players were controlled by agents. As depicted in Table 5.1, a total of 9 simulation configurations were tested. The cells marked Emo means that the specified country ran with the Emotion module initialised, later referred to as an emotional agent. An empty cell means that the agent was launched without the Emotion module, later referred to as a regular agent.

The *All emo* and *No emo* listings in Table 5.1 are configurations where all the agents are either running with or without the Emotion module. *E101* through *E107* are configurations where only one of the agents are running the Emotion module. *Amount* describes how many simulations were run for a given configuration (e.g. All emo was run 50 times).

The data used for analysis was mainly the number of supply centres at the end of the game (Spring 1911).

Table 5.2 and 5.3 contain the columns Country, Mean, Std dev, Median, Victories and Extinctions. *Country* contains the name of the country. *Mean* contains the average number of supply centres based on the 50 simulations, while *Std dev* is the standard deviation. *Median* is the value separating the higher half of the sample from the lower half of the sample, but since 50 is an even number, the median can be a decimal number if the two middle values are not equal. *Victories* contains the number of games where the specified country ended with the most supply centres. Occasionally, more than one country "win", for example if both Germany and France have 9 supply centres and all the other countries have 8 or less. *Extinctions* contains the number of times where the specified country ended the game with 0 supply centres.

Table 5.2: Results from No emo simulations

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	7.10	2.978	7	27	0
Turkey	6.44	2.215	6	13	1
England	4.80	1.784	4	4	0
Austria	4.42	2.45	4	7	3
France	4.38	2.39	4.5	5	5
Italy	4.16	1.267	4	0	0
Russia	2.54	2.27	2	2	6

As one can see in Table 5.2, Germany is usually the best country, ending the game with an average of 7.10 supply centres, winning 27 of 50 games. Russia, which is the only country that starts with 4 supply centres, averages at only 2.54 supply centres. However, Russia actually wins 2 of the games, while Italy never win a game, despite having an average of 4.16 supply centres, a much higher average than Russia. Italy seems to be the most stable country, with a standard deviation of only 1.267 supply centres. Italy seems to behave pretty similar in each round, that is, they only move down to Tunisia while keeping their original provinces. England also has a pretty low standard deviation, and the common denominator for England and Italy is that their provinces are mostly surrounded by water, hence more difficult to occupy for opponents. This can explain why both Italy and England never become extinct in neither *No emo* nor *All emo*.

Table 5.3: Results from All emo simulations

Country	Mean	Std dev	Median	Victories	Extinctions
Germany	7.22	3.388	7	23	2
Turkey	6.40	1.796	6	14	0
France	4.96	2.222	5	2	2
England	4.44	1.704	4	5	0
Austria	4.00	2.785	4	7	2
Italy	4.16	1.754	4	2	0
Russia	2.70	2.384	2	8	8

As seen in Table 5.2 and Table 5.3, there are small differences between the simulations *No emo* and *All emo*. One of the biggest differences is that Russia, despite having the lowest average, wins as much as 8 games, compared to only winning 2 of 50 games as a regular agent. The beginning seems to be very critical for Russia. If Russia is too aggressive and leave its borders too open, Germany and Turkey takes advantage of this and occupies Russia's home provinces. Russia wins the two games where Germany is exterminated.

Table 5.4: Example of comparison for testing of individual differences

Configuration	Austria	England	France	Germany	Italy	Russia	Turkey
No emo							
E104				Emo			

Table 5.5: Results from E101 to E107 simulations

Country	Config	Mean	Std dev	Median	Victories	Extinctions
Turkey	E101	6.37	1.542	6	8	0
France	E106	4.87	2.909	5	8	3
Germany	E104	4.73	3.433	5	7	4
England	E107	4.63	1.377	4	1	0
Italy	E103	4.50	1.480	5	0	1
Austria	E102	4.33	1.807	4	1	0
Russia	E105	2.00	1.597	1.5	0	5

5.1.1.1 Individual differences

To study the effects of the Emotion module for a given country, we compared two data sets where the only difference was the country at hand was running the Emotion module in *one* of the data sets, while all the other countries were controlled by regular agents (see Table 5.4 for an example). This enabled us to compare the number of supply centres gained by for example Germany with emotions to the number of supply centres gained without emotions. We decided to run 30 simulations where each country played with emotions against all the other countries without emotions (E101 through E107 in Table 5.1). Then we compared the number of supply centres for each country in the 30 first *No emo* simulations to the number of supply centres the emotional agent had in E101 to E107.

The results from E101 through E107 are presented in Table 5.5. What we notice by comparing the results from Table 5.2 to the results in Table 5.5 is that the emotional agents win fewer times than the regular agents (11.9% to 16.47% of the games), and they become extinct more often (6.19% to 4.29% of the games)[1].

[1]Bear in mind that Table 5.2 is based on 50 simulations, while 5.5 is based on 30 simulations

5.1.1.2 Results for Turkey

Table 5.6: Emotion's effect on performance: Turkey

(a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	6.33	30	1.882
E101	6.37	30	1.542

Turkey	Others	Mean difference	Z	p-value	Based on
Emotion	Regular	−0.033	−.058	0.954	Negative ranks

(b) Paired Samples Test: Emotional agent vs Regular agent

We conducted a Wilcoxon signed-rank test to compare the number of supply centres for Turkey with and without emotions, since only one of the data samples were normally distributed. The results from the Wilcoxon test gave $Z = -0.58$, $p = 0.954$ based on negative ranks. The number of supply centres increased from $6.33 + / - 1.882$ with a regular agent to $6.37 + / - 1.542$ with an emotional agent, meaning that Turkey performed slightly better with emotions. Since $p = 0.954 > 0.05$, we can conclude that the results were not statistically significant.

5.1.1.3 Results for Austria

Table 5.7: Emotion's effect on performance: Austria

(a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	4.60	30	2.298
E102	4.33	30	1.807

Austria	Others	Mean difference	*t*-value	p-value
Emotion	Regular	0.267	0.492	0.627

(b) Paired Samples Test: Emotional agent vs Regular agent

We conducted a paired sample t-test to compare the number of supply centres for Austria with and without emotions, since the data for both samples were normally distributed. The regular agent averaged $4.60 + / - 2.298$ supply centres, while the emotional agent

averaged $4.33 + / - 1.807$. This means that the emotional agent performed 5.8% worse than the regular agent. The results of the t-test was $t(29) = 0.492$, $p = 0.627$. Since $p = 0.627 > 0.05$, the small difference of 0.267 supply centres was not statistically significant.

5.1.1.4 Results for Italy

Table 5.8: Emotion's effect on performance: Italy

(a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	4.07	30	1.172
E103	4.50	30	1.480

Italy	Others	Mean difference	Z	p-value	Based on
Emotion	Regular	−0.433	−1.083	0.279	Negative ranks

(b) Paired Samples Test: Emotional agent vs Regular agent

We conducted a Wilcoxon signed-rank test to compare the number of supply centres for Italy with and without emotions, since none of the data samples were normally distributed. The results from the Wilcoxon test gave $Z = -1.083$, $p = 0.279$ based on negative ranks. The number of supply centres increased from $4.07 + / - 1.172$ with a regular agent to $4.50 + / - 1.480$ with an emotional agent, meaning that Italy performed 10.6% better with emotions. Since $p = 0.279 > 0.05$, we can conclude that the results were not statistically significant.

5.1.1.5 Results for Germany

Table 5.9: Emotion's effect on performance: Germany

(a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	7.0	30	3.248
E104	4.73	30	3.433

Germany	Others	Mean difference	*t*-value	p-value
Emotion	Regular	2.267	2.524	0.017

(b) Paired Samples Test: Emotional agent vs Regular agent

We conducted a paired sample t-test to compare the number of supply centres for Germany with and without emotions, since the data for both samples were normally distributed. The results from the t-test gave $t(29) = 2.524$, $p = 0.017$. The number of supply centres decreased from $7.0 + / - 3.248$ with a regular agent to $4.73 + / - 3.433$ with an emotional agent. This means that Germany plays 32.3% worse with an emotional agent compared to playing with a regular agent. The results are statistically significant, since $p = 0.017 < 0.05$.

5.1.1.6 Results for Russia

Table 5.10: Emotion's effect on performance: Russia

(a) Paired Samples Statistics

Configuration	**Mean**	**N**	**Std. Deviation**
No emo	2.97	30	2.512
E105	2.00	30	1.597

Russia	**Others**	**Mean difference**	***t*-value**	**p-value**
Emotion	Regular	0.967	1.672	0.105

(b) Paired Samples Test: Emotional agent vs Regular agent

We conducted a paired sample t-test to compare the number of supply centres for Russia with and without emotions, since the data for both samples were normally distributed. The results of the t-test gave $t(29) = 1.672$, $p = 0.105$. The regular agent averaged at $2.97 + / - 2.512$ supply centres, while the emotional agent averaged at $2.00 + / - 1.597$. This means that the emotional agent performed 32.6% worse than the regular agent. Since $p = 0.105 > 0.05$, the results are not statistically significant.

5.1.1.7 Results for France

Table 5.11: Emotion's effect on performance: France

(a) Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	4.17	30	2.534
E106	4.87	30	2.909

France	Others	Mean difference	Z	p-value	Based on
Emotion	Regular	−0.700	−0.663	0.508	Negative ranks

(b) Paired Samples Test: Emotional agent vs Regular agent

We conducted a Wilcoxon signed-rank test to compare the number of supply centres for France with and without emotions, since none of the data samples were normally distributed. The results from the Wilcoxon test gave $Z = -0.663$, $p = 0.508$ based on negative ranks. The number of supply centres increased from $4.17 + / - 2.534$ with a regular agent to $4.87 + / - 2.909$ with an emotional agent, meaning that France performed 16.7% better with emotions. The results were not statistically significant, since $p = 0.508 > 0.05$.

