NTNU

Norwegian University of
Science and Technology

Case Based Reasoning and Adaptation for Warmachine

Tor Egil Fusdahl

Master of Science in Computer Science
Submission date: January 2017
Supervisor: Anders Kofod-Petersen, IDI
Co-supervisor: Jarle Svendsrud (BEKK), Bekk

Norwegian University of Science and Technology
Department of Computer Science

Abstract

The project that is detailed in this report was performed by a student at the Norwegian University of Science and Technology as the final part of a master's degree in Computer Science and Artificial Intelligence. It focuses on a Case-Based Reasoning system working with a miniature war game called Warmachine. Case based reasoning is a powerful method for computer reasoning that mimics human problem solving.

Earlier there have been two other master degrees from NTNU aimed at the war gaming domain and this project contributes with an analysis and introduction to a new one. It explores adaptation as a focal point for improving user queries, and attempts to compare the benefits and drawbacks of different techniques. Some of these techniques are implemented and tested, while others are rejected base on their limitations with regards to warmachine.

The systems performance is judged by an expert group of warmachine players who have extensive experience with the game. This allows for a less partial review and each adaptation technique is evaluated based on its proposed solutions. Finally, the overall performance is covered and some possible avenues for future research are examined.

Preface

This project report is the deliverable for a master's thesis written at the Norwegian University of Science and Technology (NTNU) in the fall of 2016. It is provided by the Department of Computer and Information Science.

I would like thank my supervisors Anders Kofod-Petersen and Jarle Svendsrud for their continuous feedback and help with the development of this project.

Finally, I would also like to thank the members of the expert group who helped provide feedback on the systems performance: Fredrik Raben, Serge Devriendt, Jarle Svendsrud and Christian Tellefsen.

Tor Egil Fusdahl
Trondheim, January 25, 2017

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This paper describes the architecture and implementation of a case-based reasoning (CBR) system targeted at the Warmachine domain. The system builds upon CBR fundamentals and seeks to explore case adaptation to create good and useful solutions for the end user.

CBR systems are based upon human recollection methods and have been a topic of AI research since the 80's. The method tries to leverage the knowledge learned from previous problems to propose solutions to new ones [Schank, 1982] and the area has seen a lot of progress over the years [Mantaras and et al, 2005]. Most CBR systems conform to a design consisting of a cyclic process with four steps:

1. *Retrieve* problems that are similar to the one the system is currently facing.

2. *Reuse* and adapt the old solution so it is capable of dealing with the new problem.

3. *Revise* the proposed solution to ensure that it works and that the user is satisfied.

4. *Retain* the new solution if it adds value to the system and it has performed well.

Adaptation has been a much researched topic in CBR and it is often considered one of the most complicated parts of a good CBR system [Patterons et al., 2002]. This project implements and experiments with transformational adaptation techniques that were originally conceived in the 90s when CBR was a fresh and new area of research. Adaptation is highly reliant on domain and implementation details and the different possibilities for a warmachine system are discussed in 2.2.3.

A system that is able to adapt and alter a user query in a deliberate fashion is capable of reasoning about its actions which is often considered to be one of the fundamental signs of intelligence [Sormo et al., 2005]. AI programs that are capable of reasoning around their choices show a targeted purpose and they can be expanded to create human readable explanations. These explanations can help humans gain new and valuable understandings in different domains.

An area where this has been showcased repeatedly is in competitive board games such as Chess and Go. Since the 90's when Deep Blue beat Kasparov chess computers have been used to study the game and they have helped human players improve immensely. Recently something similar happened in Go, a very popular board game in southeast Asia, when the AI AlphaGo beat one of the worlds best players. In the coming years the game is likely to evolve as the professional players learn from how AlphaGo plays the game. Another important aspect of systems that can reason about their choices is their ability to receive outside feedback and use this as a basis to alter their reasoning. This feedback loop is often an important tool for learning and it is handled by the reuse and revision parts of the CBR-cycle explained above.

1.1 Background and Motivation

The motivations for this project are founded both in theory and in practice. Adaptation is an integral part of the CBR cycle and it is often an essential part of capable systems. The main theoretical contributions of this project is implementing adaptation techniques in a previously unexplored domain and attempting to both quantify and compare the effects of different techniques.

The primary practical motivation is based on the authors experience from being a novice warmachine player. New warmachine players often have a hard time understanding what constitutes a good army and it is difficult to recognize how an army can be improved. This project attempts to alleviate this problem by creating a system that is able to improve unoptimized armies through adaptation.

Building Warmachine armies is also a topic of great personal interest for the author. It is a process that requires a great deal of tacit knowledge about the game. Building a CBR system that is able to understand and process this kind information in a satisfactory way would be very hard given the projects time frame. As a result the proposed CBR system relies on utilizing explicit knowledge and computational power when adapting armies.

1.2 Goals and Research Questions

This section covers the research questions and goals of the project and explains their reasoning and significance.

Goal *How can adaptation guide a CBR system that helps warmachine players develop army concepts into specific, refined armies.*

The main and overall goal is to implement a CBR system that can create and adapt army compositions for warmachines. This system is used to explore adaptation techniques to see if they are able to improve input armies in any meaningful way. This is a complicated task if the system focuses on optimized and competitive armies. However if the system tries to improve unrefined and imperfect army ideas instead there could be room for optimizations. To further explore and define this research goal there are three research questions that must be answered:

Research Question 1. *How do various adaptation techniques compare to each other with regards to the warmachine domain? Which ones produce the best armies according to an expert group?*

Adaptation techniques for cased-based reasoning systems perform differently for each domain they are used in depending on factors such as case-structure and query-structure. There is no ideal adaptation technique that is applicable to every domain. Some domains require complex adaptation procedure with a lot of general knowledge [Aamodt, 2004] while a more structural approach is sufficient in others. This project focuses on exploring less complicated techniques that might function without detailed domain knowledge. They are tested and compared to each other to see how they perform in the warmachine domain.

Each time an adaptation procedures creates an output army an expert group is asked to rate the army on a scale of 1 to 7, where 1 is the lowest and 7 is the highest. Research within AI should be focused on testing and performing concrete experiments. By assembling an expert group it is possible to conduct a relatively impartial review of the systems capabilities and compare the merits of the different adaptation techniques. These army evaluation are essential to answering the research questions.

Research Question 2 *If an army receives an average score of 3 or lower, what does the expert group think is the biggest weaknesses of the proposed army? Are there any systematical errors in the armies generated from a specific adaptation technique? Are there any mistakes that are frequent in all armies?*

One of the most interesting aspects of these research questions is identifying potential weaknesses that are prevalent in the output armies. Are certain weaknesses only common to one adaptation technique? Do all the proposed armies share some common problem, and how might this be improved?

Research Question 3 *Even if the system does not usually create armies that receive high scores (4 or higher) it might still be improving the user query. The expert group will be asked to rate the input army so they can be compared to the armies that are created by system.*

By evaluating the original user query it is possible to perform a more detailed analysis of the adaptation procedures. Are some adaptation procedures only capable of improving bad armies? Do some methods struggle to improve bad armies? How large are the improvements made by a technique?

1.3 Research Method

AI research is usually focused on building new systems and implementing novel ideas. This means that research methodology and rigid evaluation is often ignored in favour of an increased focus on system development [Cohen and Howe, 1988]. To ensure the project does not follow this pattern it uses the article "How Evalutaion Guides AI Reseach" [Cohen and Howe, 1988] as a foundation and builds upon the principles explained there. In the article Cohen and Howe identify five areas that should be considered when evaluating research within artificial intelligence:

1. Evaluating the Research Problems
2. Evaluating the method
3. Evaluating method implementation
4. Evaluating experiment design
5. Evaluating experiment results

To make sure that the research problems are sound it is important to consider if the task you are trying to solve is significant and if the research is likely to meaningfully contribute in any way. Specifically this project implements and evaluates adaptation techniques for CBR systems in an unexplored domain. This could prove valuable for other researches that are interested in miniature wargames, warmachine or the utilized adaptation techniques.

Both the theory and implementation of a new method needs to be evaluated. This project does not attempt to introduce any new techniques, but it has to customize the adaptation procedures that are evaluated. The methods needs to be tailored to the domain which means that there is a set of assumptions and implications tied to the implementation. These topics are explored further explained in chapter 3.

Once a system has been implemented it is important to consider the experiment design and its coverage. How many examples are covered? Are they different in important ways? Are the criteria for a good performance well defined and sensible? Evaluating CBR systems is a difficult task and the quality of an army is a highly subjective topic. To ensure an objective evaluation the use of an expert group is vital. This guarantees that the quality of the systems work is measured by outsiders that do not have a stake in the project. More details around the experiments is covered in chapter 4.

Finally, it is important to evaluate what the experiments actually tell us. This project focuses on performance by evaluating the strength of the generated armies and tries to highlight how well the different methods perform. The evaluation in chapter 5 attempts to answer the research questions and discusses both merits and limitations of the utilized adaptation techniques.

1.4 Thesis Structure

The first chapter of this report focused on introducing background information, clarifying the research goals of the project and discussed the importance of clear methodology when researching AI techniques.

Chapter 2 explains the theoretical background for CBR systems and covers several adaptation techniques. It also gives an introduction to Warmachine and some of the challenges that an AI situated in the domain must handle.

Chapter 3 presents the systems architecture, design and implementation: focusing on the CBR cycle and the adaptation process. It covers functional requirements for a warmachine CBR and provides insight into relevant frameworks. It highlights important implementation details for each step in the CBR cycle, with special attention given to retrieval and adaptation.

Chapter 4 covers the experimental plan for the project. It describes how the experiments are conducted and how the proposed armies are presented to the expert group. The chapter also presents the results of the experiments and explains how the experts provide feedback.

Chapter 5 focuses on what can be learned from the project and its implementation and execution. The system is evaluated and both strengths and weaknesses are discussed as well as some implications and potential avenues for future research.

Chapter 2

Background Theory and Motivation

This chapter introduces background theory clarifying important concepts in case-based reasoning and warmachine. Section 2.1 covers warmachine rules, army creation and some critical theory concepts and expressions. 2.2 looks at case-based reasoning systems in some detail, with special focus being given to adaptation. It gives an introduction for new readers and describes important concepts for later discussion. The motivations for the project are covered in 2.3 while the state of the art is presented in 2.4.

2.1 Warmachine

Warmachine is a miniature war game played between two players on a standardized 4' by 4' table. The game is developed by an American company named Privateer Press and is set in a fictional world called the Iron Kingdoms. There are two main factions in the game: Warmachines and Hordes. Each faction consists of multiple sub-factions and there are no restrictions on which factions can play against each other. Section 2.1.6 can be consulted for a brief overview of some of the terminology that is used in this chapter

2.1.1 Fundamentals

Each player usually brings two armies when they play. When the match is about to start the players are informed of what armies the opponent has brought. Each player then makes a choice of which of their armies they will play without knowing what the opposing player will field.

The game is divided into game turns and each player gets to play one player turn for each game turn that passes. Once a game turn ends, a new one is immediately started. A chess clock is used while playing and each player has one hour to complete all player turns. There are three ways a game can end: A player runs out of time, a player completes a scenario objective (this takes multiple game turns and involves controlling one or multiple key positions on the board) or one of the players loses his warcaster. The warcasters are army-defining miniatures that act as the commander and general of the army.

Miniatures interact with each other through ranged attacks, melee attacks and magic attacks. They all have offensive and defensive stats that determine their combat prowess and two six sided dice are used to determine if an attack hits its intended target and how much damage it does.

The picture in figure 2.1 shows an infantry focused Cygnar army, one of many factions in the game, that uses the warcaster Lord General Coleman Stryker. He can be seen in the middle of the army carrying an ensign. He has three warjacks with him (the robots), some heavy cavalry in the back, a unit of trenchers in the front and a few key solo miniatures in the middle.

Figure 2.1: Warcaster Lord General Coleman Stryker and his army

2.1.2 Miniatures, Units and Stats

Warmachine consists of five different miniature archetypes: Warcasters, warjacks, units, attachments and solos. There are some rules for what combination of miniatures a player can take, which is expanded on in 2.1.4. Each miniature is represented by a special card that describe its combat abilities containing the following information:

Name	SPD	STR	MAT	RAT	DEF	ARM	CMD	HP	PC
Major Victoria Haley	6	6	6	5	16	14	8	15	-25
Ironclad	5	11	7	6	12	18	-	30	12
Arcane Tempest Gun Mages	6	4	5	7	14	11	7	1	11
Arcane Tempest Gun Mage Officer	6	4	6	8	14	11	9	5	4
Gun Mage Captain Adept	6	4	5	8	14	11	7	5	5

Table 2.1: Unit stats

The table shows the stats for five different miniature archetypes in the same order as they were introduced.

1. The warcaster Major Victoria Haley
2. A Warjack called the Ironclad
3. A unit of Arcane Tempest Gun Mages
4. A unit attachment for the Gun Mages called the Arcane Tempest Gun Mage Officer
5. A ranged combat solo called the Gun Mage Captain Adept

The attributes covered in the table are extremely important as they describe how far the model can move, its combat proficiency and how expensive it is to include the model in an army.

- SPD stands for Speed and it represents how many inches a miniature can move before making its combat action.
- STR stands for Strength and describes how hard a miniature hits in melee. When making an attack with a melee weapon the miniatures STR is added to the power of the weapon it is using to attack with.

- MAT stands for Melee Attack and shows how proficient a miniature is at hitting miniatures with its melee weapon.
- RAT stands for Ranged Attack and represents how good a miniature is at hitting miniatures with its ranged weapons.
- DEF stands for Defense. It represent a miniatures ability to evade enemy attacks.
- ARM stand for Armour. It describes how tough it is to damage a miniature after it has been hit.
- CMD stand for Command. If a miniature has a value here it represents the command range and command stat for the miniature.
- HP stand for Hit Points. The amount of hit points a miniature has varies wildly depending on what the miniature is intended to do.
- PC stand for Point Cost. This describes how expensive the model is to include in an army. Warcasters have a negative point cost i.e. they allow you to take extra miniatures. This is expanded upon later.

With the above information in mind it is possible to explain how miniatures interact: by attacking and damaging each other

1. **Attacking:** When a miniature declares an attack against another miniature the attacker first has to roll dice to hit. This is done by taking the MAT, for melee attacks, or RAT, for ranged attacks, stat of the attacker and adding the result of rolling two six-sided dice. The sum of the dice and the relevant attack stat has to be equal to or exceed the defending models DEF stat to hit. If it is lower the model evades the attack and takes no damage. Rolling 6 on both die results in an automatic hit while rolling two 1s results in at automatic miss.
2. **Damage:** Once a hit has been landed the damage needs to be determined in a similar way. Each weapon has an offensive power which is also visible on the miniatures card. The power of the weapon is added together with two six-sided dice and the defending models ARM stat is subtracted. If the total sum is 0 or lower the defending model takes no damage despite being hit by the attack. Some miniatures specialize in defense such as the Gun Mage Captain Adept while others are sturdier but easier to hit such as the Ironclad. Some miniatures also have the ability to boost their attacks and their damage rolls which means they can add an additional dice when trying to hit or damage another miniature.

From table 2.1 it is clear that some models have much better stats than others, such as the ironclad with its 30 hit points and high armour. Warcasters also have better stats than most combat solos. To make sure players do not only pick models for their raw stats many miniatures have special rules and abilities. This means that even if a miniature has low overall stats it might still have very powerful abilities, making it a valuable choice for some armies.

2.1.3 Terrain and Objectives

Due to the relatively large table size warmachine is played on it is necessary to ensure some differences between each match. This is done through terrain and objectives. Before every game starts, and even before the players decide which armies they are playing, some terrain is placed on the table. There are 6 different terrain features that are present in most matches: forests, hills, water, walls, impassable objects and rough terrain. These terrain features help provide protection against miniatures that can shoot or they impair movement in some way. The terrain should be placed in order to favour one side of the table. This is critical to ensure that the inherent advantage given to the player who moves his models first is not too large. After the players have seen the terrain and decided on what armies they are playing each player rolls one six-sided dice. The winner can either pick his starting side or decide to move his models first, leaving the choice of table side to the opponent.

Every warmachine game also has objectives. Objectives can be controlled by placing miniatures next to certain objects or by standing inside predefined zones. When a player controls an object or a zone he gains control points at the end of each player turn, starting at the end of the second players second turn. An object can be controlled if there are no enemy miniatures close-by. If a player reaches five control points while his opponent has fewer than five the player immediately wins the game. Control points are very important for the competitive nature of the game, and they play an essential part in allowing different army archetypes to exist. If the only way of winning a game was to kill the opponents caster the game would focus exclusively on tough armies and warcasters that are good at fighting. To avoid staleness Privateer Press annually releases a new rule book for competitive play called a Steamroller. This is used as an opportunity to try new things and they create new map layouts and scenarios that are used in both their official tournaments and smaller local ones.

2.1.4 Armies

Army composition and army construction is a highly complex problem in warmachine, and it is the main topic of this project. There are rules for what models can be brought in an army and there are a lot of external factors to consider when a player decides which models to include in an army. This section first introduces the basics of army building and then proceeds to give an introduction to some of the more complex considerations a player must make when creating an army.

Every army must have one miniature that acts as the commander: the warcaster. The warcaster is the most powerful miniature in an army and they often have extremely strong abilities. Every warcaster comes with warjack points which are extra army points that must be spent on warjacks. A player must spend, as a minimum, all his warjack points to create a legal army. The player is allowed to spend more than his warjack points on warjacks should he wish to do so. Some miniatures also have special rules that prevent them from being in the same army. A legal army can at most be two points below the maximum army size. For example if two players are playing a match where each army has a max size of 75 points, the minimum acceptable army size is 73 points.

Armies vary greatly in warmachine, both on the casual and the competitive side, but there are some general archetypes that are common: Unit focused armies, warjack heavy armies and solo heavy armies. These archetypes can then be tailored to focus on fighting the opposing army in close combat, by ranged superiority or battlefield control. Warcasters cater to different army types and to create a successful army it is important that the warcaster and the army composition complement each other. These synergies are often highly abstract and rely on understanding complicated abilities and their interactions. By utilizing a CBR system where the case base consists of properly composed armies there is less need for the system to have an in-depth understanding of these interactions. Instead it relies on the already proven armies and attempts to tune these to the users wishes.

When creating an army it is important to consider its strengths and weaknesses and what races and warcasters you want the list to be able to play against. In most tournaments each player brings two armies, so it is vital that the weakness of one army is covered by the strength of its partner. There are also certain warcasters and army types you have to consider when making a pairing, as they are extremely hard to deal with if no precautions are taken. These cases are largely ignored in this project since the current implementation of the system does not consider army pairings when it creates and adapts armies.

Due to the complexity of the game and the many different factions some limitations are necessary. The system only focuses on creating armies for a single race: Cygnar. This is the faction the author has the most experience with. This greatly reduces the scope of the project, ensuring that similarity functions and domain knowledge only needs to be implemented for a single race. It is also one of the most popular races in the game which makes assembling an expert group much easier. The system-design is extendable and as such it should be easy to include other races at a later time if necessary. With all of the previous information in mind it is now possible to show how a ready to play warmachine table looks like.

Figure 2.2: Warmachine table with armies deployed.

The picture in figure 2.2 shows two armies that are ready to play with 75 army points for both sides. The closest army is the Cygnar army from figure 2.1 and the opposing player is fielding an army from the Retribution faction. Several terrain pieces have been placed on the board and the two flags are scenario objectives that can be controlled. On the right side a chess-clock can be seen together with some measuring tools that are used to measure the miniature movements.

2.1.5 Metagaming

In most competitive games metagaming is any strategy, action or method relying on out-of-game information or resources to affect one's plans or decisions in the game. In warmachine this usually means having knowledge about what miniatures are strong, what combinations are popular and which warcasters are played the most. This information can be used to make decisions around army building: favouring miniatures that are strong against the expected opponents. The meta game is constantly in flux and can vary greatly between areas. Most warmachine players have a local game club where they play their weekly games. The player pool is limited at a place like this and skill disparity between players can be large. This can greatly affect what decisions a player should make when building an army or a pairing to play against an opponent from the local game club. For example, if 50% of the players at the local club all play the faction Khador it is important that any army or army pairing you bring is capable of playing an even game against casters from this faction.

At larger international events the meta game is usually different to the local meta. There are more players playing the strongest warcasters and there is usually a less skewed attendance, with representation from most of the races in the game. These tournaments are an important part of forming the future meta game. Casters and armies that perform well at these events often become popular at both the local level and future large events.

Twice a year Privateer Press releases erratas, which are updates to the game. In these documents miniatures are changed and they try to lower the power of the over performing models, while increasing the power of those that rarely see play. Together with tournament results these erratas make sure that the meta game is constantly changing over time and it is important to stay updated on what is happening if you want to be a competitive warmachine player.

2.1.6 Terminology

This section describes some key terminology related to warmachine and gives a brief explanation. This list serves as a look up table in case later terminology is confusing.

1. Army: A collection of miniatures put together following the warmachine rulebook.

2. Army size: The size of an army, measured in points. For example most games are played with an army size of 75 points. This means a player can

spend 75 point on miniatures plus the amount of warjack points the chosen warcaster has.

3. List: Short for Army List, a commonly used synonym for an army.
4. Warcaster: The general or commander of an army. An army can only contain one warcaster.
5. Warjack: A special miniature that is attached to the warcaster or specific solo miniatures that can control warjacks.
6. Unit: Multiple miniatures that can only be taken together. It is not possible to just bring a single model.
7. Unit Attachment: A miniature that can be attached to a Unit. An attachment usually brings extra abilities that apply to the whole unit.
8. Warcaster Attachment: A miniature attached to the Warcaster, granting the warcaster various new capabilities.
9. Solo: A miniature that can be included in an army by itself.
10. Battle group: The warjacks and warcaster attachments a player chooses to bring to a match is often called a battle group.
11. Pairing: Two armies that are brought together to a warmachine match. Only one of the armies is actually played.
12. Points: Each miniature in warmachine costs a set amount of points to include in an army.
13. Eratta: A pdf document released by privateer press detailing changes to models that have already been released. Essentially a small update to the game.
14. Privateer Press: The owners of Warmachine.
15. Meta: Strategies or army building decisions relying on out-of-game information or resources.
16. Steamroller: A pdf document released online that gives information and guidance to how warmachine should be played competitively.
17. Model: Synonym for miniature.

2.2 Case-Based Reasoning

Case-based reasoning is imagined as a cyclic process consisting of four main steps: Retrieve, Reuse, Revise and Retain. These are commonly referred to as the four R's of case based reasoning. A much used depiction of the structure was first published in the article "Case-Based Reasoning: Foundational Issues" [Aamodt and Plaza, 1994].

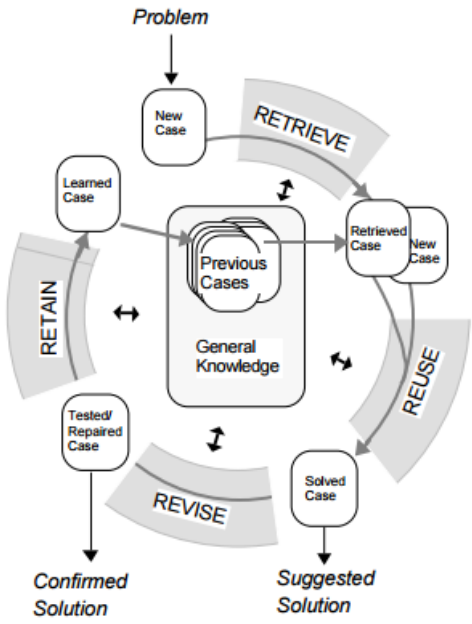
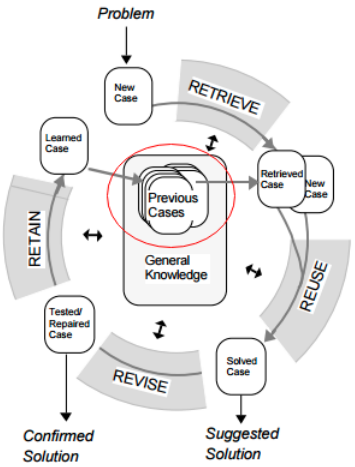


Figure 2.3: The CBR cycle as presented by Aamodt and Plaza [1994]

From the image it is clear that each part of the cycle feeds the next one with the necessary information. There is also a set of previous cases stored in the middle in what is commonly referred to as the case base. A CBR system also requires some general knowledge about the domain the AI is working in. This knowledge can be crucial to compare and analyze cases, perform adaptations or to evaluate the performance of a proposed solution before it is retained. This section examines the crucial parts of the CBR cycle and gives an introduction to the theory surrounding it.

2.2.1 CaseRrepresentation

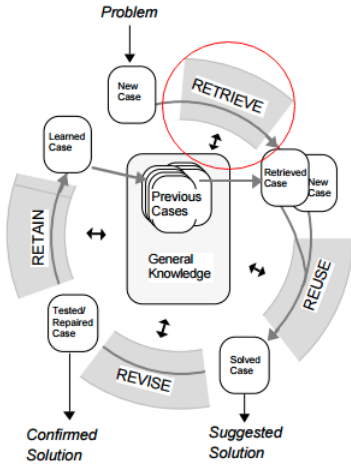


Case representation is essential for all case-based reasoning systems due to its central role. The case representation affects all parts of the CBR methodology since every step of the CBR cycle operates on case instantiations. A proper representation is important in all AI system, and there have been some attempts at defining what a good representation entails. According to a CBR book released by Richter and Weber [2013] some of the most important factors are:

- Representational Adequacy - Can the system represent everything of interest?
- Inferential Adequacy - Is the system able to infer new knowledge on its own?
- Inferential Efficiency - From a computational perspective, how easy is it to infer new knowledge?
- Acquisitional Efficiency - How easy is it to formalize new information?
- Syntax and Semantic - Is it easy to clarify what is allowed and what is not.
- Naturalness - Is the representation easy to use and understand? Does it fit intuitively into its domain?

To ensure that the systems case representation is adequate it attempts to fulfill these requirements. The class representation is based on the warmachine rulebook and their representation of miniatures. This way the system should be able to represent everything that is needed. New information is inferable through the adaptation process at a relatively low computational cost and as the case base grows new knowledge is introduced. Knowledge is formalized and acquired in the retention step. The object oriented approach the project uses makes the representation both natural and expressive.