5.1.1.8 Results for England

Table 5.12: Emotion's effect on performance: England

Table 5.13: Paired Samples Statistics

	Supply centres for England		
Configuration	Mean	N	Std. Deviation
No emo	4.73	30	1.856
E107	4.63	30	1.377

Table 5.14: Paired Samples Test: Emotional agent vs Regular agent

Country		Supply centres comparison			
England	Others	Mean difference	Z	p-value	Based on
Emotion	Regular	0.100	−0.130	0.897	Positive ranks

We conducted a Wilcoxon signed-rank test to compare the number of supply centres for England with and without emotions, since none of the data samples were normally

distributed. The results from the Wilcoxon test gave $Z = -0.130$, $p = 0.897$ based on positive ranks. The number of supply centres decreased from $4.73 +/- 1.856$ with a regular agent to $4.63 +/- 1.377$ with an emotional agent, meaning that England performed 2.1% worse with emotions. The results were not statistically significant, since $p = 0.897 > 0.05$.

5.1.1.9 Results for all countries

Table 5.15: Emotion's effect on performance: All countries

Table 5.16: Paired Samples Statistics

Configuration	Mean	N	Std. Deviation
No emo	4.84	210	2.603
E101...E107	4.49	210	2.438

Table 5.17: Paired Samples Test: Emotional agent vs Regular agent

Mean difference	Z	p-value	Based on
0.348	−1.089	0.276	Positive ranks

Last, we conducted a Wilcoxon signed-rank test to compare the number of supply centres for all countries with and without emotions, since none of the data samples were normally distributed. The results from the Wilcoxon test gave $Z = -1.089$, $p = 0.276$ based on positive ranks. The regular agent has an average of $4.84 + / - 2.603$ supply centres, while the emotional agent has an average of $4.49 + / - 2.438$ supply centres. This means that based on the simulations for all countries, the agent performs 7.2% worse if it has emotions. However, the results are not statistically significant, since $p = 0.276 > 0.05$.

5.1.1.10 Summary

As we can see from Table 5.18, the only statistically significant result is the change in number of supply centres for Germany. A positive mean difference indicates that the agent performs worse with emotions than without emotions. In other words, Germany ends up with 2.267 fewer supply centres when controlled by an emotional agent.

5.1.2 Analysis of data simulations

To make sense of the results from the simulations, we needed more information than supply centres and geographic position. Therefore, we analysed which emotions the different

Table 5.18: Summary of statistical analysis

Country	Config 1	Config 2	Mean difference	Improvement	Test	Significance
Turkey	No emo	E101	-0.033	$0,63\%$	Wilcoxon	0.954
Austria	No emo	E102	0.267	$-5,8\%$	Paired t-test	0.627
Italy	No emo	E103	-0.433	$10,6\%$	Wilcoxon	0.279
Germany	No emo	E104	2.267	$-32,3\%$	Paired t-test	0.017
Russia	No emo	E105	0.967	$-32,6\%$	Paired t-test	0.105
France	No emo	E106	-0.700	$16,7\%$	Wilcoxon	0.508
England	No emo	E107	0.100	$-2,1\%$	Wilcoxon	0.897
All	No emo	E101...E107	0.348	$-7,2\%$	Wilcoxon	0.237

countries were experiencing, so all the log files containing the different emotion's intensities were parsed and each emotion was counted and given a score.

5.1.2.1 Counting emotion occurrences

The strongest emotion for each country was counted, and the percentages of how frequent each emotion occurred for each country can be found in Table 5.19. Joy is the most frequent emotion because it is general and not country specific. Thus if the joy is strong it is probable that it has a higher value than most, or all, of the other country specific emotions, and hence is counted up to six times. This may lead to skewed data, with joy being overrepresented.

Table 5.19: Number of emotion occurrences

Country	Joy	Anger	Guilt	Fear	Admiration	No emotion
Turkey	30.4%	1.4%	1.0%	0.7%	2.9%	63.6%
Austria	41.2%	9%	1.92%	10.2%	26.2%	12.5%
Italy	23.9%	5.9%	2.1%	3.0%	6.1%	58.9%
Germany	18.2%	6.4%	2.4%	6.2%	15.5%	51.3%
Russia	4.1%	6.3%	1.3%	10.1%	8.3%	69.5%
France	36.6%	3.2%	1.2%	3.8%	8.9%	46.3%
England	52.6%	5.2%	2.6%	1.7%	3.4%	34.4%

As seen in Table 5.19, Austria is the most emotional country, feeling emotions 87.5% of the time, while Russia and Turkey only feel emotions $30 - 40\%$ of the time. Guilt is the rarest emotion, while joy is the most common.

5.1.2.2 Emotion score

The score was calculated like this: If an agent was experiencing a certain emotion (e.g. guilt) and gained a province the round afterwards, then that emotion would gain a point. If the agent loses a province after a round of feeling the emotion, then the emotion would lose one point. The emotion scores can be seen in Table 5.20.

Table 5.20: Emotion scores for different countries

Country	**Joy**	**Anger**	**Guilt**	**Fear**	**Admiration**	**No emotion**
Turkey	184	4	7	8	9	392
Austria	33	−4	4	−15	73	179
Italy	−14	3	7	15	38	227
Germany	32	−18	8	−22	50	264
Russia	−66	−7	−14	−43	−58	−166
France	120	−13	−2	−15	7	239
England	−43	30	5	7	7	348

Table 5.21: Normalised emotion scores for different countries

Country	**Joy**	**Anger**	**Guilt**	**Fear**	**Admiration**	**No emotion**
Turkey	0.04	0.02	0.05	0.08	0.02	0.04
Austria	0.01	−0.01	0.03	−0.02	0.04	0.21
Italy	−0.01	0.00	0.03	0.04	0.05	0.03
Germany	0.02	−0.03	0.03	−0.03	0.03	0.05
Russia	−0.13	−0.01	−0.09	−0.03	−0.06	−0.02
France	0.04	−0.04	−0.02	−0.04	0.01	0.06
England	−0.01	0.07	0.02	0.05	0.03	0.13

Table 5.21 shows the results from Table 5.20 divided by the number of times the emotion was experienced. Russia, which has a joy score at −0.13, loses an average of 0.13 provinces each time it feels joy. A positive joy score indicates that the agent on average *gains* provinces when it feels joy. Austria gains as much as 0.21 provinces each time it does not feel emotion. The explanation for this is probably that most of the rounds where Austria does not feel anything are the first rounds of the game, when it usually occupies the unoccupied supply centres in Serbia and Romania, thus gaining several points for *No emotion*.

5.1.2.3 Analysing effect on performance for individual countries

As we see in Table 5.18, Germany performs more than 30% worse with emotions. It seems like the same might be the case for Russia, but we could not find any significant difference with only *30* cases, even if the mean difference was above 30%. France and Italy performs $10 - 20\%$ better with emotions, but these results are not statistically significant either. The other three countries perform pretty similar with emotions. The different results for the different countries can be explained by their geographical positions.

Germany is geographically located next to many empty supply centres, indicating that the agent often will go towards these supply centres in the early rounds. This will lead to an early increase of joy, which will make Germany take many riskful moves the following rounds. Riskful moves do not seem to be a good idea for Germany, since that often results in it moving away from its home provinces. The home provinces are very crucial, since they are a necessity to build more units, and Germany's home provinces are close to many neighbours, meaning that they are easily taken over by other countries. Russia averages at 4.57 supply centres when Germany is emotional compared to 2.54 when all agents are regular. As seen in Table 5.21, Germany performs best when it experiences the emotions admiration and fear, while it performs the worst when it is angry. This indicates that Germany performs best with a more defencive strategy.

Austria is located south of Germany, and is in the same situation as Germany with borders to many opponents. However, Austria only performs 5.8% worse with emotions, while Germany performs more than 30% worse. One of the explanations can be that Germany has more potential to expand than Austria. When Germany is being played by an emotional agent, Russia takes advantage of Germany's aggressiveness and steals its home provinces. From observing the gameplay, no country seem to cause Austria these problems. In addition to this, Austria's home provinces are able to support each other and can easily squeeze out intruders to a less threatening position, because there are few neighbouring provinces that can support an attack.

Russia begins with 4 supply centres, while all the other countries begin with 3 supply centres. Therefore, Russia is not close to many unoccupied supply centres and will feel less joy, since it does not gain many new provinces in the beginning. This corresponds with the data from Table 5.19, where we see that Russia is only joyful 4.1% of the time. However, when it is joyful, it performs bad (see Table 5.21), losing an average of 0.13 supply centers, probably because it has a very wide border to Turkey, Austria and Germany, which is hard to defend if it takes huge risks. It loses fewer supply centres with the defencive emotions fear, admiration and guilt than it does when it feels joy.

France is located west on the map, and can basically only be infiltrated through Picardy, Burgundy and Marseille without going the sea way. France is also close to two unoccupied supply centres (Spain and Portugal), which none of the other countries are close to. This will lead to an early increase of joy, and therefore more risky tactics. However, taking big risks are not necessarily a stupid idea for France, since it is hard for the enemies to get pass France's units and into its home provinces. This means that, unlike Russia and Germany, it might get away with making riskful moves.

Italy is positioned in a way that makes it really hard to infiltrate it without fleets. But at the same time, it is difficult for Italy to expand, since most of the traffic goes through their two provinces Venice and Piedmont. Therefore it makes sense that it is able to make more riskful moves and get away with it.