2.2.2 Retrieval

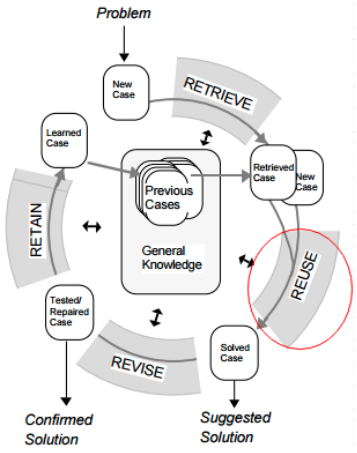


Retrieval is the first part of the case-based reasoning cycle and it happens as soon as the user has entered a query. The system retrieves previously used cases that are similar to the requested query. Different retrieval techniques should be considered depending on the size of the case base, its structure and the type of similarity the system wants to use. An overview of different techniques are covered in a review article by Lopez De Mantaras et al. [2005]. They mention two main techniques for retrieval: Surface- and Structural similarity retrieval.

Surface retrieval compares the query to each case in the case base and assigns a value between 0 and 1 to each case, depending on some similarity metric decided by the implementation. Structural similarity methods are usually more complex and they attempt to leverage domain knowledge to discover deeper structural connections between cases and the query. Structural similarity retrieval is a more computationally expensive process, but the cases that are returned can have important advantages in some domains. For example, they might be more suited for adaptation.

Since the main focus of this project is adaptation the retrieval method is kept simple. The system initially retrieves all its cases from the case bases and then utilize a k-NN method to select those that seem promising. The most similar case can then be passed to the reuse step.

2.2.3 Reuse and Adaptation



At its core the reuse step is the application of similar cases to a newly proposed problem. There is rarely a one-to-one match between a previously encountered problem and a new one, and as such the solution is usually to adapt parts of the previous solution to match the new problem. For the proposed system this means that it has to replace some of the under performing miniatures in the user query with better performing miniatures from a similar army. This is not a trivial task due to the different point cost of miniatures, keeping synergies intact and the fact that miniatures can rarely be substituted one-for-one.

As mentioned earlier adaptation is considered one of the most difficult parts of the CBR cycle [Patterons et al., 2002] and it is often simplified while focusing on other areas. In some cases it is possible to lessen the need for adaptation by increasing the capability of the retrieval process. This ensures that the retrieved cases do not require a lot of changes to be applicable to the user query. However this is not realistic for all domains and especially domains that focus on design, configuration, and planning often require case adaptation [Lopez De Mantaras et al., 2005]. Creating armies for warmachine can be viewed as a configuration problem and there are usually two categories of adaptation depending on the implementation of the CBR system:

1. **Transformational adaptation** takes place when rules or formulae are applied to a solution to alter it for the current problem.

2. **Derivational/generative adaptation** reuses the algorithms, method or rules that generated the original solution to produce a new solution to the user query. This means that the system must store the planning sequence that was used to create a solution along with the solution itself.

This project is focused on testing transformational adaptation methods that fits the case representation. Comparing derivational and transformational adaption methods requires a system that is capable of performing both types of adaptation, which is outside the scope this project to implement.

Once the decision to focus on transformational adaptation methods has been made it is necessary to consider and decide which types of transformational adaptation the system will test and compare. There are several possibilities.

1. Methods of Adaptation:
   (a) **Random adaptation** can be used as a baseline to compare other procedures to. The method takes randomly chosen parts of the user query and replaces them with random miniatures.
   (b) **Null adaptation** is the simplest form of adaptation. The case base is searched for a similar case and it is returned as a solution to the current problem without any adaptation.
   (c) **Transformational adaptation** means that an old case solution is transformed into a new solution for the user query. It supports reorganisation of solution elements and permits modifications, additions and deletions of these elements.
      i. *Substitutional adaptation* is an adaptation model where it is only possible to change the values of attributes. These are usually numerical values that can be tuned depending on the case. It is a fitting technique when the retrieved solution is a close match to the user query and consequently only requires minor adaptations for some of the variables. The values of the attributes are often completely reused from the most similar case, but some of them are tuned to fit the current problem. Examples of system using substitutional adaptation include the JUDGE system [Bain, 1986a] and a general framework applied to P-truck curing [Sara Manzoni and Vizzari, 2007]
      ii. *Structural adaptation* supports changing the structure of the solution during adaptation. This is done by the addition and deletion of solution elements according to a predefined set of rules. An example system is available in "Using adaptation knowledge to retrieve and adapt design cases" by Smyth and Keane [1996]
   (d) **Compositional adaptation** is a method where a solution is created by combining several partial solutions to a user query. Due to the complexity of some domains and queries, sometimes a single similar case is not enough. By combining cases that have different parts of the needed information a CBR system might be able to create a solution where it would otherwise be unable to. The technique was explore as early as 1990 in an article called "Distributed cases for case-based reasoning; facilitating use of multiple cases [Redmond, 1990].

2. Concrete techniques

   (a) **Reinstantiation**: In renstantiation, the old and new problems are structurally similar, but differ in the values of elements. Reinstantiation involves replacing the old values with new ones from the most similar case. An example of this is the CBR system CHEF where it creates a chicken and snow peas recipe based on its beef and broccoli recipe. The *meat* role of beef is replaced by chicken and the v*vegetable* role if filled by snow peas. [Hammond, 1986]

   (b) **Structure transfer**: Structure transfers can be done by identifying and extracting substructures, and possibly combining them. This requires a case representation where it is possible to both extract and replace substructures. The system must either be implemented with knowledge to identify these substructures or it must be able to learn how to identify them over time. The technique is a somewhat novel take on several of the structure based adaptation techniques described by Lopez De Mantaras et al. [2005] and is intended to work with the warmachine domain.

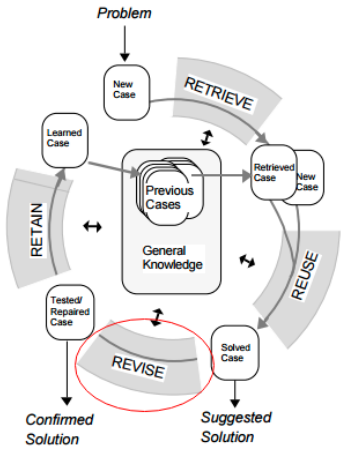
   (c) **Parameter adjustment**: This is a substitutional adaptation technique where specified parameters of the most similar case and the user query are compared to each other and changed. It usually requires a numerical representation of the values that are altered. An early example of this technique is the JUDGE system. [Bain, 1986b]

   (d) **Abstraction and respecialization** this technique (also called local search) is a type of substitution that allows a novel solution to be generated from an example which differs only in a small part from the user query. The core idea is to take the piece of a similar case that does not fit with the user query, look for a good abstraction and then try other specializations of the abstraction for the query. An early example can be seen in "An adaptiver Planner" [Alterman, 1986]

As stated in 1.2 the goal of the system is to test some of these methods against each other in the warmachine domain and see how they perform: **Random adaptation** is used as a baseline and makes it possible to compare the other adaptation methods to a random selection. **Null adaptation** is tested by identifying the most similar army to a user query and presenting this as the solution. The technique is dependant on the case base and the quality of the armies within. **Structure transfer** is implemented to identify key structures in the armies that perform well with a specific warcaster and then tries to adapt these miniatures into the user query. The **Reinstantiation** procedure is modified to utilize the most similar case as a basis and then adapts parts of the user query into this case.

Parameter adjustment does not fit the domain or the case representation. In warmachine every miniature is either included or not, there are no real numerical parameters to adjust. **Abstraction and respecialization** could be an interesting technique for the warmachine domain, but it would likely require the system to have a large amount of domain knowledge and a larger case base. Deep domain knowledge would be crucial to identify and find abstractions for key miniatures that are missing in a query. The case base might also need to be larger to have a better chance at finding a good starting point where there is less adaptation required. The method would also struggle to create large improvements for bad input armies since it only tries to identify a few local improvements.

2.2.4 Revision



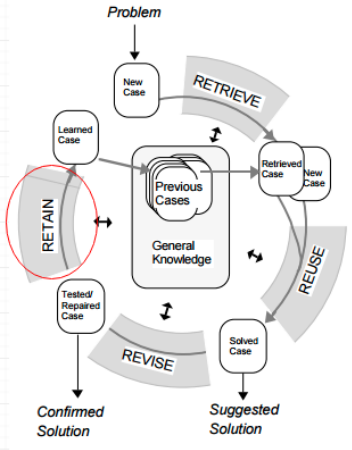
Once adaptation has finished the system presents its new solution to the user. There are two commons way to evaluate the performance:

1. **Field test** the solution by trying it out in the real world. This is usually the best way of thoroughly testing a solution, but it is costly and hard to do to. For the warmachine domain a proposed solution would ideally be played several times, against different races while paying close attention to how the proposed changes perform in practice.

2. **Expert feedback** can be used to form an opinion of the proposed solution without having to test it in the real world. It requires access to knowledgeable individuals and if only a single expert is consulted the feedback can be skewed.

Based on the expert feedback or the field tests the system should either move to retention or the proposed solution should be revised. The revision can be performed manually by the users or the system can go through another reuse and adaptation process with manually inserted feedback feedback.

2.2.5 Retention



Retention is the final step of the CBR cycle. The system must decide if the derived solution should be added to the case base. While this might seem like a trivial decision superficially (adding more cases means the system has access to more knowledge) there are several factors that should be considered before a case is stored.

If a CBR system has no retention strategy its case base grows indefinitely as the system is used. There are both negative and positive side effects to this. The case base might eventually have a debilitating amount of cases, which means that the retrieval process slows down due to the large amount cases that have to be searched. Conversely, storing a lot of cases means the retrieval process is more likely to find a good match to the current problem, which can play a big role in simplifying the adaption process. This problems occurs in many different AI algorithms and is commonly referred to as the utility problem: *An increase in the systems overall knowledge can actually lead to a degradation of performance.* This topic is covered in "The Utility Problem in Case-based reasoning" by Francis and Ram [1993] where they discuss some important coping strategies.

One solution is to implement a deletion policy that prunes the database, ensuring that the slowdown never becomes unreasonable. This requires a deletion strategy that makes sure the system does not delete crucial cases. Crucial cases can be both frequently accessed cases, or cases that contain unique information for niche situations. This problem is discussed by Smyth and Keane [1995] and they suggest using a competence model to evaluate individual cases. It is also possible to limit the case base size by being critical of which cases are added. By only including cases that include a concrete amount of new information the case base can be made to slowly increase in size.

Another interesting proposition for a system that sees use over a long time period is case relevance. As discussed in 2.1.5 the game naturally changes over time and both new miniatures and new army compositions become popular. One way of dealing with this is by utilizing a time stamp or a date attached to each case. This allows the system to use the case-age as a weight when comparing similarity in the retrieval process. Old cases are less relevant while still providing possibilities if there are no recent cases that are applicable to a user query.

2.3 Motivation

There are several external motivating factors for the creation and development of this system. Initially there is the promising development where AI systems are becoming increasingly capable as both techniques and computing power improve. This progress is demonstrated by the earlier examples from Chess and Go. Once computers reach a level where they can compete with humans they are able to calculate and evaluate very complex situations. They see moves and connections that humans do not and can play an important part in the continued development of a game.

Miniature war games are clunky and complicated to evaluate for a computer due to the difficulty of accurately representing and transferring game states to the computer. Because of this it is natural to focus any research on the army building aspect where an AIs computational power can be utilized. In warmachine the meta game is constantly changing as new models are released and older models are updated. An AI that can access and collect data from a large amount of matches might be able to accurately predict popular armies, and even identify key weaknesses or potential opportunities that can be exploited.

This project follows two other master degrees from NTNU that have attempted to combine miniature war games and case-based reasoning. Both Warhammer Fantasy[1] [Strandbraten and Petersen, 2011] and Warhammer 40K[2] [Zikic, 2016] have been explored with some success. The different games have distinctive army building processes which pose interesting challenges for the respective systems. Initially it is interesting to explore how this affects the AI systems and as the domain matures the research might focus more on a single game, trying to optimize the AI techniques that are being used. This project acts as an initial foray into warmachine specifically and explores how well the domain is suited for case-adaptation.

[1] https://en.wikipedia.org/wiki/Warhammer_Fantasy_(setting)
[2] https://en.wikipedia.org/wiki/Warhammer_40,000

Adaptation is both complex and powerful which makes it an inherently interesting research area. All of the adaptation procedures that are tested in this project have been used before, but most AI techniques require changes and tuning to be applicable in a new domain. It is intriguing to examine what kind of changes are necessary to make them work for warmachine and the lessons learned might be valuable for other miniature war games.

2.4 State of the Art

The state of the art for this paper is based on a similar body of work that laid the groundwork for the creation of this system: Case Based Reasoning for Warmachine and Hordes [Fusdahl, 2016]. The articles have been found by performing a structured literature review as outlined in "How to do a Structured Literature Review in Computer Science" [Kofod-Petersen, 2014]. For additional details regarding the review and its execution the original paper should be consulted.

There are some key differences between the focus of this system and the one covered in "Case Based Reasoning for Warmachine and Hordes". The originally designed system focuses on the potential for utilizing semantic networks or a CSP solver to aid the adaptation. Articles focusing on these aspects have been replaced with key articles that utilize and explore the adaptation techniques used in this system. The articles that focus on important and general concepts have also been included.

Article	Authors
Case-Based Reasoning Integrations	C. Marling, M. Sqalli, E. Rissland, H. Avila and D. Aha
Integrations with Case based reasoning	C. Marling, E. Rissland and A. Aamodt
Integrating Bayesian Networks into Knowledge-Intensive CBR	A. Aamodt and H. Langseth
Knowledge-Intensive Case-Based Reasoning in CREEK	A. Aamodt
Explanation-driven case-based reasoning	A. Aamodt
Learning adaptation knowledge to improve case-based reasoning	S. Craw, N. Wiratunga and R. C. Rowe
Explanation in Case-Based Reasoning - Perspectives and Goals	F. Sormo, J. Cassens

Table 2.2: Articles returned from the structured literature review

Article	Authors
Myrmidia - Case-based reasoning for Warhammer Fantasy Battle army building	G. R.Strandbraaten and Anders Kofod-Petersen
Explanation-aware army builder for Warhammer 40k	Nenad Zikic
MAC/ FAC: A Model of Similarity-based Retrieval	Kenneth D. Forbus, Dedre Gentner, Keith Law
Retrieval, reuse, revision, and retention in case based reasoning	R. Mantaras et al.
A new adaptation method based on adaptability under k-nearest neighbors for case adaptation in case-based design	J. Qi, J. Hu, Y. Peng
A Case-Based reasoning system for subjective assessment	Bain
Substitutional adaptation in case based reasoning: a general framework applied to p-truck curing	Sara Manzoni and Fabio Sartori and Giuseppe Vizzari
Using adaptation knowledge to retrieve and adapt design cases	Smyth and Keane
CHEF: A model of case-based planning	Kristian J. Hammond
Distributed Cases for Case-Based Reasoning; Facilitating Use of Multiple Cases	Micheal Redmond
Retrieval, Reuse, Revision and Retention in Case-based Reasoning	Lopez de Mantaras et al.
Techniques and Knowledge used for Adaptation during Case-Based Problem Solving	Wolfgang Wilke and Ralph Bergmann

Table 2.3: Articles returned from the structured literature review

Chapter 3

Architecture

An architecture is a conceptual model that is used to define the structure and expected behavior of a system. It is a formal description and representation that makes it possible to reason about and discuss both the structure and behavior. The architecture shows how different components interact without examining too many details. Most architectural descriptions also include a set of requirements that the system must fulfill. These requirements can be used as goals during development and serve as an especially helpful part of the architecture in small projects.

Section 3.1 looks at what functional requirements the system must meet to be useful in exploring the research questions. 3.2 gives an overall look at how the system is built and some important framework decisions.

3.1 Functional Requirements

The functional requirements describe capabilities the system must fulfill. The requirements are designed to support the project goals and enable the system so it can be used to answer the research questions. They describe *what* the system must be able to do:

Nr.	Requirement	Description
1.0	Case-base	Legal cases can be stored in a database where they are accessible by the system.
2.0	Retrieval	The system can retrieve cases from the case base.
2.1	Retrieval	The system can compare cases to each other, returning a numeric evaluation of their similarity.
2.2	Retrieval	The retrieved cases can be compared to a user query and the k most similar armies can be identified.
3.0	Reuse and adaptation	The system must be able to adapt solutions.
3.1	Reuse and adaptation	The system must be able to use different adaptation algorithms.
3.2	Reuse and adaptation	The system must be able to focus on adapting different parts of an army.
4.0	Revision	An adapted solution can be presented to the user, clearly showing what changes have been made.
5.0	Retention	The system can retain a solution if it receives promising feedback from an expert group.

Table 3.1: Functional Requirements

3.2 System Architecture

This section covers the systems architecture. It examines the chosen frameworks for the program and how the CBR cycle is implemented. It also gives some information about the similarity measure used to compare cases, how the adaptation techniques are implemented and how maintenance is handled. The architecture is designed to follow the CBR cycle closely with a few auxiliary support structures that aid the retrieval and adaptation process.

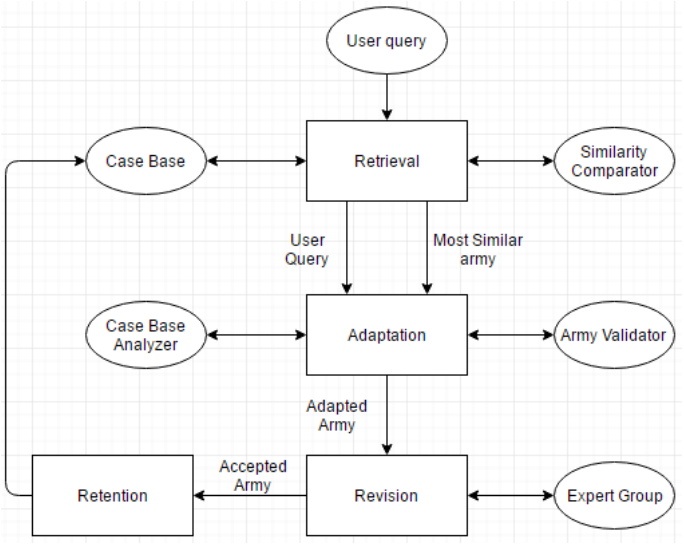


Figure 3.1: A simplified description of the systems architecture.

The architecture is mostly focused on the two initial steps of the CBR cycle: Retrieval and Revision. The retrieval process gets a user query as input and utilizes a similarity comparator and the case base to identify the most similar army. This army is sent to the adaptation procedure together with the original user query. The user gets to choose what kind of adaptation technique the system uses and the adaptation relies on a case base analyzer to perform some of the adaptations. It also requires an army validator that ensures the adaptations are legal. Once adaptation has finished the army is sent to revision where an expert group evaluates both the adapted army and the user query. If the feedback is positive the army is re-entered into the system and passed to retention where it might be added to the case-base.

3.2.1 Framework

There have been several frameworks developed for CBR system and they focus on different programming languages, and have different barriers of entry. Using a framework is always a risk, because the decrease in development time might be lost while learning and working with the framework. This project uses java as its programming language, and as a results there are three main possibilities: jCOLIBRI[1], myCBR[2] and no framework. These choices all come with advantages and disadvantages.

1. *myCBR*: As stated on their webpage myCBR is "an open-source similarity-based retrieval tool and software development kit (SDK). With myCBR Workbench you can model and test highly sophisticated, knowledge-intensive similarity measures in a powerful GUI and easily integrate them into your own applications using the myCBR SDK."[3] For this project the tools that are supported by myCBRs GUI seem insufficient, and the SDK will have to be included into the project. The SDK offers decent amounts of custimization and there is a complete javadoc available online, making it possible to get an understanding of how the SDK works. There is however a lack of in-depth tutorials that show how the SDK should be leveraged and utilized to include its functionality in your own program.

2. *jCOLIBRI*: A project that provides a reference platform for developing CBR applications: it is a clear architecture designed for CBR systems and provides a reference implementation of the different components. jCOLIBRI provides functionality to drive the whole CBR cycle and their SDK allows a lot of customization. They also provide a large amount of examples where they showcase how a CBR system can be set up with jCOLIBRI and how the SDK should be leveraged.

[1] http://gaia.fdi.ucm.es/research/colibri/jcolibri
[2] http://www.mycbr-project.net/
[3] http://www.mycbr-project.net/

3. *No framework*: Without an outside framework it is possible to change and implement the CBR cycle to be tailored specifically for the warmachine domain. It also reduces the workload required to properly learn and leverage a new SDK. On the other hand there is more development time focused on implementing the CBR cycle and creating a case base.

The final choice eventually fell on jCOLIBRI. The framework provides a lot of advantages with regards to setup time while it also provides a well documented and example driven introduction to its implementation.

3.2.2 Case Base

The case base is stored in memory and it is created from a simple sql file each time the system is used. It uses Hibernate for object-relational mapping, as recommended by jCOLIBRI. The database consists of three tables:

1. *Miniatures*: The miniature table consists of all the miniatures that were played during the world team championship(WTC) held in 2016. This is one of the biggest tournaments organized for warmachine and every army used is competitively viable. The detailed information for each miniature is stored in separate JSON files, while the database stores a miniature ID, a miniature name and a miniature type.

2. *Army*: The army table is very simple and only contains an army ID.

3. *ArmyMiniatureMapping*: This table allows the system to connect army IDs with miniature IDs, thus allowing the system to store armies and the miniatures that they contain.

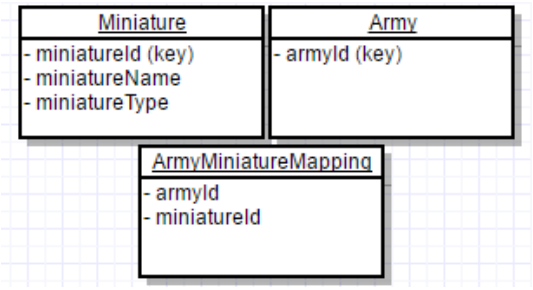


Figure 3.2: The three database tables that store miniatures and armies.

3.2.3 Case Retrieval

The CBR cycle is driven by a class named warmachineCBR. The users query is read from a specified text file and stored in an Army class. The retrieval process is initiated once the user query has been handled and it retrieves ever case from the case base and stores them in a List structure. The most similar case is identified in a specialized method and returned for adaptation.

There are two criteria that are used to determine the numerical similarity of armies. The user query is compared to each item in the case base and the weighted sum of the following criteria is used to determined the similarity to the user query.

1. Warcaster: The warcaster is the most important similarity measure and it has the largest impact on the army. Armies are always designed around the capabilities and utilities of the chosen warcaster. For the sake of simplicity the system does not try to adapt an army if the case base does not contain an army with the warcaster in question. With the current implementation this is unlikely to lead to productive results due to the tight coupling between an army and its warcaster.

2. Army composition: The army composition is split into three separate considerations and each case is given a similarity score for each one.

   (a) Battle group composition: The battle group similarity considers what warjacks and what attachment have been chosen for the warcaster.

   (b) Unit composition: For the unit composition it is important to consider what units have been chosen and what their cost and capabilities are.

   (c) Solo composition: Solos are often support pieces that are designed to help parts of the army. They provide utility and often support the units or the warcaster in some way.

In essence the retrieval identifies armies that use the same warcaster as the input army and does a rough comparison of how the armies spend their army points. In some cases there are several armies that end up with an equal similarity score and when this happens they are all returned to the user who can choose which one should be used for adaptation.

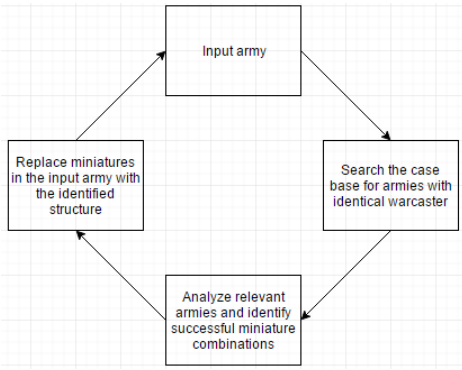
3.2.4 Case Adaptation

Once the most similar case has been found, or chosen, it is passed to the adaptation method where the user has to choose what kind of adaptation approach should be used. There are four options, as detailed in 2.2.3: Null adaptation, Random adaptation, Structure transfer and Reinstantiation. Each adaptation method, except the null adaptation, has to change at least 1/3 of the army or try to make changes at least 40 times. This threshold ensures that a strong output army is usually the result of an adaptation method and not just the original army with one miniature altered. There are however cases where it is hard for the algorithm to identify which miniatures should be changed and in these scenarios the adaptation should not go on forever. When replacing miniatures they do not always have the same point cost and once adaptation has finished any extra army points are replaced with randomly selected miniatures.

1. The **null adaptation** simply returns the most similar case as a solution and does nothing more. This method is likely to return a strong army since the case base only has armies that were used at WTC-2016. However the method is unlikely to include much of the users original army. This could be a problem in a real world system if the user does not have access to all Cygnar models or if the user wants an army that is true to the ideas present in the user query.

2. The results from the **random adaptation** varies greatly. A random amount of units, solos and warjacks amounting to at least 1/3 of the original query are chosen and replaced with miniatures of similar cost. As mentioned in 2.1.4 most armies are created around a specific themes, or with a strategic goal. Randomly changing models is unlikely to remain true to this goal, except in rare cases.

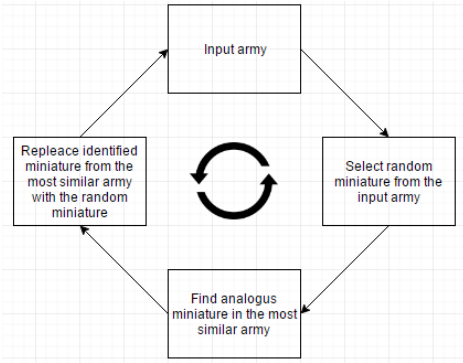
3. The **structure transfer** technique is made possible by utilizing a method that can analyze the case base and recognize common themes with different warcasters. This can be small trends, such as a warcaster always taking a specific warjack, or larger trends such as a miniature combination or a battle group winning a lot of games.



The method is implemented by retrieving multiple cases that are similar to the user query and analyzing these to recognize themes. Since the case base is relatively small compared to the number of warcasters in the game this adaptation method will take all armies using the same warcaster as the user query into consideration. To choose specific structures that are used with a warcaster the procedure considers how many times the structure has been seen and the overall win rate of the miniatures that are included.

Ideally the method would evaluate the whole case base to single out weak miniatures in the user query and replace these with the identified strong structures from the retrieved cases. However due to time constraints the procedure simply picks miniatures that have a similar total cost to the structure that is adapted into the user query. The structures should consist of at least 4 miniatures and the total miniature cost must exceed 30 army points if possible.

4. The **reinstantiation** adapts parts of the user query into the most similar item from the case base. By instantiating parts of the users query into an already assumed strong army the system adapts the recipe for a good army and includes parts of the users query. This is a modified version of the reinstantiation technique that fits both the domain and the case representation.