England is the only country surrounded by water on all sides. This keeps them rather safe from intruders, since the opponents have to build fleets to attack them. However, it also makes it difficult to expand and gain new provinces. England is the country which feels the most joy, and the reason for that is probably that England often gains a couple of provinces in Norway and Denmark in the early rounds and keeps them throughout the game.

Turkey is isolated in the south east corner of the map, and all the traffic to Turkey's home provinces usually goes through the Black Sea and Bulgaria. It is pretty safe for Turkey to expand aggressively, because if it fails, it can always go back and try again without being afraid of opponents taking its home provinces.

5.1.3 User testing

In order to study research questions *RQ2a* and *RQ2b* we needed to involve users in the tests. The participants consisted of five males and one female, aged between 23 and 31 years. All of the participants were of Norwegian nationality. Four of them had prior experience with Diplomacy, while two had never played the game before, but all of the players had previous experience with other computer games. The inexperienced players were given a brief introduction to the rules and game mechanics of Diplomacy. Each participant played *3* games of StateCraft to 1905. The participants played Austria in all games, because of Austria's geographic position, with many neighbouring countries. First, they played a game against a mixture of emotional agents and regular agents (referred to as *Game 0*) and were asked to identify exactly which emotions the agents were feeling towards them. Then they played one game against all emotional agents (referred to as

Game A) and one game against all regular agents (referred to as *Game B*). The sequence of Game A and B were mixed to prevent skewed results due to the players becoming better at the game. After playing through Game A and B they were asked if they observed any differences between the two games, and if so, which differences they found and if they thought one of the games were more fun than the other.

5.1.4 Findings

5.1.4.1 Identifying different emotions

Table 5.22: Summary of emotion identification

Player	Correct	Wrong	N	Missed	Prev. experience	Probability (Correct > 0)
Player A	0	7	7	7	Yes	94.4%
Player B	0	1	1	6	Yes	20.0%
Player C	0	2	2	6	No	38.7%
Player D	0	4	4	2	Yes	39.3%
Player E	1	3	4	3	No	48.0%
Player F	1	2	3	6	Yes	55.0%

The results from Game 0, where the users tried to identify the agent's emotions, can be found in Table 5.22. The *Correct* column contains the number of emotions the player was able to identify correctly, while *Wrong* contains the number of misinterpreted identifications. $Correct + Wrong = N$, where N is the number of emotions the user thought he observed for all agents combined. *Missed* contains the number of emotions that the agent was feeling, but the user was unable to identify correctly. *Probability (Correct > 0)* contains the probability of a user identifying at least one emotion correctly by randomly guessing N times.

When asked to identify the different emotions, only two players managed to pick an emotion that the agent actually felt towards them. By this, we can conclude that players are unable to identify what the agent is feeling, at least on such a short time frame. Another explanation may be that the players are too focused on playing the game, especially the inexperienced ones. However, 1/2 of the inexperienced users managed to identify an emotion, while only 1/4 of the experienced users did. It is also important to point out that the users do not know how emotions would be expressed through the actions of the agent. Also, an agent's aggression caused by a high joy intensity may be misinterpreted by the users as the aggression caused by a high anger intensity. Last but not least, one might claim that the interface between an agent and the players is too limited for the agent to convey its emotion to the player. It is very difficult to convey an emotion through the

Table 5.23: Summary of user tests

Player	Game 0	Game A	Game B	Most fun
Player A	4	4	1	Game A
Player B	0	2	5	Game A
Player C	1	6	2	Game B
Player D	2	6	1	Game A
Player E	6	3	2	No difference
Player F	7	12	4	Game A
Average	3.33	5.5	2.5	Game A

orders in a board game, since the only orders a player can perform in Diplomacy are move, hold, convoy and support. If there was a better way to communicate, like for example a chat service, the emotions would be much easier to express.

5.1.4.2 Increasing player experience

The results from Game A and Game B, where the users played one game against all emotional agents (Game A) and one game against all regular agents (Game B), can be seen in Table 5.23. Four of the players thought that it was more fun to play against emotional agents, while one player thought it was more fun to play against the regular agents. Player E did not notice any difference between the two games. Several of the participants mentioned that the game against emotional agents were more fun since they performed better themselves, and therefore the game appealed to their feeling of mastering. This is also supported by the fact that the only player who performed better against the regular agents is the same person that found it more fun to play against the regular agents. Player F also supported this: "*Game A was more fun. Simply because I got 12 centres, but it was the second game that was most challenging. But since I lost in the end, it was less fun*".

When asked if they observed any differences between the two games, all participants except from Player E thought there was a difference. Player C stated that: *"Assuming Game A had emotions, Game B felt very straightforward. Enemies never backed down and kept pushing at all fronts".*

On average, the players end up with an average of 2.5 supply centres when they play against regular agents compared to an average of 5.5 supply centres when they play against emotional agents. This corresponds with the results from the data simulations, where Germany and Russia performed far worse with emotions. Austria, which is the country that the users played, is a neighbour to both Germany and Russia, and this can explain why most of the players performed better against the emotional agents. One

cannot argue that the players had become better at the game when playing against the emotional agents, because half of the test group played against the emotional agents first.

5.1.5 Summary

The results from the data simulations show that some countries perform worse with emotions. The countries that perform worse are the countries which are better off using a more conservative approach. However, not all countries are punished for using emotions, and some countries even perform better with emotions. The common denominator for these countries is that their home provinces are hard to infiltrate for opposing countries.

We were able to determine that it is very difficult for a human player to identify which emotions the agents were feeling in such a short amount of time. But even if the players did not manage to identify the agents' emotions, it appears that most of the participants thought it was more fun to play against agents with emotions than agents without emotions.

Chapter 6

Evaluation of Prisoner's Dilemma module

6.1 About this chapter

In this chapter we will evaluate the Prisoner's Dilemma module. In the introduction of this thesis we described the following research question for the Prisoner's Dilemma module.

RQ How does the PD strategies affect the performance of the agent and player experience in StateCraft?

This was divided into three more specific research questions:

RQ1 How do the Prisoner's Dilemma strategies affect the performance of the agent in terms of supply centres?

RQ2 How do the strategies affect the performance of the agent in terms of score?

RQ3 Can players make a distinction between StateCraft with and without the Prisoner's Dilemma module?

RQ1 and RQ2 is explored and tested in Section 6.2. Statistical analysis of data collected by running simulations of StateCraft using various configurations of the Prisoner's Dilemma module is used to evaluate the questions.

RQ3 is explored and tested in Section 6.3, where user testing is used as the basis of the evaluation.

Table 6.1: Prisoner's Dilemma simulation configuration

Configuration	Austria	England	France	Germany	Italy	Russia	Turkey
All Dummy	Dummy	Dummy	Dummy	Dummy	Dummy	Dummy	Dummy
All Friedman	Friedman	Friedman	Friedman	Friedman	Friedman	Friedman	Friedman
All TFT	TFT	TFT	TFT	TFT	TFT	TFT	TFT
T101			TFT				
T102				TFT			
T103		TFT					
T104						TFT	
T105	TFT						
T106							TFT
T107					TFT		
F101	Friedman						
F102		Friedman					
F103			Friedman				
F104				Friedman			
F105					Friedman		
F106						Friedman	
F107							Friedman

Note that "Dummy" is removed from the configurations T101 through F107 to enhance readability.

All the data have been tested for normality, by using a Shapiro-Wilk test (Shapiro and Wilk, 1965). For the normally distributed data, we used paired sample t-test (Wohlin et al., 2000). For the data that were not normally distributed, we used Wilcoxon signed-rank test (Wohlin et al., 2000). In all the t-tests, we use 95% confident interval.

6.2 Simulations

To evaluate how the agent performed in terms of supply centers and score, we ran a series of simulations where all seven players were controlled by agents. As depicted in Table 6.1, a total of 17 simulation configurations were tested. Each configuration used a different combination of strategies for the country, where each country was configured to use the Dummy, Friedman or Tit-For-Tat-strategy (TFT). The Dummy strategy is, as the name implies, a placeholder for a real strategy, meaning that the term can be interpreted as the regular agent without TFT or Friedman, used as the baseline for comparison. Using the Dummy strategy is equivalent to not using a strategy at all, and the term *Dummy* will be used interchangeably to describe simulations or a configuration where the country at hand is not using a Prisoner's Dilemma strategy.

The *All Dummy*, *All Friedman* and *All TFT* listings in Table 6.1 are configurations where all the agents are using the same strategy (either Dummy, Tit-For-Tat or Friedman), while T101-T107 and F101-F107 are configurations where only one of the agents are running the Friedman or Tit-For-Tat strategy. The Dummy, Friedman and Tit-For-Tat configurations were run 50 times each, while T101 through T107 and F101 through F107 were run 20 times each. Each simulated game started in Spring 1901, and lasted until Spring 1911. A total of 430 simulated games were collected for the analysis of the Prisoner's Dilemma module.

The data used for the analysis was mainly the score and number of supply centres each country had gained in the last round (Spring 1911) of the simulations.

6.2.1 Effect on performance : mean difference

To measure the general effect on performance, we looked at the number of supply centres each country had in the last round of the game (Spring 1911). We calculated the standard deviation for each simulation for each country, meaning that we found how much the average country differed from the average number of supply centres. We then used this data to combine and calculate an average standard deviation for a given configuration, i.e. we combined the standard deviation for each country to find the average standard deviation for all countries running a given configuration. These numbers are presented in Table 6.2a under the column *Mean difference*.

We created three such data sets: one where all agents were running the Dummy strategy (*All Dummy*), one where all the agents were running the Tit-For-Tat strategy (*All TFT*), and one where all the agents were running the Friedman strategy (*All Friedman*).

The data sets were tested for normality, and we found that all sets were normally distributed. Because of this, the sets for All TFT and All Friedman where compared with the set for All Dummy using a paired sample t-test.