The adaptation technique is implemented by initially selecting a random unit, warjack or solo miniature from the input army. This miniature is then compared to the miniatures that are included in the most similar army and an analogous miniature is identified. This miniature is replaced with the one that was selected from the input army and any spare army points are used on random miniatures that are similar to the one being replaced.

This process is then repeated several times until the threshold for army changes has been reached. At this point many of the miniatures from the input army have been adapted into the most similar army and this new construction is returned as the adapted army.

3.2.5 Case Revision

Once adaptation is finished the army is printed to the console and made available to the user. The program finishes its current execution and waits for feedback, either from play testing the army or by consulting an expert with domain related experience.

3.2.6 Case Retention

After revision the system must decide if it should add the proposed solution to the case base. The army is sent back into the system through a text file together with an evaluation. If enough changes have been made and the evaluation is positive(e.g. the expert group assigns it an average score of 4 or higher) the army could be included in the case base.

However for the duration of this project none of the adapted armies are added to the case base. This is a deliberate choice that makes executing and reproducing the experiments much simpler while simultaneously taking little away from them. Not including new armies into the case base guarantees that it is stable and that the most similar army to a user query is static. This means that the armies presented to the expert group can be prepared before the group is contacted making the review process easier. Another reason for this choice is the size of the case base. A small case base is more vulnerable to drastically changing its results because of a single new inclusion.

Chapter 4

Experiments and Results

This chapter covers the conducted experiments: their setup, execution and results. Enough information is given that the experiments are repeatable in the future given access to the program and the case base. Evaluations of the army quality would of course vary, but the generated armies should be relatively similar.

4.1 Experimental Plan and Expert Group

As discussed in 1.2 and 1.3 the project attempts to gather concrete results about the use of different adaptation techniques in the warmachine domain. This is done through the use of an expert group that reviews the quality of the adaptations made by the system. The expert group has some similarities to a focus group which is a data collection method where data is collected through a semi-structured group interview process.

The system is tested by taking a set of 13 predetermined armies that were assembled by Jarle Svensrud, one of the project supervisors, or generated randomly. The designed armies are created to pose different challenges. Some armies are bad and require major overhauls while others are strong compositions that can be improved by making key changes. A few of the armies are fined tuned and could be used at a competitive event. The armies are entered as queries and each of the adaptation procedures is applied once. For each adaptation method the expert group is asked to judge the generated armies on a scale from 1-7, while also looking for common issues that happen repeatedly.

The experiments provide all of the necessary information needed to answer the research questions and makes it possible to discuss the strengths and weaknesses of the different adaptation methods.

1. RQ1 can be addressed by reviewing the scores given by the expert group to each adaptation procedure and looking at both average values and potential outliers.

2. To be able to answer RQ2 it is necessary to ask for more detailed feedback from the experts when they consider an army to be weak.

3. RQ 3 can be answered by asking the expert group to also rate the original user query. This makes it possible to compare the values and see if there are any significant differences between the rating given to the input army and the output army.

4.1.1 Rating Scale

The use of a scale from 1 to 7 is a deliberate choice for several reasons. Using an even scale forces the respondent to towards favoring one of the possible answers while an odd scale allows the respondent to chose a midpoint where they maintain a neutral stance. It ensures sufficient granularity which allows the expert group to differentiate between similar armies. Having seven options is based on the findings in an article called "Effect of the number of response categories on the reliability and validity of rating scales" by [Lozano et al., 2008]. Through a series of experiments they found that with fewer than four alternatives reliability and validity decreases, while with more than seven there is hardly any gain in accuracy.

4.2 Experimental Execution

The experiments are performed with 11 designed armies and two randomly generated ones. To run the program it is necessary to have java 8 installed, while everything else is available in the project folder. File paths have to be changed to run the program on different computers. Query armies are supplied in a specialized query file where keywords are used to describe the role of each model included. If there are several possibilities for the most similar case they are presented to the user who then has to decide which one to use for adaptation.

After retrieval has finished the user is prompted to choose which adaptation technique should be applied. The adaptation procedures sometimes create illegal armies due to residual bugs and during the experiments this was handled by reentering the user query and running the program again. The proposed solution is presented to the user through console once adaptation has finished.

4.2.1 Parameters and User Queries

The user is prompted for some input during execution as explained above. Some of the armies that resulted in interesting adaptations can be seen on the next few pages together with an exaple query file. Appendix A can be consulted for a complete list of the user queries.

Example Query File	***R1***
warcaster:haley1	Nemo3
warcasterWarjack:centurion	-Avenger
warcasterWarjack:charger	-Defender
warcasterWarjack:defender	-Grenadier
warcasterWarjack:dynamo	-Ironclad
juniorWarcaster:junior	-Lancer
juniorWarjack:hunter	Max precursor knights
unit:max long gunners	Max Tempest Blazers
unitAttachment:long gunner ua	Anastasia
solo:gmca	

Case 2

Haley2
-Squire
-Firefly
-Dynamo
-Thorn
Max storm lances
Max storm lances
Lanyssa
Laddermore
Gun Mage Captain Adept

Case 5

Caine2
-Reinholdt
-Stormwall
Journeyman Warcaster
-Charger
Rangers
Alexia1
Max trenchers
-Trenchers UA
Arlan
Gobber Tinker

Case 9

Stryker2
-Squire
-Ironclad
-Ironclad
-Thorn
Journeyman Warcaster
-Charger
Max Trenchers
Max storm lances
Major Harrison Gibbs
Alain Runewood
Ragman

Case 10

Stryker2
-Ol'rowdy
-Hunter
-Firefly
-Lancer
Arcane tempest gun mages
Black 13th
Min mechanics
Min blazers
Max stormguards
Maxwell Finn

Case 6

Maddox
-Squire
-Ironclad
-Stormclad
-Thorn
Max storm lances
Max long gunners
Max trenchers
Gobber tinker

Case 11

Maddox
-Squire
-Avenger
-Centurion
-Ironclad
-Lancer
Max Long gunners
Stormblades
Max trenchers

4.3 Experimental Results

The results of the experiments are presented in a series of graphs showcasing the different scores that were given to the army proposals for each adaptation procedure. There are also graphs and tables examining how the adaptation methods perform. Appendix B provides a complete list of the evaluations done by the expert group.

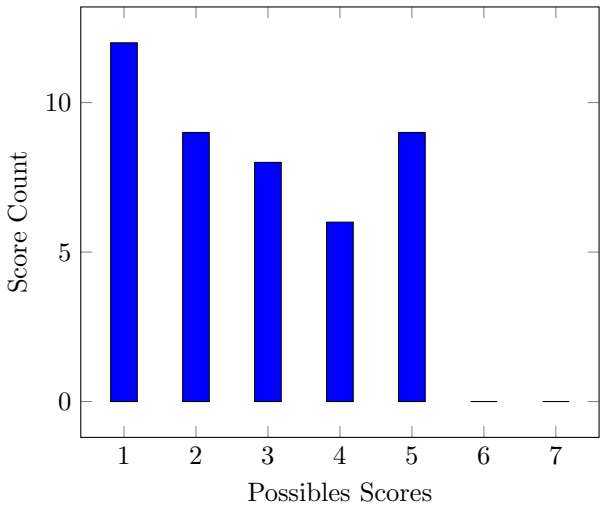
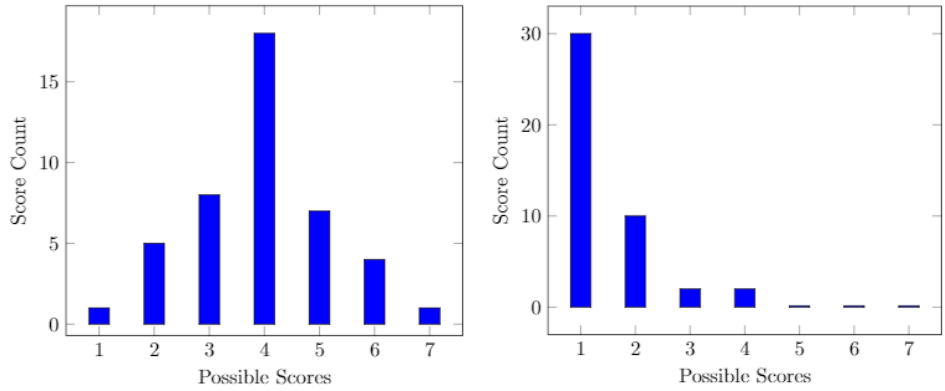


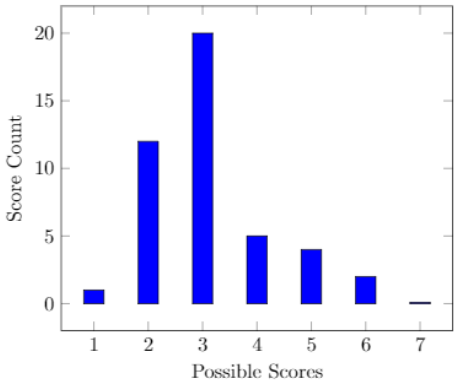
Figure 4.1: Number of times each allowed evaluation score was given to the input armies

Figure 4.1 counts individual ratings from the experts: For example out of the thirteen input armies the score 1 was given 12 times: three experts gave input army R2 a 1 rating, resulting in three additions to the 1 bar. This process is repeated for all the adaptation techniques in figure 4.2.

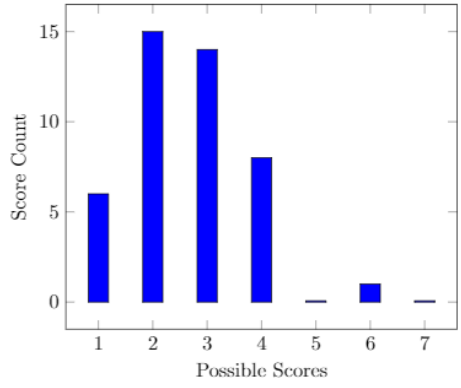


(a) Null Adaptation

(b) Random Adaptation



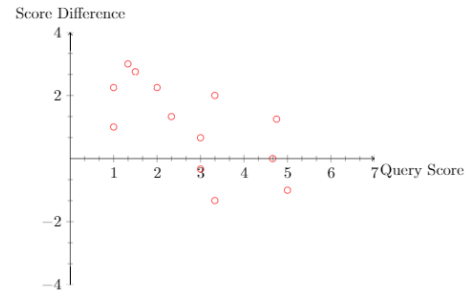
(c) Structure Transfer Adaptation



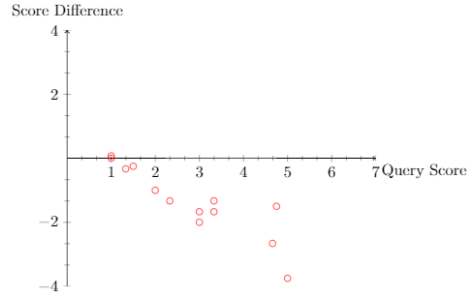
(d) Reinstantiation Adaptation

Figure 4.2: Number of times each allowed evaluation score was given to the different adaptation techniques

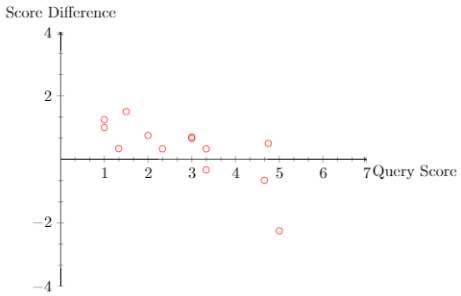
Another interesting aspect to examine is how the adaptation methods perform depending on the quality of the input army. It is possible to explore this relationship by calculating the average score given to each user query and comparing it to the average score given to the proposed solutions created by the adaptation procedures. In figure 4.3 there are four graphs showcasing this relationship. The x-axis represents the query score while the y-axis describes the difference between the query score and the adaptation score. If the value is positive it means that the proposed solution was given a higher average rating than the input army. The graphs are based on the data available in table 4.1.



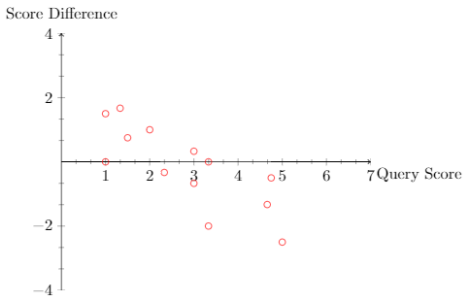
(a) Null Adaptation



(b) Random Adaptation



(c) Structure Transfer Adaptation



(d) Reinstantiation Adaptation

Figure 4.3: Average score for each adaptation technique compared to the corresponding average score for each user query.

Case ID	11	12	10	8	1	3	4	7	6	13	9	5	2
Avg. Query Score	1	1	1.33	1.5	2	2.33	3	3	3.33	3.33	4.66	4.75	5
Avg. Null Score	2	3.25	4.33	4.25	4.25	3.66	2.66	3.66	2	5.33	4.66	6	4
Avg. Random Score	1	1	1	1.25	1	1	1	1.33	1.66	2	2	3.25	1.25
Avg. Structure Score	2	2.25	1.66	3	2.75	2.66	3.66	3.66	3	3.66	4	5.25	2.75
Avg. Reinst. Score	1	2.5	3	2.25	3	2	3.33	2.33	1.33	3.33	3.33	4.25	2.5

Table 4.1: Average Scores for user queries and the resulting proposed armies

Based on figure 4.3 and table 4.1 it seems like null-, structural- and reinstantiation adaptations might be leading to overall improvements, while the random adaptation almost always creates a worse army. To further examine this a series of t-tests can be used. The scores given to each adaptation technique can be compared to the scores given to the query armies. The null hypothesis (H_0) being tested is that there exists a significant difference between the mean of the queries and the mean of the armies created by each adaptation technique.