The results of this comparison is presented in Table 6.2

All Dummy versus All TFT

All Dummy vs All TFT was compared using the paired sample t-test. The countries running the Dummy strategy had an average standard deviation of 1.66, while the countries running all TFT had an average standard deviation of 1.33. The result of the t-test showed $t(49) = 5.521$, $p = 0.000$. This shows that the simulations where the agents are all running the Tit-For-Tat strategy have considerably lower average standard deviation than the simulations where the agents are all running the Dummy strategy, with a difference of 0.331. A lower average standard deviation for a given strategy can be interpreted as a

Table 6.2: Effect on performance: Mean difference

(a) Paired Samples Statistics

Configuration	Mean difference	N	Std. Deviation
All Dummy	1.660	50	0.474
All TFT	1.329	50	0.467
All Friedman	1.172	50	0.522

Configuration				
Sample 1	**Sample 2**	**Mean**	***t*-value**	**p-value**
All Dummy	All TFT	0.331	5.521	0.001
All Dummy	All Friedman	0.488	3.702	0.000

(b) Paired Samples Test comparison table

more *balanced* game for the strategy, i.e. that the agents performs more equal using this strategy, if all agents are using the same strategy, compared to a test where none of the agents are using any strategies for the Prisoner's Dilemma. We conclude that All TFT makes the game more balanced.

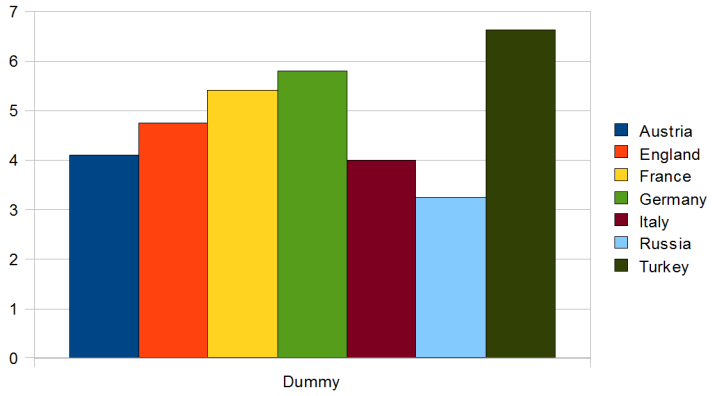
All Dummy versus All Friedman

All Dummy vs All Friedman was compared in the same manner as *All Dummy* vs *All TFT.* A paired sample t-test was used to perform the comparison between the two data sets, and the outcome showed that the countries running *All Dummy* had an average standard deviation of 1.66, while the countries running *All Friedman* had an average standard deviation of 1.17. $t(49) = 3.702$, $p = 0.001$. The results show the same trend as for All TFT, i.e. that the strategy makes the game more balanced if all the other agents are using Friedman, compared to a game where the agent and all the other players are playing no strategy.

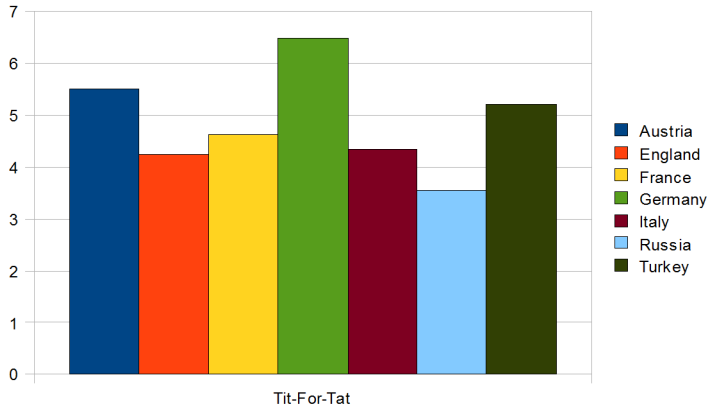
All Friedman showed an ever lower average standard deviation than All TFT, with a difference of 0.448 compared to All Dummy.

Summary

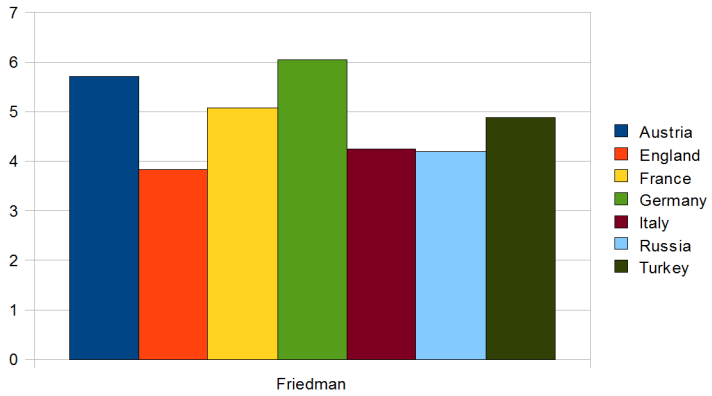
What we found was that in a game where all the agents were running either the Tit-For-Tat or the Friedman strategy, all the agents would perform more similar or *balanced* compared to a game where all the agents were using the Dummy strategy. Figure 6.1 shows how Friedman and Tit-For-Tat leads to a more balanced game, visualised through the average number of supply centres each country gets using the different strategies.



(a) Dummy



(b) Tit-For-Tat



(c) Friedman

Figure 6.1: Average supply centre differences

Table 6.3: Example of comparison for testing of individual differences

Configuration	Austria	England	France	Germany	Italy	Russia	Turkey
All Dummy	Dummy	Dummy	Dummy	Dummy	Dummy	Dummy	Dummy
T105	TFT	Dummy	Dummy	Dummy	Dummy	Dummy	Dummy

6.2.2 Effect on performance : individual differences

To measure the effect on performance for individual countries, we had to look at how a individual country performed when it was the only country using a strategy, compared with how it would perform when none of the countries were using a strategy.

We used 20 of the 50 simulations where all the countries were using the Dummy strategy (i.e. no strategy), and for each country, we compared this to 20 simulations where only the country at hand were running either Tit-For-Tat or Friedman, while the rest were running Dummy. All in all, we performed 14 such comparisons, two for each country. Table 6.3 depicts an example of how the comparison was performed, where the different rows represents a different configuration. In the example table we can see how Austria's performance was compared in two configurations. We looked at the performance of Austria when it was the only country using the Tit-For-Tat strategy versus how it performed when none of the countries were using any strategy.

The data samples were tested for normality, and we found that the majority of the data sets were not normally distributed. Because of this, a Wilcoxon signed-rank test was chosen to perform the comparison. Wilcoxon signed-rank test is a type of non-parametric test used to find differences between two related samples. It is used instead of the t-test because it, unlike the t-test, does not assume that the samples are normally distributed.

In the tables presented below, a negative mean value implies that the tested strategy performed better than the Dummy strategy it was compared against, while a positive mean value implies that the Dummy strategy performs better than the tested strategy.

The tables are divided into three parts, "(a) Paired Samples Statistics", "(b) Paired Samples Test: supply centres" and "(c) Paired Samples Test: score". The *Paired Samples Statistics* table show how the country at hand perform individually using either Dummy, TFT or Friedman, looking at the the number of supply centres and the Prisoner's Dilemma score respectively. The two *Paired Samples Test* tables depicts the results from the Wilcoxon signed-rank test, that is, how the country at hand performs in terms of supply centres and Prisoner's Dilemma score using either TFT or Friedman, compared to when it is using the Dummy strategy.

"Sample 1" in the two Paired Samples Test tables tables refers to the base case the country is compared to, i.e. "All Dummy", while "Sample 2" refers to the configuration where the country is the only country using either Tit-For-Tat or Friedman. Sample 2 will be

described using a single character plus three digits, e.g. F101. The letter will be either T or F, and tells us whether the country is using Tit-For-Tat or Friedman, where Tit-For-Tat configurations will be prefixed with the T, and Friedman will be prefixed with the F. The digits have no importance, other than that the different simulations using e.g. Tit-For-Tat is numbered sequentially from 1 to 7. F101 tells us that this is a configuration where the country at hand is using the Friedman strategy, and all the other countries are using the Dummy strategy.

6.2.2.1 Austria

The results for Austria are presented in Table 6.4.

Table 6.4: Effect on performance: Austria

(a) Paired Samples Statistics

	Supply centres			Score		
Strategy	**Mean**	**N**	**Std. Deviation**	**Mean**	**N**	**Std. Deviation**
Dummy	4.750	20	2.049	201.000	20	66.961
TFT	3.750	20	1.386	212.450	20	64.689
Friedman	2.650	20	1.996	198.800	20	67.777

Configuration		Supply centres comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T105	1.000	−1.583	0.113	Pos. ranks
All Dummy	F101	2.100	−3.004	0.003	Pos. ranks

(b) Paired Samples Test: supply centers

Configuration		Score comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T105	−11.450	−0.616	0.538	Neg. ranks
All Dummy	F101	2.200	−0.149	0.881	Neg. ranks

(c) Paired Samples Test: score

Austria: supply centres

We were able to find a statistical significance for the Friedman strategy, but not for Tit-For-Tat. Austria running the Tit-For-Tat strategy gave $Z = -1.583$, $p = 0.113$ based on positive ranks. The mean value for Tit-For-Tat indicates that Austria gains fewer supply centres with Tit-For-Tat than with Dummy, but the difference is not significant.

Austria running Friedman gave $Z = -3.004$, $p = 0.003$ based on positive ranks. Due to the direction of the mean we can conclude that there was a statistically significant decrease in the number of supply centres for Austria using the Friedman strategy compared to Austria using the Dummy strategy, from $4.75 + / - 2.049$ to $2.650 + / - 1.996$, a decrease of 2.1.

Austria: score

We were unable to find any significance for neither Tit-For-Tat nor Friedman. Austria running Tit-For-Tat gave $Z = -0.616$, $p = 0.538$ based on negative ranks. The mean value for TFT indicates that Austria performs better with Tit-For-Tat than with Dummy in terms of score, but the difference is not significant.

Austria running Friedman gave $Z = -0.149$, $p = 0.881$ based on negative ranks. The mean value for Friedman indicates that Austria using the Friedman strategy performs worse than Austria using the Dummy strategy in terms of score, but the difference is not significant.

6.2.2.2 England

The results for England are presented in Table 6.5.

England: supply centres

We were unable to find any significance for neither Tit-For-Tat nor Friedman. England running Tit-For-Tat gave $Z = -1.780$, $p = 0.075$ based on positive ranks. The mean value for Tit-For-Tat indicates that England gains fewer supply centres with Tit-For-Tat than with Dummy, but the difference is not significant.

England running Friedman gave $Z = -1.834$, $p = 0.067$ based on positive ranks. The mean value for Friedman indicates that England gains fewer supply centres with Friedman than with Dummy, but the difference is not significant.

England: score

We were unable to find any significance for neither Tit-For-Tat nor Friedman. England running Tit-For-Tat gave $Z = -1.493$, $p = 0.135$ based on negative ranks. The mean value for TFT indicates that England performs better with Tit-For-Tat than with Dummy in terms of score, but the difference is not significant.

Table 6.5: Effects on performance: England

(a) Paired Samples Statistics

	Supply centres			Score		
Strategy	**Mean**	**N**	**Std. Deviation**	**Mean**	**N**	**Std. Deviation**
Dummy	4.900	20	1.619	197.150	20	57.860
TFT	4.100	20	1.020	221.300	20	61.434
Friedman	3.900	20	1.651	206.800	20	54.878

Configuration		Supply centres comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T103	0.800	−1.780	0.075	Pos. ranks
All Dummy	F102	1.00	−1.834	0.067	Pos. ranks

(b) Paired Samples Test: supply centers

Configuration		Score comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T103	−24.150	−1.493	0.135	Neg. rank
All Dummy	F102	−9.650	−0.411	0.681	Neg. rank

(c) Paired Samples Test: score

England running Friedman gave $Z = -0.411$, $p = 0.681$ based on negative ranks. The mean value for Friedman indicates that England using the Friedman strategy performs better than England using the Dummy strategy in terms of score, but the difference is not significant.

6.2.2.3 France

The results for France are presented in Table 6.6.

France: supply centres

We were unable to find any significance for neither Tit-For-Tat nor Friedman. France running Tit-For-Tat gave $Z = -0.088$, $p = 0.930$ based on positive ranks. The mean value for Tit-For-Tat indicates that France gains fewer supply centres with Tit-For-Tat than with Dummy, but the difference is not significant.

France running Friedman gave $Z = -0.878$, $p = 0.380$ based on positive ranks. The mean value for Friedman indicates that France gains fewer supply centres with Friedman than with Dummy, but the difference is not significant.

Table 6.6: Effect on performance: France

(a) Paired Samples Statistics

	Supply centres			Score		
Strategy	**Mean**	**N**	**Std. Deviation**	**Mean**	**N**	**Std. Deviation**
Dummy	4.150	20	2.254	225.350	20	70.906
TFT	4.050	20	2.762	214.350	20	70.434
Friedman	3.650	20	2.084	238.800	20	40.984

Configuration		Supply centres comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T101	0.100	-0.088	0.930	Pos. ranks
All Dummy	F103	0.500	−0.878	0.380	Pos. ranks

(b) Paired Samples Test: supply centers

Configuration		Score comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T101	11.000	−0.299	0.765	Pos. ranks
All Dummy	F103	−13.450	-0.766	0.444	Neg. ranks

(c) Paired Samples Test: score

France: score

We were unable to find any significance for neither Tit-For-Tat nor Friedman. France running Tit-For-Tat gave $Z = -0.299$, $p = 0.765$ based on positive ranks. The mean value for TFT indicates that France performs worse with Tit-For-Tat than with Dummy in terms of score, but the difference is not significant.

France running Friedman gave $Z = -0.766$, $p = 0.444$ based on negative ranks. The mean value for Friedman indicates that France using the Friedman strategy performs better than France using the Dummy strategy in terms of score, but the difference is not significant.

6.2.2.4 Germany

The results for Germany are presented in Table 6.7.

Table 6.7: Effect on performance: Germany

(a) Paired Samples Statistics

	Supply centres			Score		
Strategy	**Mean**	**N**	**Std. Deviation**	**Mean**	**N**	**Std. Deviation**
Dummy	7.450	20	2.305	377.000	20	99.992
TFT	4.550	20	2.258	338.250	20	81.977
Friedman	3.100	20	2.149	298.800	20	73.576

Configuration		Supply centres comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T102	2.900	−2.929	0.003	Pos. ranks
All Dummy	F104	4.350	−3.731	0.000	Pos. ranks

(b) Paired Samples Test: supply centers

Configuration		Score comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T102	38.750	−1.979	0.048	Pos. ranks
All Dummy	F104	78.200	−2.837	0.005	Pos. ranks

(c) Paired Samples Test: score

Germany: supply centres

We were able to find a statistical significance for both Tit-For-Tat and Friedman. Germany running Tit-For-Tat gave $Z = -2.929$, $p = 0.003$ based on positive ranks. Due to the direction of the mean we can conclude that there was a statistically significant decrease in the number of supply centres for Germany using the Tit-For-Tat strategy compared to Germany using the Dummy strategy, from $7.450 + / - 2.305$ to $4.550 + / - 2.258$, a decrease of 2.9.

Germany running Friedman gave $Z = -3.731$, $p = 0.000$ based on positive ranks. Due to the direction of the mean we can conclude that there was a statistically significant decrease in the number of supply centres for Germany using the Friedman strategy compared to Germany using the Dummy strategy, from $7.450 + / - 2.305$ to $3.100 + / - 2.149$, a decrease of 4.3.

Germany: score

We were able to find a statistical significance for both the Friedman and Tit-For-Tat strategy. Germany running Tit-For-Tat gave $Z = -1.979$, $p = 0.048$ based on positive

ranks. Due to the direction of the mean we can conclude that there is a statistically significant reduction in performance in terms of score for Germany using the Tit-For-Tat strategy compared to Germany using the Dummy strategy, from $377.000 + / - 99.992$ to $338.250 + / - 81.977$, a decrease of 38.75.

Germany running Friedman gave $Z = -2.837$, $p = 0.005$ based on positive ranks. Due to the direction of the mean we can conclude that there is a statistically significant reduction in performance in terms of score for Germany using the Friedman strategy compared to Germany using the Dummy strategy, from $377.000 + / - 99.992$ to $298.800 + / - 73.576$, a decrease of 78.2.

6.2.2.5 Italy

The results for Italy are presented in Table 6.8.

Table 6.8: Effect on performance: Italy

	Supply centres			Score		
Strategy	**Mean**	**N**	**Std. Deviation**	**Mean**	**N**	**Std. Deviation**
Dummy	4.150	20	1.843	200.300	20	63.952
TFT	4.600	20	2.010	228.250	20	55.45778
Friedman	4.700	20	1.592	207.700	20	40.694

(a) Paired Samples Statistics

Configuration		Supply centres comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T107	−0.450	−0.790	0.430	Neg. ranks
All Dummy	F105	−0.550	−0.835	0.404	Neg. ranks

(b) Paired Samples Test: supply centers

Configuration		Score comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T107	−27.950	−1.736	0.083	Neg. ranks
All Dummy	F105	−7.400	−1.307	0.191	Neg. ranks

(c) Paired Samples Test: score

Italy: supply centres

We were unable to find any significance for neither Tit-For-Tat nor Friedman. Italy running Tit-For-Tat gave $Z = -0.790$, $p = 0.430$ based on negative ranks. The mean

value for Tit-For-Tat indicates that Italy gains more supply centres with Tit-For-Tat than with Dummy, but the difference is not significant.

Italy running Friedman gave $Z = -0.835$, $p = 0.404$ based on negative ranks. The mean value for Friedman indicates that Italy gains more supply centres with Friedman than with Dummy, but the difference is not significant.

Italy: score

We were unable to find any significance for neither Tit-For-Tat nor Friedman. Italy running Tit-For-Tat gave $Z = -1.736$, $p = 0.083$ based on negative ranks. The mean value for TFT indicates that Italy performs better with Tit-For-Tat than with Dummy in terms of score, but the difference is not significant.

Italy running Friedman gave $Z = -1.307$, $p = 0.191$ based on positive ranks. The mean value for Friedman indicates that Italy using the Friedman strategy performs better than Italy using the Dummy strategy in terms of score, but the difference is not significant.

6.2.2.6 Russia

The results for Russia are presented in Table 6.9.

Russia: supply centres

We were able to find a statistical significance for the Tit-For-Tat strategy, but not for Friedman. Russia running the Tit-For-Tat strategy gave $Z = -2.059$, $p = 0.039$ based on negative ranks. Due to the direction of the mean value we can conclude that there was a statistically significant increase in the number of supply centres for Russia using the Tit-For-Tat strategy compared to Russia using the Dummy strategy, from $2.500 + / - 1.433$ to $3.650 + / - 1.899$, an increase of 1.15.

Russia running Friedman gave $Z = -1.104$, $p = 0.270$ based on positive ranks. The mean value for Friedman indicates that Russia gains fewer supply centres with Friedman than with Dummy, but the difference is not significant.