Adaptation technique	N	Mean	stDev	Variance	T-value compared to Query	Evaluation
Query Scores	44	2.79	1.503	2.26	-	-
Null Adaptation	44	3.93	1.26	1.6	3.84	Accept H_0
Random Adaptation	44	1.46	0.79	0.63	5.24	Accept H_0
Structural Adaptation	44	3.11	1.13	1.27	1.12	Reject H_0
Reinstantiation Adaptation	44	2.63	1.08	1.17	0.57	Reject H_0

Table 4.2: Two tailed T-test with a 5% confidence interval. Critical T-Value: 1.664.

These tests show that there is a significant difference between the mean values of the query score and both the null adaptation and the random adaptation. However the null hypothesis can not be rejected for the structural and reinstantiation adaptation techniques. This means that the average score given to the armies created by these adaptation procedures is not significantly different than the average score given to the input armies.

Chapter 5

Evaluation and Conclusion

This chapter evaluates the results of the experiments and looks at both overall performance and examines interesting individual cases. It explores the expert feedback and attempts to answer the original research questions. The chapter also looks at potential validity threats and discusses both merits and limitations of the adaptation procedures. Finally, contributions from the project and potential future work is discussed.

5.1 Evaluation

This project has studied four different CBR adaptation techniques and their merit in the warmachine domain. User queries were passed through the system to generate output armies that were then given to an expert group for consideration. This section presents an evaluation of central points from the experiments while also examining specific cases.

5.1.1 Army Quality

The general performance of each adaptation method can be examined by studying figure 4.2 and 4.3, but it is also interesting to look at the overall average scores given to the input armies and the proposed solutions.

Adaptation Technique	Average Score
User Query	2.80
Null Adaptation	3.93
Random Adaptation	1.46
Structure Adaptation	3.12
Reinstantiation Adaptation	2.63

Table 5.1: Average overall score for Query and Adaptation techniques

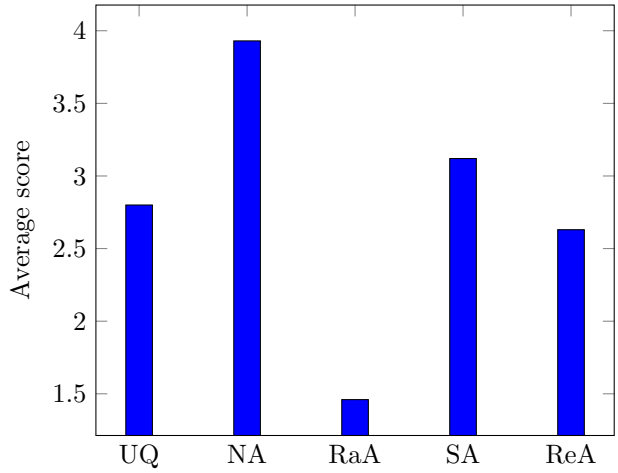


Figure 5.1: Average overall score for input armies and proposed solutions

The overall quality of the input armies is rather low as can be seen from figure 4.1 and 5.1. With an average score of only 2.8 out of 7 there is a lot of room for improvement in almost all of the input armies. The best performing adaptation method turns out to be the null adaptation. This method returns an army used at the WTC-2016 that is similar to the user query. It is unsurprising that this method performs better than the adaptation techniques since the army quality at a large international competitive event is assumed to be high.

However the technique cares nothing for the original army idea used by user and the average score given a null adapted army is only 3.93, lower than the middle evaluation of 4. There might be several factors causing this relatively low average score:

1. Most importantly not all warcasters are good enough to be included in a very strong army. When asked about this members of the expert group explained that only a handful of Cygnar warcasters are considered competitively viable and could be included in army able to receive a 6 or 7 rating.

2. The sample size is quite low and only a few of the query armies use the best warcasters. This implies that many of the input armies have a maximum rating that is lower than 7.

3. A lot of the armies that are played, even at a competitive level, are relatively weak when compared to the absolutely best armies that can be made. Ultimately the game is only played at an amateur level (no one gets payed to play the game) and there is virtually no money involved in winning tournaments. This results in a player base that is more willing to experiment and play for fun even at competitive events.

In reality it is likely that all of these factors are combined to cause the relatively low average score given to the null adapted armies. Thirteen user queries is not a large sample size, and many of the armies stored in the case base were never returned to the user as a null adapted army. Another likely contributing factor is that once a game play meta has settled down there are probably only 3 or 4 warcasters and perhaps ten different army lists that can receive a 7 rating. However there are near infinite army possibilities that would receive a 1 or 2 rating.

When examining figure 4.3 it is clear that the random adaptation is the weakest method. The adapted armies are almost exclusively rated 1 or 2 and the detailed feedback given from the experts describe an adaptation technique that is completely unable to identify or preserve synergies and often manages to make bad armies even worse. Some example feedback from the experts supports this notion:

1. Rating: 1. About as bad as it gets.

2. Rating: 1. Sorry, just random crap.

3. Rating: 1. Just bad, lack of obvious synergies.

4. Rating: 1. Missing so many crucial choices.

Important model choices are often removed by the adaptation technique and there are no guarantees that they are replaced with anything that fits with the army theme. At any point in time there are usually two important aspects to building an army around a warcaster: Utilizing strong synergies created by the warcaster and complementing the warcaster with overall strong miniature choices. The random adaptation does not accomplish any of these goals and consequently it almost always creates bad armies.

The best performing adaptation method that makes deliberate changes to the query army is the structure transfer adaptation. The average score across all cases is slightly higher than the user queries as seen in figure 5.1, however the difference is not large enough to be statistically significant as shown in table 4.2. An interesting and promising trend can be discerned from the structure adaptation graph in figure 4.3: The structure transfer adaptation is able to improve all query armies that have an average score of 3 or lower. A probable explanation for this result is that weak armies are likely to be improved by simply replacing bad miniatures with a strong structure suited for the relevant warcaster. In most cases it is unimportant to consider what miniatures are being removed, since they are likely to be worse than the ones that are adapted into the user query.
The method is less successful when trying to improve already strong armies, only improving 2 out of the 5 input armies with an average score above 3. These armies are more likely to already include the miniatures the procedure wants to adapt into the user query. As a result it often searches for other, less ideal miniature combinations and attempts to find a place for these. When trying to replace miniatures the procedure only identifies combinations of miniatures that have a similar cost to the structure it wants to adapt into the army. This leads to an issue where the procedure is likely to remove important models from to army in order to make room for the now less than ideal structure it has created.

The final adaptation procedure, reinstantiation adaptation, does not perform as well as the structure transfer procedure. Its average score is actually lower than the average query score and it even fails to improve some of the bad input armies. This is likely an effect of its design and displays a crucial weakness regarding the methodology in this domain. Since the procedure tries to adapt user choices into an assumed strong army it is vulnerable to making two key mistakes:

1. It often removes critical synergistic miniatures from the retrieved army.
2. It is vulnerable to select in bad miniatures from the user army and trying to adapt these into the retrieved army.

In the current iteration of the program neither of these issues are handled appropriately which leads to mistakes being made in several cases.

5.1.2 Specific Cases

This section examines some of the specific user queries the system was asked to improve and looks at why some methods were more successful than others.

1. Case 2 - This user query presents the system with the warcaster Major Victoria Haley. Her army is focused on lightning synergy(firefly, dynamo, storm lances, Laddermore) and spell casting (squire, thorn). The squire + thorn package is considered a must in all Haley2 armies since she is very reliant on using her spells. The original army received an average rating of 5 from the expert group.

   (a) **Null adaptation:** The method identifies a similar army that also utilizes two units of storm lances and the solo Captain Laddermore, while also using thorn and a squire. However the identified army makes some extra changes that weakens the shooting ability of the army and it gets an average rating of 4.

   (b) **Random adaptation:** The procedure makes a lot of bad decisions and ends up removing almost all the synergistic choices, including thorn which is considered a must have for Haley. It gets an average rating of 1.25

   (c) **Structure adaptation:** The technique identifies that stormclads are frequently used with Haley and opts to adapt these miniatures into the user query. However to find room for these models it decides to remove both of the storm lance units which leaves the army with too few units, and too many non-synergistic solos and too many warjacks. The army gets an average rating of 2.75 with expert comments focusing on the wrong support choices.

   (d) **Reinstantiation adaptation:** The procedure decides that it wants to adapt the firefly, a GMCA and some of the storm lances from the user query and into the most similar army which received an average rating for 4. To make room for this the procedure ends up removing thorn which results in a much weaker army than it could have been and it gets and average score of 2.5. This is reflected in the experts comments: " If Thorn had replaced a Firefly and a GMCA it's suddenly a good looking list by and large.", "I mean honestly removing thorn just isn't a good idea. Maybe this list could've been a 3 really but it's such a bad change".

2. Case 6 and Case 11 - These queries are focused on the warcaster Maddox and the original user queries both receive an average expert rating of 3.33. However the quality of the resulting adapted armies vary greatly. This leads to an interesting study where it is possible to examine how the initial similarity assessment affects the output armies.

   (a) **Null adaptation:** For user query 6 the null adaptation identifies a strange army without a clear aim as the most similar army in the case base. Despite being played at WTC-2016 the army lacks a lot of synergies and it received an avg. score of 2 from the expert group. For user query 11 however the retrieval process identifies a very strong army as the most similar to the query. It contains a lot of strong models with relevant synergies and might be as good as it gets for Maddox. It receives an average rating of 5.33 from the experts.

   (b) **Random adaptation:** The random adaptation struggles with both armies and creates weak alternatives that receive avg ratings of 1.66 and 2.0 respectively.

   (c) **Structure adaptation:** For user query 6 the structure adaptation identifies a warjack heavy structure that is commonly used with Maddox armies. It tries to adapt this into the user query and the expert group considers the resulting army to be too heavily invested in warjacks. It receives av avg. rating of 3. The exact same process is repeated for case 11, but this time the user query has fewer warjacks in it originally. This results in an army that is slightly more balanced and it gets an avg score of 3.6.

   (d) **Reinstantiation adaptation:** The most similar case for input army 6 is not very good as explained in (a) and the original query can only be described as okay. This means that the reinstantiation method uses a bad initial army and tries to adapt models from a mediocre army into it. The end result is a mess and the proposed army only gets an average score of 1.3. This does not repeat for input case 11. The user query is okay, but the most similar army is very strong. This implies that the procedure is adapting okay miniatures into a strong army foundation. As a result the proposed solution receives an average score of 3.33.

5.1.3 Research Question Answers

By looking at the results and the evaluation of the system it is now possible to answer the research questions presented in 1.2.

- RQ1. How do various adaptation techniques compare to each other with regards to the warmachine domain? Which ones produces the best armies according to an expert group?

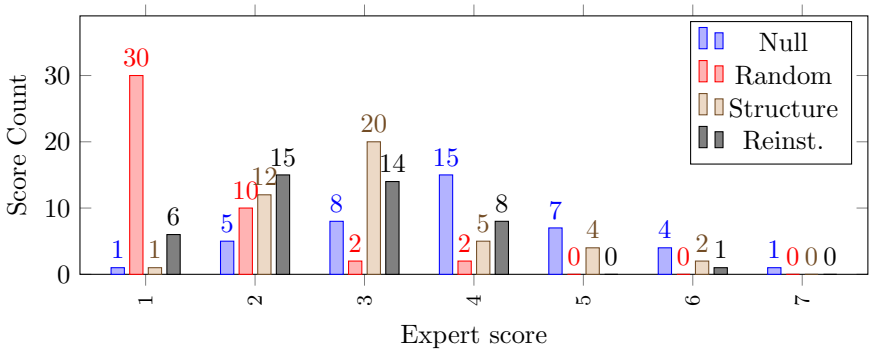


Figure 5.2: Comparing the adaptation procedures

From the results it is clear that a simple null adaptation utilizing similar and already tested armies ends up performing the best. However a system that relies on a strong case base and no adaption will be slow to accommodate to changes to the game, and will always be highly dependent on a consistently updated case base. Another issue with null adaptation in the warmachine domain is the complete lack of priority in keeping miniatures from the user query.

The two adaptation techniques that make concrete and deliberate changes to the user armies are structural and reinstantiation adaptation. The reinstantiation procedure runs into concrete and worrisome issues as explained in 5.1.2 and is outperformed by the structural adaptation. Structural adaptation is able to improve almost all bad user queries and while it struggles to improve stronger armies this could likely be enhanced with further development time and refinement of the method. The changes it makes are currently limited and it usually does not change more than three or four miniatures. This means that the improvements it makes are often quite small, but this could be improved with further development. This is all supported by figure 5.2. The random adaptation gets mostly 1 or 2 ratings, while the null adaptation outperforms all other procedures due to its many 4+ ratings. The structure adaptation outperforms the reinstantiation procedure with more 3, 5 and 6 ratings while also having fewer 1 and 2 ratings.

- RQ2: If the score is 3 or lower, what does the expert group think is the biggest weaknesses of the proposed army? Are there any systematical errors in the armies generated from a specific adaptation technique? Are there any mistakes that are frequent in all armies?