Russia: score

We were able to find a statistical significance for the Tit-For-Tat strategy, but not for Friedman. Russia running Tit-For-Tat gave $Z = -2.240$, $p = 0.025$ based on negative ranks. Due to the direction of the mean we can conclude that there is a statistically significant increase in performance in terms of score for Russia using the Tit-For-Tat

Table 6.9: Effect on performance: Russia

(a) Paired Samples Statistics

	Supply centres			Score		
Strategy	**Mean**	**N**	**Std. Deviation**	**Mean**	**N**	**Std. Deviation**
Dummy	2.500	20	1.433	180.350	20	66.725
TFT	3.650	20	1.899	228.700	20	67.943
Friedman	2.100	20	1.916	212.550	20	67.950

Configuration		Supply centres comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T104	−1.150	−2.059	0.039	Neg. ranks
All Dummy	F106	0.400	−1.104	0.270	Pos. ranks

(b) Paired Samples Test: supply centers

Configuration		Score comparison			
Sample 1	**Sample 2**	**Mean difference**	**Z**	**p-value**	**Based on**
All Dummy	T104	−48.350	−2.240	0.025	Neg. ranks
All Dummy	F106	−32.200	−1.475	0.140	Neg. ranks

(c) Paired Samples Test: score

strategy compared to Russia using the Dummy strategy, from $180.350 + / - 66.752$ to $228.700 + / - 67.943$, an increase of 48.350.

Russia running Friedman gave $Z = -1.475$, $p = 0.140$ based on negative ranks. The mean value for Friedman indicates that Russia using the Friedman strategy performs better than Russia using the Dummy strategy in terms of score, but the difference is not significant.

6.2.2.7 Turkey

The results for Turkey are presented in Table 6.10.

Turkey: supply centres

We were unable to find any significance for neither Tit-For-Tat nor Friedman. Turkey running Tit-For-Tat gave $Z = -1.331$, $p = 0.183$ based on negative ranks. The mean value for Tit-For-Tat indicates that Turkey gains more supply centres with Tit-For-Tat than with Dummy, but the difference is not significant.

Table 6.10: Effect on performance: Turkey

(a) Paired Samples Statistics

	Supply centres			Score		
Strategy	**Mean**	**N**	**Std. Deviation**	**Mean**	**N**	**Std. Deviation**
Dummy	5.900	20	1.483	110.050	20	51.786
TFT	6.450	20	1.848	164.550	20	62.146
Friedman	5.600	20	1.846	158.500	20	48.866

Configuration		Supply centres comparison			
Sample 1	**Sample 2**	**Mean difference**	***Z***	**p-value**	**Based on**
All Dummy	T106	−0.550	−1.331	0.183	Neg. ranks
All Dummy	F107	0.300	−0.373	0.709	Pos. ranks

(b) Paired Samples Test: supply centers

Configuration		Score comparison			
Sample 1	**Sample 2**	**Mean difference**	***Z***	**p-value**	**Based on**
All Dummy	T106	−54.500	−2.389	0.017	Neg. ranks
All Dummy	F107	−48.450	−2.576	0.010	Neg. ranks

(c) Paired Samples Test: score

Turkey running Friedman gave $Z = -0.373$, $p = 0.709$ based on positive ranks. The mean value for Friedman indicates that Turkey gains fewer supply centres with Friedman than with Dummy, but the difference is not significant.

Turkey: score

We were able to find a statistical significance for both Tit-For-Tat and Friedman. Turkey running Tit-For-Tat gave $Z = -2.389$, $p = 0.017$ based on negative ranks. Due to the direction of the mean we can conclude that there is a statistically significant increase in performance in terms of score for Turkey using the Tit-For-Tat strategy compared to Turkey using the Dummy strategy, from $110.050 +/- 51.786$ to $164.550 +/- 62.146$, an increase of 54.5.

Turkey running Friedman gave $Z = -2.576$, $p = 0.010$ based on negative ranks. Due to the direction of the mean we can conclude that there is a statistically significant increase in performance in terms of score for Turkey using the Friedman strategy compared to Turkey using the Dummy strategy, from $110.500 +/- 51.786$ to $158.500 +/- 48.866$, an increase of 48.45.

6.2.2.8 Summary

Table 6.11 and Table 6.12 summarise the findings in individual differences. The tables show that both Friedman and Tit-For-Tat gave very few significant results. The low number might indicate that the strategies simply do not affect the agent all that much, but there is also a possibility that the results had been different if the data used for the analysis (20 simulations for each configuration) had been greater.

Table 6.11: Effect on performance: Supply centers

Strategy	Better	Worse	Sig. better	Sig. worse	Total sig.	Total
Single TFT	3	4	1	1	2	7
Single Friedman	1	6	0	2	2	7

Table 6.12: Effect on performance: Score

Strategy	Better	Worse	Sig. better	Sig. worse	Total sig.	Total
Single TFT	6	1	2	1	3	7
Single Friedman	5	2	1	1	2	7

Supply centres

For supply centres, 2/7 countries showed significant effect of the strategy when using **Tit-For-Tat**. 1/7 performed better and 1/7 performed worse. Looking past statistical significance, 3/7 of the countries indicated an improvement in performance, while 4/7 of the countries indicated a decrease in performance. Based on these findings we are unable to give any clear conclusions for the effect of Tit-For-Tat on the performance of the agent in terms of supply centres.

For the countries using **Friedman**, 2/7 countries showed significant effect of the strategy. Both the countries showing significant effect from Friedman performed worse, indicating that Friedman decreases the performance of the agent in terms of supply centres. The countries performing worse, Austria and Germany, are both surrounded by enemies on all fronts, and we assume that the reason for Friedman's decrease in performance is related to the fact that once Friedman is defected against, it will become a very aggressive strategy, leading to a lack of protection for the home provinces.

Score

3/7 countries using **Tit-For-Tat** showed a significant effect of the strategy. 2/7 showed a significant increase in performance in terms of score, while 1/7 countries showed

a statistical significant decrease in performance. Looking past significance, 6/7 countries indicated an increase in performance, while 1/7 countries indicated that they performed worse. Looking past significance, the mean value indicate that 6 of the 7 countries performs better in terms of Prisoner's Dilemma score using TFT compared to the dummy strategy. Based on these findings, we conclude that in terms of score, the Tit-For-Tat strategy improves the performance of the agent.

For the countries using the Friedman strategy, 2/7 countries showed significant effect of the strategy. 1/7 performed better and 1/7 worse. These findings are too ambiguous to make any clear conclusions to the effect of the Friedman strategy in terms of score, but if we look at the total number of countries reporting an increase in performance (5/7) compared to the total number of countries reporting a decrease in performance (2/7) we get a clear indication that Friedman increases the performance of the agent in terms of score.

6.3 User testing

To evaluate the third research question, "Can players make a distinction between StateCraft with and without the Prisoner's Dilemma module?", we gathered six persons to participate in a user testing of the game. Every participant received a document that explained the purpose of the test, and they were given a short introduction to Diplomacy, StateCraft and the user interface of StateCraft before they started their tests. The introduction was performed partly by us, and partly by letting the participants try for themselves.

Five of the participants were male, one was female[1]. All participants were in their twenties and were pursuing higher education in computers. Five of the participants were of Norwegian nationality while one was Spanish. Three of the participants had played Diplomacy and / or StateCraft before, while three had no experience with either game. All participants had played computer games before, but not everyone had experience with computerised war games (one participant stated that his only prior experience with computer games was limited to games such as Othello, Patience and The Sims).

Each participant was asked to play two games: *Game A* where all the opponents were controlled by agents using the Dummy strategy, and *Game B* where three of the agents used Tit-For-Tat and three used the Friedman strategy. Both games lasted from Spring 1901 until Spring 1905. The sequence the games were played in was mixed - three of the participants played Game A before Game B, and three of the participants played Game B before Game A.

[1]Note that all participants will be referred to using the male gender to obfuscate the gender of the participants.

After playing the games, they were asked to answer three questions:

1. did they observe any differences between the two games
2. which of the two games was more fun
3. which of the opponents used which strategy in Game B

6.3.1 Findings

Differences between the two games were easy to identify for most of the participants. Many reported that the agent was more aggressive in Game A, something which is reflected by the higher average number of supply centres the participants was able to gain in Game B. Table 6.13 lists the number of supply centres the different participants was able to gain by Spring 1905 for both games played. The average number of supply centres gained by Spring 1905 for Game A is $4,5$ compared to the average of 6.0 for Game B. This is a good indicator that the introduction of Tit-For-Tat and Friedman makes the agent less aggressive.

However, not everyone reported that the agents were more aggressive in Game B. One of our participants stated that "*The countries were better in Game B. More aggressive*". Three users stated that they found specific countries to be more aggressive in Game B, something which might be the result of a defection performed by the player, which would lead both Tit-For-Tat and Friedman to retaliate by attacking the player. However, all countries mentioned as more aggressive in Game B were countries directly adjacent to the user, something we think might be caused by limited visibility in the game by inexperienced users (i.e. players might not be able to interpret the actions performed on the other side of the map as they are concentrating on the actions performed in close proximity to the players' units).

All participants reported that they found some difference between Game A and Game B. Most comments referred to the aggressiveness of the agent, either that some country was acting more aggressive in one of the games, or that the game in general was more aggressive, something we interpret as that all the countries or some unspecified set of countries were acting more aggressive.

When asked which of the games they preferred half of the participants preferred Game A, and half of the participants preferred Game B.