The adaptation techniques have all been tested on thirteen different query armies and evaluated by an expert group consisting of four members. The complete results of the evaluations are presented in Appendix B. For all armies with a rating below 4 the experts were asked to give some reasoning as to why the army was bad and some examples of the feedback can be seen in Appendix C. There were three mistakes that were mentioned frequently:

1. Lack of synergy:

   The lack of synergy is often the result of adapting away synergistic miniatures in favour or others, a mistake that is made frequently by all of the adaptation procedures, except for the null adaptation.

2. Bad miniatures:

   Bad miniatures can appear in armies as a result of all the adaptation procedures. Armies suggested by the **Null adaptation** sometimes contain strange and suboptimal choices. **Random adaptation** frequently adapts bad miniatures into the proposed army while the **structure adaptation** struggles when it attempts to adapt miniatures into an army that already contains the structure it would prefer to add to the army. Finally the **reinstantiation technique** often prioritizes suboptimal miniatures when the input army is inadequate.

3. Non-optimal army compositions and a lack of army focus:

   This issue is a result of the two previous ones. Removing synergistic miniatures and prioritizing bad ones often results in armies that have a lack of focus. The techniques also struggle to understand what kind of miniatures should be adapted into different armies. For example different warcasters are able to utilize different amounts of warjacks. A warcaster that is based around utilizing troops should limit the size of its battle group.

- RQ3: Does the system improve on the initial query?

Some of the adaptation methods are able to improve the user query in most cases. The null adapted armies have an average score that is 1.13 points higher than the average user query and as shown in table 4.2 the difference between the means is statistically significant. The structural adaptation also produces armies that seems to be slightly better than the user queries. As shown in figure 4.3 the technique is able to improve all user queries given a score of 3 or lower. The created armies also have a marginally improved overall score, with an average value that is 0.32 higher than the query armies. However the difference is not large enough to be statistically significant and should not be considered an assurance that the structural adaptation is improving the user queries on average.

Figure 5.3 showcases a comparison of the improvements that are made by the null adaptation and the structural adaptation. They are both able to improve virtually all of the input armies that are rated 3 or lower, but the improvements made by the null adaptation is usually much larger than the one made by the structural adaptation. The null adaptation also makes smaller errors when it struggles with an army.

5.2 Discussion

The discussion attempts to cover some of the promising aspect of the experiments and the adaptation techniques. It also looks at important limitations and possible validity threats regarding the experiments.

5.2.1 Merits

As shown in figure 4.3 the two most promising adaptation methods are null adaptation and structure transfer adaptation. The null adapted armies have the highest average score and the method improves 9 out of the 13 query armies. Structure transfer adaptation also improves 9 out of the 13 query armies, but the improvements made are much smaller, as shown in figure 5.3. Null adaptation is probably the procedure that is be able to achieve the highest scores consistently from an expert group: If a CBR system was specifically designed with achieving high score as its only goal it could limit its case base to one or two pre-designed armies for each warcaster and only return these. The process would quickly become repetitive and relatively useless in a real world setting, but the proposed armies would always be of high quality. Unfortunately the method has several limitation which are discussed in section 5.2.2.

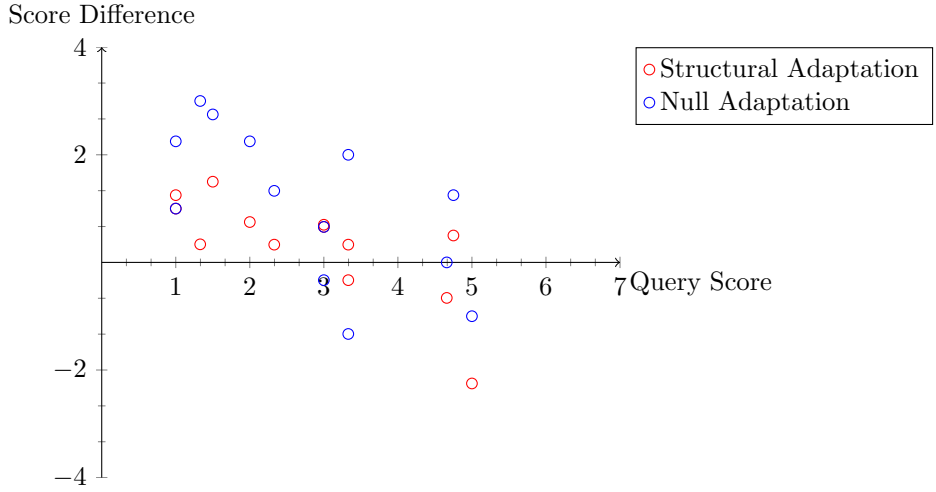


Figure 5.3: Comparison of the improvements made by the null adaptation and the structural adaptation.

Figure 5.3 shows that while the structure transfer procedure does improve most of the input armies, its improvements are much smaller than those made by the null adaptation. However the technique is much more likely to function well in a long lived system and deal with the many changes the game undergoes. Its current performance is not amazing, but it does improve most cases.

Section 5.1.2 showcases both the positive and negative sides of the structure adaptation. For user query 2 and 6 the method struggles to improve the input army because it replaces the wrong miniatures to make room for the structure it deems appropriate. For user query 11 the identified structure is a desirable addition to the input army and the models that are replaced are less important, resulting in a better army. Interestingly there is a lot of room for further improvement and refinement of the technique within the domain, especially if more domain knowledge can be incorporated.

The aforementioned small improvements is a side effect of the procedures current design: it does not want to replace too many of the original miniatures in the query army. However this assumption might be hindering the method. With increased domain knowledge the query army can be analyzed to determine its strength and weaknesses. This information can then be leveraged to determine which miniatures should be replaced. As covered in 2.1.2 there is a lot of information detailing each miniatures capabilities and the system could be extended to utilize this to better compare miniatures. This might enable the system to fine tune already capable input armies while simultaneously allowing it to rarely remove crucial models.

Another possible improvement for the procedure would be to add support for adapting multiple structures into a user query. This would allow the system to identify several different miniature-structures with varying capabilities. Together with an increased ability to understand which miniature should not be removed from the input army the procedure might be able to make much larger improvements. Access to a larger case-base is another possible improvement that would allow the procedure to perform a more thorough analysis of what miniatures are beneficial for an input army.

5.2.2 Limitations

There are several limitations and problems with the adaptation techniques that were tested in the experiments. The two worst performing techniques are the random adaptation and the reinstantiation adaptation: As shown in table 5.1 both of the procedures have a lower average score than the input armies across all the user queries. This struggle to improve the input armies is reinforced by figure 5.4

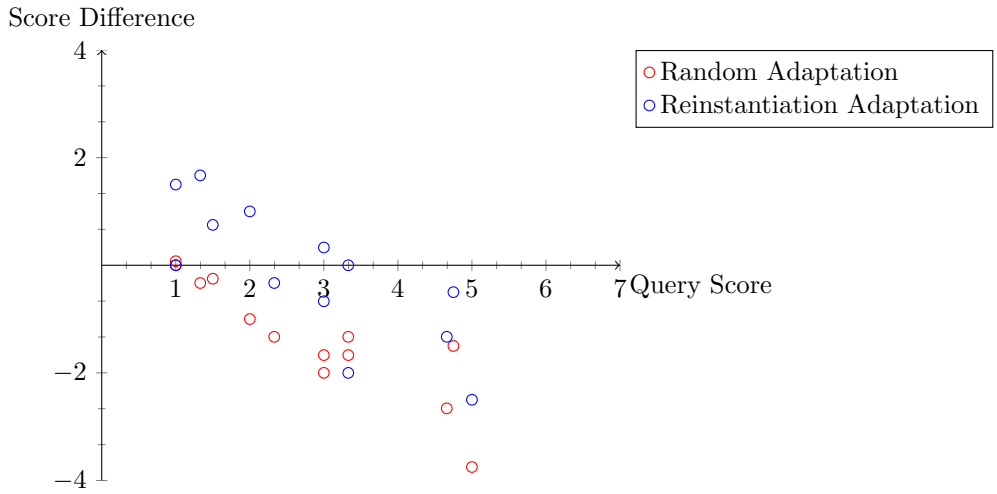


Figure 5.4: Comparison of the improvements made by the random adaptation and the reinstantiation adaptation.

The random adaptation performs terribly which strongly indicates that the domain requires deliberate and focused army changes to improve a user query. It makes the same mistakes as the other techniques, but much more frequently. This is exemplified in section 5.1.2 where the random adaptation makes terrible changes and gets average ratings between 1 and 2.
The reinstantiation adaptation struggles heavily and while it seems to improve bad armies this is often due to luck. The essence of the method is adapting user miniatures into an assumed strong and similar army. Sadly, as shown in 5.1.2, there are several core weaknesses to this approach the became apparent during the experiments.

1. **Trouble improving bad armies:** Since the technique always tries to incorporate large portions of the users army it is prone to keeping bad miniatures that lower the overall strength of the created army.

2. **Removing synergies:** When adapting user miniatures into the most similar army the procedure often removes important synergistic miniatures.

3. **Composition issues:** The method struggles to find a correct balance between warjacks, units and solos.

Performance is not the only way of judging a methods performance. While the null adaptation performs much better than the random and reinstantiation adaptation, there are some limitations that makes the procedure problematic for the warmachine domain.

1. **Reliance on case base:** Since the method does no actually adaptation it is completely reliant on an up-to-date case base. This is problematic in a domain such as warmachine where it is hard to gather data in an automated fashion. This is an issue for all the adaptation methods, but it is especially problematic for the null adaptation.

2. **Vulnerable to change:** As explained earlier Warmachine is a constantly changing game. New models are being released, and old models are updated and changed. The changes can be drastic, and could potentially invalidate the whole case base.

3. **Unable to compose new armies:** Since the procedure only proposes cases that are already in the case base it will never create new armies or discover and innovative army compositions. This also means that even if the system is improved with increased domain knowledge and understanding of synergistic miniatures the null adaptation is unlikely to benefit from this.

5.2.3 Validity Threats

When designing experiments it is important to consider potential validity threats concerning the produced results. The key factor for the experiments performed in this project is the expert group. It conforms to some of the same criteria that define focus groups in social sciences: a small structured group with selected participants. The group is focused on a specific topic, the army building capabilities of the proposed CBR system. Typically focus groups gather and discuss a topic in person but due to geographical constraints this was unachievable. Facebook messenger[1] was used instead for communication and armies were posted to a shared board on a web platform called Trello[2].

[1] https://messengerplatform.fb.com/
[2] www.trello.com

Focus groups was a heavily researched area in the 90s and there some important limitations that should be considered when using them as shown by Morgan [1997] and Krueger and Casey [2000]. There are especially two key points that are highly relevant for this project:

- *Bias and manipulation:* Some participants may be affected by the situation and answer questions with what they think the person in charge wants to hear.
- *'False' consensus:* Strong personalities or opinions may dominate the discussion, while others stay silent.

The most important considerations for this project are the two first points. To ensure that the author does not influence the participants of the expert group he must avoid direct comments on the armies that are presented to the group. A false consensus can be reached by members changing their army assessment after seeing how others have rated army. The only way of avoiding this is by paying close attention when an expert is rating armies and recording the evaluations quickly. In addition to these drawback there are some other distinct disadvantages that are hard to avoid:

1. Experiment repetition - While the experiments that are performed are simple to repeat, their evaluation depends on assembling and having access to talented warmachine players.
2. Biased Evaluation - Most of the members of the expert group already know the author through various online discussions, which might impact their evaluations.
3. Asynchronous feedback - The feedback process is done asynchronously on Trello[3] and the ratings already given by other members can be seen by everyone. This means that if one expert rates an army later than another he might be influenced by the ratings that have already been given.

[3] www.trello.com

5.3 Contributions

The contributions from this project are mostly domain specific, but there are also some general contributions to consider. This is the first time an AI has been utilized in the warmachine domain and it seems clear that there is unexplored potential. This project has served as a first step in exploring the domain, and gives clear indications as to where future research could be interesting.

The project has explored the capability of several adaptation techniques and based on figure 4.3, 5.2 and table 5.1 it is clear that the most promising techniques are structure adaptation and null adaptation. Both methods have their merits and limitations as discussed in 5.2.1 and 5.2.2. Based on this information these techniques can be utilized in other miniature war games with some level of expected success. Compared to the earlier research in the miniature war-gaming domain this focus on the reuse step in the CBR-cycle is a novelty and it explores new avenues related to dealing with updates and changes to a game system.

Warmachine is a knowledge intensive domain where deep domain specific expertise is important. However as the system shows this type of knowledge can be simulated by exploiting the case base and analyzing the cases to identify common trends and important themes, but it is not enough all by itself. Many of the issues that are recurrent in the adapted armies could be alleviated by an increased understanding of how miniatures interact with each other, and what synergies they provide.

5.4 Future Work

Future work could go in a few different directions. There could be increased focus on the warmachine domain and several of the proposed improvements from this paper could be implemented. It would be interesting to explore what an implementation with increased domain knowledge could accomplish. This could then lead into experiments and research that tries to create human readable explanations based on the solutions made by an improved system.

Another possible direction for future work could be to utilize the discussed adaptation techniques in other miniatures games that are similar to warmachine. As mentioned earlier there has already been some research into Warhammer and Warhammer 40k. It would be interesting to see how the adaptation techniques used in this system would function in a different miniature war game.

Warmachine is a knowledge intensive domain and another possibility would be to test the utilized adaptation techniques in other such domains. Especially the structure transfer technique show promise while being a relatively uncomplicated procedure. The method requires little concrete domain knowledge and is able to produce decent results.