A general trend seems to be that those that preferred Game A also described it as more challenging. This corresponds to the performance of the users preferring Game A, as all the users preferring Game A also gained a lower number of supply centres in Game A

Table 6.13: Participant performance and preferred game

Participant	Previous experience	Number of supply centres in Game A	Number of supply centres in Game B	Most fun
Participant 1	No	6	7	Game B
Participant 2	Yes	4	6	Game A
Participant 3	Yes	8	4	Game B
Participant 4	No	4	7	Game B
Participant 5	Yes	1	7	Game A
Participant 6	No	4	5	Game A
Average	-	4.5	6.0	-

compared to Game B. Although not very clear, this could indicate that the users preferring Game A found it more fun because it was more challenging.

One participant, Participant 2, reported that he preferred Game A because it was easier than Game B, even though he performed worse in Game A than in Game B. This seems strange, but might be explained by his comment about "stupid" actions performed by the agents: "*In Game A, they kept doing the same stuff all over again without support*". Scott (2002) argues that AI developers sometimes should focus on creating the *Illusion of intelligence* rather than making the AI always perform the heuristically determined best moves. Maybe an agent which keeps repeating its actions, a very non-human like behaviour, might give the users the impression that the agent is actually stupid and thus a less worthy opponent.

Participant 1 and Participant 3, two of the participants that preferred Game B over Game A, both played Game A before Game B. Participant 1 did not give any reason for his preference, while Participant 3 stated that Game B contained more excitement and fewer situations where the agents performed the same action over and over, resulting in "never-ending" standoff situations. However, due to the small data set from the user testing we cannot rule out that the preference might be caused by a better understanding of StateCraft due to more experience with the game after playing the same country in Game A.

The third participant who played Game B before Game A answered that B was more fun because he "*got to rule the world*", indicating that at least for this participant, the game was more fun because the agent was performing worse.

When asked to identify which strategies the agents used few users managed to identify which strategy the different countries were using. Table 6.15a shows the participants' take on which strategy each country was using, where the numbers represents how

many users that thought the country was using that particular strategy, e.g. two users thought England was using TFT. Table 6.15b shows which strategy the different country was actually using.

Table 6.14: Strategy configuration and participants' guesses

Country	TFT	Friedman	None
England	2	2	2
Turkey	3	2	1
France	1	1	4
Italy	2		4
Austria	1	2	3
Germany	3	1	2

(a) Participants' take on which strategy each country was using

Country	TFT	Friedman	None
England	x		
Turkey		x	
France	x		
Italy		x	
Austria		x	
Germany	x		

(b) Strategy actually used by the different countries

As the tables show, England, Turkey and Austria was identified correctly by 33.3% of the participants, France by 16.7%, Italy by 0%, and Germany by 50%. This tells us that it is hard to identify which strategy each country is using, something that corresponds with the written feedback from the users: they really didn't know what was going on other than that some countries acted more or less aggressive than other countries.

This isn't all that surprising. We suspect that users need both much experience with the old agent in StateCraft as well as thorough knowledge about the strategies to be able to separate Tit-For-Tat and Friedman from the Dummy strategy. There is also the possibility that due to the short duration of the games (four years), the users did not have time to study the two games as thoroughly as needed to find any substantial or pronounced differences between the different agents, and thus could not identify the different strategies correctly.

6.3.2 Summary

The results from the user tests are inconclusive. All the participants reported some difference between the two games, either that Game A or some of the agents in Game A was more aggressive than the ones in Game B, or vice versa, that Game B or some of the agents in Game B was more aggressive than the ones in Game A.

We were unable to determine which game the users preferred. Half the participants preferred Game A (without other strategies than the Dummy, which only performs logging), while half the participants preferred Game B.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

The main objective of the thesis was to study whether we could improve the player's experience by modifying the autonomous agent in StateCraft. The modifications were implemented as two separate modules, the Emotion module and the Prisoners's Dilemma module. We also wished to study how our modifications affected the agent's performance in terms of supply centers, and whether or not strategies used for the Prisoner's Dilemma could be used in a n-player game such as StateCraft.

The Emotion module tries to improve the user's experience of the game by giving the autonomous agent in StateCraft emotions. The emotions affect the decision making of the agent with the intention of making it appear more human-like.

The Prisoner's Dilemma module was created to give the autonomous agent in StateCraft an understanding of the concept of the Prisoner's Dilemma. StateCraft contains several situations similar to the ones expressed in the Prisoner's Dilemma, and we wished to study whether the agent would perform differently if it understood this concept. The module was implemented with two strategies for the Prisoner's Dilemma module, Tit-For-Tat and Freidman, which both modify the behaviour of the agent according to the original specification of the strategies.

7.1.1 Performance

The effects the Emotion module and the Prisoner's Dilemma module had on agent performance were measured using statistical analysis on data collected from simulations of the game using different configurations.

7.1.1.1 Emotion module

The results show that the Emotion module decreases the performance of the agent in terms of supply centres in general. We were unable to find significant differences between the regular agent and the agent using the Emotion module for all countries except for Germany. Germany performs significantly worse with emotions. Due to its close proximity to neighbouring enemies, it becomes vulnerable if it takes unnecessary risks in the game. The feeling of joy leads to the agent taking risky decisions, and because Germany will most likely occupy the neutral provinces in the early rounds of the game, which leads to the feeling of joy, it becomes joyful and perform bold moves.

However, we cannot conclude that the agent performs worse using emotions because we have so few significant findings. This could be caused by the limited amount of data we collected during the data simulations, and we suggest that given enough data, more countries would give significant results.

7.1.1.2 Prisoner's Dilemma module

The results for the Prisoner's Dilemma are two-fold. For Tit-For-Tat, we found a significant effect of the strategy for two countries. One performed better, while the other performed worse. Because of these results, we cannot conclude either way. Looking past significance, we find that four of the seven countries showed a decrease in performance based on mean values, while three of the seven countries showed an increase in performance using Tit-For-Tat. Needless to say, these results are too ambiguous to make any clear conclusions.

For Freidman, the results are clearer. Two of the seven countries performed worse using Freidman. These countries, Austria and Germany, are both surrounded by enemies on all fronts, and we assume that the reason for Freidman's decrease in performance is related to the fact that once Friedman is defected against, it will become a very aggressive strategy, leading to a lack of protection for the home provinces.

However, if all the agents are running either Freidman or Tit-For-Tat, the results are quite different. What we found was that in a game where all the agents were running either the Tit-For-Tat or the Freidman strategy, all the agents would perform more similar or *balanced* compared to a game where all the agents were using the Dummy strategy. Therefore, we conclude that Tit-For-Tat and Freidman will lead to a more balanced game if all the agents are using the same strategy.

7.1.1.3 2- and 3-player IPD strategies in n-player game

The effects the Prisoner's Dilemma module had on performance in terms of Prisoner's Dilemma score were measured using statistical analysis on data collected by running simulations of the game. The results for Tit-For-Tat show that 3/7 countries had a significant effect of the strategy, where 2/7 countries increased their score, and 1/7 decreased their score. Looking past significance, 6/7 countries perform better in terms of score when using the TFT strategy.

For Freidman, the results are somewhat more ambiguous, but still give a clear indication. 2/7 countries performed significantly different when using the Freidman strategy compared to using the regular agent. 1/7 performed worse, 1/7 performed better. In terms of mean value, 5/7 countries performs better using the Freidman strategy.

Based on these results we conclude that both Tit-For-Tat and the Freidman strategy are suitable for the domain of StateCraft.

7.1.2 User experience

To evaluate the user experience we conducted a user test where the participants were asked to play StateCraft with and without our modifications enabled. The users were not informed whether or not the modifications were present during the games, which means that they were participating in a blind experiment. They were then asked a series of questions, tailored with the intention of studying whether the user experience increased or decreased as a result of our modifications.

7.1.2.1 Emotion module

When asked to identify the agents' internal emotional states, the participants were unable to correctly identify the emotions. There may be several reasons for this.

1. The time span used for testing was too short
2. The interface between the agent and the user is too limited
3. StateCraft provides the players with a limited set of actions (e.g. move, hold, support and convoy)
4. The users have limited knowledge about emotions and how they affect decisions

When asked which of the games were most fun, 4/5 of the players who noticed a difference[1] between the two games found it more enjoyable to play against the emotional agents. However, the players also performed much better against the emotional agents, and because of this we cannot conclude whether the participants had more fun as a result of the agent being emotional or as a result of the agent performing worse.

Based on the limited study performed we cannot make a clear conclusion on the user experience, but the results *indicate* that the Emotion module increases the user experience.

7.1.2.2 Prisoner's Dilemma module

The results from the user testing for the Prisoner's Dilemma module were unclear.

When asked to identify differences between the two games, all the participants reported some differences between the game where all opponents were controlled by regular agents, and the game where three of the opponents used the Freidman strategy and three of the opponents used the Tit-For-Tat strategy. However, the differences reported by the users were most often related to a more aggressive style by the regular agent, indicating that the user perceives the Prisoner's Dilemma as a defensive addition to the agent. When asked which strategy the different countries were using, the users were unable to identify the strategies. This leads us to believe that the participants' limited knowledge of both the original agent and how the strategies affect the agent is a limiting factor when it comes to identify this particular addition, and that the time span used for testing the two games was much too short. We conclude that the users are able to distinguish the two games, but in most cases unable to identify which strategy the different countries are utilising.

The participants were divided in terms of which game they preferred. 3/6 users preferred the game using the regular agent, while 3/6 users preferred the game with the Prisoner's Dilemma module. Because of this, we cannot conclude either way, suggesting that more research is required.

7.2 Future work

7.2.1 TacticTree

The TacticTree was implemented to combat performance issues related to generating tactics for the agent in StateCraft. It succeeded in doing so, but there is still room for improvement. The agent is still having difficulties in coping with more than 11 units, because the number of possible tactics is very large. We therefore propose that future

[1]Not all users were able to identify differences between the emotional agent and the regular agent

implementations of the TacticTree should have an upper limit of for example 1000 tactics. However, since the current TacticTree is generated by an algorithm utilising a depth-first approach, it is difficult to implement such a limit without diminishing heterogeneity among the tactics. The reason is that such a limit will potentially exclude the entire right side of the TacticTree if the number of potential legal tactics is far greater than the limit, thus reducing the variance. Therefore, we propose to generate the TacticTree by using a breadth-first approach. This would make it possible to discard for example 80% of the tactics without necessarily reducing variance.

7.2.2 Emotion module

In order to obtain more significant differences between the emotional agent and the regular agent, we need to run more data simulations, especially for countries such as Russia, where it seems likely that we would obtain a significant difference.

As our results from the user testing only gave indications that playing against emotional agents is more fun, we suggest that it should be tested on both more experienced users and users in general. It would also be interesting to research if the preference of playing against emotional agents is a result of the agents performing worse, or a result of the agents appearing more human-like.

We have concluded that it is difficult for the agents in StateCraft to express their inner emotional state through actions only. Therefore, we propose using emoticons[2] for expressing the agents' emotional state towards the player (e.g. when the agent controlling Russia is angry at the player, an icon of an angry face would appear next to the flag of Russia) and study whether this improves the user's game experience.

7.2.3 Prisoner's Dilemma

The most obvious addition to the Prisoner's Dilemma module is the inclusion of other strategies. Several strategies have proved to be good for the Prisoner's Dilemma, and we have only implemented two of them in this thesis.

We defined dilemma provinces as empty provinces adjacent to provinces with supply centres, where two or more units were able to invade the province at the same time. However, one can argue that dilemma provinces could be defined differently. Although our preliminary testing has not provided us with conclusive results , it could be interesting to perform extensive testing in a game where the dilemma provinces are defined as unoccupied provinces with supply centres, where two or more enemies are adjacent and able to

[2]An emoticon here refers to a graphical icon representing the agent's facial expression or mood

invade. Therefore, we propose that such a test would be a good topic for further research on Prisoner's Dilemma in StateCraft.

Additionally, because of the somewhat vague results from both the user testing and the analysis performed on the data collected from the simulations, we believe that there would be more significant results if we had had more data to run the analysis on, and more experienced users to perform the user testing. Hence we propose more testing and analysis of the current Prisoner's Dilemma module as a good focus for further research.

Bibliography

Axelrod, R. (2006). *The Evolution of Cooperation: Revised Edition*. Basic Books.

Axelrod, R. and Hamilton, W. (1981). The evolution of cooperation. *Science*, 211(4489):1390.

Bates, J. (1994). The role of emotion in believable agents. *Communications of the ACM*, 37(7):122–125. The role of emotion in believable agents.

Berkowitz, L. (1990). On the formation and regulation of anger and aggression: A cognitive-neoassociationistic analysis. *American Psychologist*, 45(4):494–503.

Brooks, R. (1990). Elephants don't play chess. *Robotics and Autonomous Systems*, 6(1&2):3–15.

Conway, M. and Bekerian, D. (1987). Situational knowledge and emotions. *Cognition & Emotion*, 1(2):145–191.

Copeland, J. (2000). A brief history of computing.

Damasio, A. (1994). *Descartes' error: Emotion, reason, and the human brain*. Papermac London. Descartes' error: Emotion, reason, and the human brain.

El-Nasr, M. and Skubic, M. (1998). A fuzzy emotional agent for decision-making in a mobile robot. *A & M University Press, Texas, US*.

Freedman, J., Wallington, S., and Bless, E. (1967). Compliance without pressure: The effect of guilt. *Journal of Personality and Social Psychology*, 7:2–Pt.

Frijda, N. (1987). Comment on oatley and johnson-laird's towards a cognitive theory of emotions. *Cognition & Emotion*, 1(1):51–58.

Grønmo, S. (2007). *Samfunnsvitenskapelige metoder*. Fagbokforlaget.

Harmon-Jones, E., Sigelman, J., Bohlig, A., and Harmon-Jones, C. (2003). Anger, coping, and frontal cortical activity: The effect of coping potential on anger-induced left frontal activity. *Cognition & Emotion*, 17(1):1–24.

Helgesen, A. S. and Krzywinski, A. (2006). The caeneus architecture: An agent architecture for social board games.

Kleiner, A. (2005). Game ai: The shrinking gap between computer games and ai systems. *Ambient Intelligence: The Evolution of Technology, Communication and Cognition Towards the Future of Human-computer Interaction*, 6:143–155.

Kleinginna, P. and Kleinginna, A. (1981). A categorized list of emotion definitions, with suggestions for a consensual definition. *Motivation and emotion*, 5(4):345–379.

Koda, T. and Maes, P. (1996). Agents with faces: The effect of personification. Citeseer.

Lerner, J. and Keltner, D. (2001). Fear, anger, and risk. *Journal of Personality and Social Psychology*, 81(1):146–159.

Lerner, J. S. and Tiedens, L. Z. (2006). Portrait of the angry decision maker: How appraisal tendencies shape anger's influence on cognition. *Journal of Behavioral Decision Making*, 19(2):115–137. English.

Loewenstein, G. and Lerner, J. (2003). The role of affect in decision making. *Handbook of affective science*, pages 619–642.

Luger, G. (2004). *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*. Addison Wesley.

Maria, K. A. and Zitar, R. A. (2007). Emotional agents: A modeling and an application. *Information and Software Technology*, 49(7):695–716.

Matsushima, M. and Ikegami, T. (1998). Evolution of strategies in the three-person iterated prisoner's dilemma game. *Journal of theoretical biology*, 195(1):53–67. Evolution of Strategies in the three-person Iterated Prisoner's Dilemma Game.

McCarthy, J. (2007). What is artificial intelligence? Access Date.

Minsky, M. (1988). *The society of mind*. Simon and Schuster.

Mowrer, O. (1960). *Learning theory and behavior*. Wiley New York.

Newborn, M. (2003). *Deep Blue: an artificial intelligence milestone*. Springer. Deep Blue: an artificial intelligence milestone.

Ortony, A., Clore, G. L., and Collins, A. (1988). *The cognitive structure of emotions*. Cambridge University Press, Cambridge England ; New York.

Ortony, A. and Turner, T. J. (1990). What's basic about basic emotions? *Psychological Review*, 97(3):315–331. Institute for the Learning Sciences, Northwestern University, Evanston, Illinois 60201.

Picard, R. (2000). *Affective computing*. The MIT Press. Affective computing.

Russell, S. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Prentice Hall. 0130803022.

Scott, B. (2002). *AI Game Programming Wisdom*, chapter The Illusion of Intelligence, pages 19–20. Charles River Media.

Shapiro, S. and Wilk, M. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, 52(3-4):591.

Sloman, A. (1993). Why we need to study architectures. Why we need to study architectures.

Sloman, A. (1998). Architectural requirements for human-like agents both natural and artificial (what sorts of machines can love?). *School of Computer Science Research Reports-University of Birmingham CSR*.

Tozour, P. (2008). Why "ai accelerators" will never happen.

Wohlin, C., H
"ost, M., Runeson, P., Ohlsson, M., Regnell, B., and Wesslén, A. (2000). *Experimentation in software engineering: an introduction*. Kluwer Academic Pub.

Woodcock, S. (2000). Game ai: The state of the industry. Access Date.

Wooldridge, M. (2002). An introduction to multiagent systems. *West Sussex, England: John Wiley and Sons Ltd*, 348.

Appendix A

Emotions user questionnaire

Emotion user testing

Participant _____

Age_____

Nationality_____

Country played_____

The Emotion module is an addition to StateCraft which tries to "give" the agents emotions. First, you will play against some emotional agents and some regular agents (without emotion). Then you will be asked to identify which agents had emotions and which emotions they expressed.

Then you will be asked to play two different versions of the game. One is the old version with regular agents, and one is the new version against emotional agents.

---

1. Can you identify which agents had emotion and which emotions they felt?

Country/Emotion	Joy	Anger	Guilt	Admiration	Fear
England					
Turkey					
France					
Italy					
Austria					
Russia					
Germany					

2. Did you observere any difference between Game A and Game B?

   (a) Which differences did you find?

   (b) Do you think one of the games were more fun than the other? Elaborate.

Appendix B

Prisoner's Dilemma user questionnaire

Prisoner's Dilemma user testing

Participant _____

Age_____

Nationality_____

Country played_____

The Prisoner's Dilemma module is an addition to StateCraft which tries to give the computer player an understanding of cooperation and revenge, and includes two strategies that changes the game somewhat.

You will be asked to play two different versions of the game, Game A and Game B. One is the old version without the Prisoner's Dilemma module, and one is the new version using the Prisoner's Dilemma module.

There are two different strategies for the computer players in this game. Each of these strategies will make the players perform somewhat different:

- The Tit-For-Tat (TFT) strategy will, once "attacked", respond by attacking the attacker once, and then go on as if nothing happened
- The Freidman strategy will, once attacked, continue to attack every opponent in the game until it either wins the game or is killed

- In addition to these, the computer players can use the "no-strategy", which is, as the name implies, no strategy.

---

1. Did you observere any difference between Game A and Game B?

   (a) Which differences did you find? Elaborate

   (b) Was there any country that distinguished itself, or you felt acted differently from the rest? Elaborate.

2. Which was more fun, game A or Game B? Why?

3. Can you identify which country played which strategy (TFT, Freidman, none)?

Country/Emotion	**TFT**	**Freidman**	**None**
England			
Turkey			
France			
Italy			
Austria			
Russia			
Germany			