The current iteration of the system only considers the input army when it is trying to make adaptations. However warmachine usually requires the player to bring two armies to a game, as explained in section 2.1. This means that every army has to consider what it is capable of playing against, where its strengths lie and what armies it can be paired with. This adds a very interesting dynamic to the army building process and could prove an entertaining challenge for future researcher.

Bibliography

Aamodt, A. (2004). Knowledge-intensive case-based reasoning in creek.

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches.

Alterman, R. (1986). An adaptive planner.

Bain, W. M. (1986a). A case-based reasoning system for subjective assessment. In *Proceedings of the Fifth AAAI National Conference on Artificial Intelligence*, AAAI'86, pages 523–527. AAAI Press.

Bain, W. M. (1986b). A process model of case-based reasoning in problem solving.

Cohen, P. R. and Howe, A. E. (1988). How evaluation guides ai research. *AI Magazine*, 9.

Francis, A. G. and Ram, A. (1993). The utility problem in case-based reasoning.

Fusdahl, T. E. (2016). Case based reasoning for warmachine and hordes.

Hammond, K. J. (1986). Chef: A model of case-based planning.

Kofod-Petersen, A. (2014). How to do a structured literature review in computer science.

Krueger, R. A. and Casey, M. A. (2000). Focus groups: A practical guide for applied research (3rd. ed.).

Lopez De Mantaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M. L., Cox, M. T., Forbus, K., Keane, M., Aamodt, A., and Watson, I. (2005). Retrieval, reuse, revision and retention in case-based reasoning. *Knowl. Eng. Rev.*, 20(3):215–240.

Lozano, L. M., Garcia-Cueto, E., and Muniz, J. (2008). Effect of the number of response categories on the reliability and validity of rating scales. *Methodology*, 4(2):73–79.

Mantaras, R. L. and et al (2005). Retrieval, reuse, revision, retention in case-based reasoning. *The Knowledge Engineering Review*.

Morgan, D. L. (1997). Focus groups as qualitative research (2nd ed.).

Patterons, Rooney, and Galushka (2002). A regression based adaptation strategy for case-based reasoning.

Redmond, M. (1990). Distributed cases for case-based reasoning; facilitating use of multiple cases.

Richter, M. M. and Weber, R. (2013). Case-based reasoning.

Sara Manzoni, F. S. and Vizzari, G. (2007). Substitutional adaptation in case-based reasoning: A general framework applied to p-truck curing.

Schank, R. C. (1982). Dynamic memory: A theory of reminding and learning in computers and people.

Smyth, B. and Keane, M. (1996). Using adaptation knowledge to retrieve and adapt design cases.

Smyth, B. and Keane, M. T. (1995). Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems.

Sormo, F., Cassens, J., and Aamodt, A. (2005). Explanation in case-based reasoning-perspectives and goals.

Strandbraten, G. R. and Petersen, A. K. (2011). Myrmidia - case-based reasoning for warhammer fantasy battle army building.

Zikic, N. (2016). Explanation-aware army builder for warhammer 40k.

Appendix A

A complete listing of the user queries used during the experiments.

Constructed Army one - C1

Haley1
-Centurion
-Charger
-Defender
-Dynamo
Journeyman Warcaster
-Hunter
Max long gunners
-Long Gunners UA
Gun Mage Captain Adept

Constructed Army two - C2

Haley2
-Squire
-Firefly
-Dynamo
-Thorn
Max storm lances
Max storm lances
Lanyssa
Laddermore
Gun Mage Captain Adept

Constructed Army Three - C3

Stryker1
-Ol'Rowdy
-Thorn
-Ironclad
-Gallant
Max trenchers
Trenchers UA
Maxwell Finn
Trench Buster

Constructed Army Four - C4

Stryker3
-Squire
-Stormclad
-Stormclad
Max Storm lances
Max Tempest Blazers
Max trenchers
-Trenchers UA
Min mechanics

Constructed Army Five - C5

Caine2
-Reinholdt
-Stormwall
Journeyman Warcaster
-Charger
Rangers
Alexia1
Max trenchers
-Trenchers UA
Arlan
Gobber Tinker

Constructed Army Six - C6

Maddox
-Squire
-Avenger
-Centurion
-Ironclad
-Lancer
Max Long gunners
Stormblades
Max trenchers

Constructed Army Seven - C7

Siege
-Reinholdt
-Defender
-Defender
-Centurion
Arcane Tempest Gun Mages
Stormblades
Stormblades
Max sword knights
Maxwell Finn

Constructed Army Eight - C8

Haley1
-Ace
-Avenger
-Hunter
Journeyman Warcaster
-Hunter
Max stormguard
Max silver line stormguard
Alten Ashley
Eiryss1
Gorman

Constructed Army Nine - C9

Stryker2
-Squire
-Ironclad
-Ironclad
-Thorn
Journeyman Warcaster
-Charger
Max Trenchers
Max storm lances
Major Harrison Gibbs
Alain Runewood
Ragman

Constructed Army Ten - C10

Stryker2
-Ol'rowdy
-Hunter
-Firefly
-Lancer
Arcane tempest gun mages
Black 13th
Min mechanics
Min blazers
Max stormguards
Maxwell Finn

Constructed Army Eleven - C11

Maddox
-Squire
-Ironclad
-Stormclad
-Thorn
Max storm lances
Max long gunners
Max trenchers
Gobber tinker

Random Army one - R1

Nemo1
-Charger
-Cyclone
-Hammersmith
-Gallant
-Minuteman
Min long gunners
Min precursor knights
Stormsmith Grenadier
Stormsmith storm tower
Min sword knights
Gobber Tinker
Victor Pendrake

Random Army two - R2

Nemo3
-Avenger
-Defender
-Grenadier
-Ironclad
-Lancer
Max precursor knights
Max Tempest Blazers
Anastasia

Appendix B

Expert ratings given to the different input armies and the resulting solution proporsals

Query ID	Adaptation Technique	Evaluations
Case 1	Query Score	[2,2,2,2]
Case 1	Null	[4,5,5,4]
Case 1	Random	[1,1,1,1]
Case 1	Structure Transfer	[3,2,3,3]
Case 1	Reinstantiation	[4,3,3,2]

Query ID	Adaptation Technique	Evaluations
Case 2	Query Score	[5,5,5,5]
Case 2	Null	[4,4,4,4]
Case 2	Random	[1,2,1,1]
Case 2	Structure Transfer	[3,3,3,2]
Case 2	Reinstantiation	[2,3,3,2]

Query ID	Adaptation Technique	Evaluations
Case 3	Query Score	[-,3,2,2]
Case 3	Null	[-,4,4,3]
Case 3	Random	[-,1,1,1]
Case 3	Structure Transfer	[-,3,3,2]
Case 3	Reinstantiation	[-,2,2,2]

Query ID	Adaptation Technique	Evaluations
Case 4	Query Score	[-,3,3,3]
Case 4	Null	[-,3,3,2]
Case 4	Random	[-,1,1,1]
Case 4	Structure Transfer	[-,4,3,4]
Case 4	Reinstantiation	[-,4,4,2]

Query ID	Adaptation Technique	Evaluations
Case 5	Query Score	[5,5,5,4]
Case 5	Null	[7,6,6,5]
Case 5	Random	[2,4,4,3]
Case 5	Structure Transfer	[6,5,5,5]
Case 5	Reinstantiation	[6,4,4,4]

Query ID	Adaptation Technique	Evaluations
Case 6	Query Score	[-,4,4,2]
Case 6	Null	[-,3,1,2]
Case 6	Random	[-,2,2,1]
Case 6	Structure Transfer	[-,3,3,3]
Case 6	Reinstantiation	[-,2,1,1]

Query ID	Adaptation Technique	Evaluations
Case 7	Query Score	[-,3,3,3]
Case 7	Null	[-,4,4,3]
Case 7	Random	[-,2,1,1]
Case 7	Structure Transfer	[-,3,4,4]
Case 7	Reinstantiation	[-,1,2,3]

Query ID	Adaptation Technique	Evaluations
Case 8	Query Score	[1,3,1,1]
Case 8	Null	[5,4,4,4]
Case 8	Random	[1,1,1,2]
Case 8	Structure Transfer	[4,3,3,2]
Case 8	Reinstantiation	[2,3,2,-]

Query ID	Adaptation Technique	Evaluations
Case 9	Query Score	[-,5,5,4]
Case 9	Null	[-,5,5,4]
Case 9	Random	[-,3,2,1]
Case 9	Structure Transfer	[-,3,6,3]
Case 9	Reinstantiation	[-,4,4,2]

Query ID	Adaptation Technique	Evaluations
Case 10	Query Score	[-,1,2,1]
Case 10	Null	[-,5,4,4]
Case 10	Random	[-,2,2,1]
Case 10	Structure Transfer	[-,2,2,1]
Case 10	Reinstantiation	[-,3,3,-]

Query ID	Adaptation Technique	Evaluations
Case 11	Query Score	[-,4,4,2]
Case 11	Null	[-,6,6,4]
Case 11	Random	[-,2,2,2]
Case 11	Structure Transfer	[-,5,3,3]
Case 11	Reinstantiation	[-,5,3,-]

Query ID	Adaptation Technique	Evaluations
Random 1	Query Score	[-,1,1,1]
Random 1	Null	[-,2,2,2]
Random 1	Random	[-,1,1,1]
Random 1	Structure Transfer	[-,2,2,2]
Random 1	Reinstantiation	[-,1,1,1]

Query ID	Adaptation Technique	Evaluations
Random 2	Query Score	[1,1,1,1]
Random 2	Null	[-,3,4,3]
Random 2	Random	[1,1,1,1]
Random 2	Structure Transfer	[2,3,2,2]
Random 2	Reinstantiation	[2,3,3,1]

Appendix C

Examples showcasing the feedback given by the experts.

Score	Expert Feedback
2	not the best picks but the slowdown might make Long Gunners do something at least
4	seems playable, I guess.
5	looks like a solid army, though haley could do with one hunter less and some more support for her cav
5	It looks strong, but Haley seems focus strapped.
4	I sort of like it. It could've used like Laddermore or something but a proper god damn gunline with Stormwall under Haley1 sort of works.
1	WTF
1	no real synergies or focus in the army
1	Dynamo is interesting to some degree but the rest is just a cavalcade of models not synergising with each other. Also no squire.
3	seems tanky but I don't think it plays to her strengths.

Score	Expert Feedback
3	I like the sheer amount of shots you get on feat turn. No UA for LG is an improvement.
2	It's mostly different in some matter of taste-choices.
1	seems random
5	should hit like a truck, no directly poor choices
4	Actually pretty good. Probably a bit vulnerable to counter-builds but decent enough
4	solid army with concentrated firepower and a punch in melee
1	WTF
1	Lots of crap.
2	WEll there's less storm guard in the new variant.. Still very bad
1	really offers nothing for H1, just a mix of units that have no real strengths in the army
4	maybe not the best but could work, not so good against armor
3	Significant improvement
2	At least it's got a good shooting element and a stormwall
3	solid but a bit underwhelming in taking advantage of H1
2	lots of shots but Stormguard and Gorman seems odd
2	Slightly better.
3	can be useful, set up a defensive line and then pour in the shots bit misses a real finisher
2	Some syngergy between THead and lots of electricity immune units but still not great
2	It's better somehow but still doens't really do anything worthwhile. Removed some of the worst unit choices though
2	Some syngergy between THead and lots of electricity immune units but still not great

Score	Expert Feedback
3	Makes use of good feat, Dynamo seems amazing with Nemo3. Stormblades seem out of place.
4	models who work well with nemo, but too many fireflies
2	Dynamo is good. Grenadiers would be cool with Nemo3 considering his ample focus supply but don't work without trenchers. Lancer is a bad Thorn
3	has some models that get work done under nemo 3 but battlegroup has some questionable choices
3	can work, but everything needs to work on it's own, which it kinda can but not with the best of results
4	solid battlegroup, rangers and nyss are feat independant but get work done on feat turn blazers are meh
1	I don't see how this list would work
1	Lots of random crap, no clouds.
4	either more for Alexia to work with or more lances, not both
7	I would play happily that army in a tournament
3	I like the centurion better than Blazers but that's one hell of a BG and no junior. Stryker wants quantity over quality I believe (possible exception: Stormwall) and this goes the opposite way.
2	No thorn, 2x firefly, not right support
2	I mean honestly removing thorn just isn't a good idea. Maybe this list could've been a 3 really but it's such a bad change
2	double firefly does little for the army, s2 has little to work with here

Score	Expert Feedback
2	not sure if this army can use H1's abilities to actually push something to fruition other than the feat turn?
3	Stormwall and Centurion is sort of cool and maaaybe feat can make it all work number of attacks-wise?
4	maybe you can shoot it out...
3	can do some serious concentrated fire on feat turn and h1's spells can give it that chance but no need for 2 tinkers
3	A slight improvement because Blazers are just bad. Finn is stronk with Stryker3 (with fury he can tear apart a number of units with close-to-no-risk). Not a fan of Alten or Sentinel.
2	Slower with less worker models.
2	I like some of the changes, like rhupert and maybe junior, but slower jacks and less work in general.
3	Feels too flimsy
3	I think he can't fuel than many heavies
1	Didn't think you could make it worse, but damn it did!
1	Wow. I can't even. It brought Eiryss1 which has some merit but removed the arc node (and the shooting to punish a KD caster).