Mendelian and Non-Mendelian Ancestral Repair for Constrained Evolutionary Optimisation

by

Amy FitzGerald BSc.

NUI MAYNOOTH

Ollscoil na hÉireann Má Nuad

A thesis presented in fulfilment of the requirements for the Degree of a Doctor of Philosophy (PhD), in Computer Science.

Supervisor: Dr. Diarmuid P. O'Donoghue

Department of Computer Science

National University of Ireland, Maynooth

April, 2013

Declaration

This thesis has not been submitted in whole or in part to this or any other university for any other degree and is, except where otherwise stated, the original work of the author.

Signed: _____

Amy FitzGerald BSc.

Abstract

Evolutionary Algorithms (EA) are excellent at solving many types of problems but are inherently ill-suited to solving constrained problems. Previously there has been four ways to adapt these algorithms to solve constrained problems - pareto optimal strategies, modified representation and operators, penalty functions and repair strategies. This thesis makes significant contributions to the topic of genetic repair and introduces a non-Mendelian repair operator that has been inspired by a naturally occurring genetic repair mechanism in the *Arabidopsis thaliana* plant. Thus, the analogy between EA and natural evolution is extended to incorporate this (still highly controversial) biological repair process.

The first and main objective focuses on Evolutionary **Algorithms**. This thesis adapts this novel genetic repair strategy to an EA to solve two benchmark constraint based problems - specifically permutation problems as this category of problem are often recognised as the most problematic problems for the canonical EA to deal with.

The second objective was more biological, relating to **Evolutionary** Algorithms. A number of algorithmic and parametric interventions were made to the EA, to examine the repair algorithm's performance under more biologically inspired conditions.

This thesis illustrates that non-Mendelian ancestral repair templates outperform their Mendelian counterparts under a wide variety of conditions and also shows that under biologically inspired conditions, the non-Mendelian repair strategy continues to outperform its Mendelian counterpart.

Acknowledgements

My sincere thanks to my supervisor Dr. Diarmuid O'Donoghue for his unwavering support, guidance, help and kindness throughout this process. I would also like to extend my gratitude to Paddy for all of his help, kind words and support from day one. Thanks to my parents - Liv and Myles, my sister Órla and my brothers, nephews, niece, friends and extended family for their support. Thank you also to Dr George Mitchell for his advice at the beginning of my research.

Thank you to Dr Susan Lolle and her lab in the University of Waterloo for hosting me for July 2009. Thank you for the patient explanations, guidance and the wonderful practical experience and inclusion that was shown to me during my visit. Thank you also for the ongoing advice and support throughout my research.

Funding & Support

- **NUIM Bursary:** The first 12 months of my PhD research were funded through a NUIM Computer Science Department bursary. I am very grateful to Dr. Adam Winstanley for the receipt of this scholarship. I appreciate the support given to my by Dr. Susan Bergin during this bursary and throughout my research.

- **IRCSET:** I was awarded three years funding by the Irish Research Council for Science, Engineering and Technology. Without this scholarship my research would not have been possible and for this I am very grateful.

- **ICUF:** The Ireland Canada University Foundation awarded me a scholarship to visit Dr Lolle in the University of Waterloo, Ontario. I appreciate this scholarship and the collaboration that it enabled.

- **Dr Susan Lolle:** During the Summer of 2009 Dr Susan Lolle hosted me in her lab in the Department of Biology in University of Waterloo, Ontario. I greatly appreciate the kind, welcoming accommodating nature of this visit.

- **ACM Scholarship:** In July 2010 I was awarded an ACM travel scholarship to present at GECCO in Portland, Oregon. This scholarship enabled me to travel to the conference which I really appreciate.

Contributing Publications

- Genetic Repair for Optimisation under Constraints inspired by *Arabidopsis thaliana*, Parallel Problem Solving from Nature (PPSN), Amy FitzGerald, D. P. O'Donoghue, Lecture Notes in Computer Science (LNCS 5199), Springer Berlin / Heidelberg, Germany, pp 399-408, 2008

- Investigating the Influence of Population and Generation Size on GeneRepair Templates (CIICT), Amy Fitzgerald, Diarmuid P. O'Donoghue, China Ireland International Conference on Information and Communications Technologies, 2009

- Genetic Repair Strategies inspired by *Arabidopsis thaliana*, Lecture Notes in Artificial Intelligence (LNAI 6206), Amy FitzGerald, Diarmuid P. O'Donoghue and Xinyu Liu, pp 61-71, 2010. The original version of this paper appears in the proceedings of AICS 2009 where it was also presented.

- Biologically Inspired Non-Mendelian Repair for Constraint Handling in Evolutionary Algorithms, The Genetic and Evolutionary Computation Conference (GECCO) Constraint Handling Workshop (chaired by Dr. Carlos Coello Coello, Dr. Dara Curran and Dr. Thomas Jansen), Amy FitzGerald, Diarmuid P. O'Donoghue, Portland, Oregon, pp 1817-1825, 7-11 July 2010

- Investigating the Effect of Stochastic Replacement Rates on GeneRe-

pair Templates (CIICT), Donagh Hatton, Diarmuid P. O'Donoghue and Amy FitzGerald, Proceedings of China-Ireland International Conference on Information and Communications Technologies (2010)

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Getting Back to Nature & Biomimicry

Biomimicry or biomimetics is the act of taking inspiration from nature in order to solve human problems. Many different leaders from very different backgrounds have looked to nature when trying to solve problems. Examples can be found in many fields from scholars to poets and gurus to scientists where leaders in the field have proclaimed that nature is the best teacher humans have available to them. In an article entitled Learn from nature (Khuvung 2009) Lolano P. Khuvung wrote

> *"Since we too are a species on this earth and not invaders, we need to create products and processes that follow natures principles. "*

Khuvung was clearly stating that as the problems we are trying to solve are in the environment in which we exist, we must look to this environment for solutions. The poet John Celes writes

"Mother Natures the best teacher we know, Teaching both young and old without a fee; Through flora, fauna and all creatures, oh! Her lessons being unique, naturally. "

in his poem *Nature's the Best Teacher*. While the poet Celes comes from the literary world and Khuvung is a scientist, the meaning of his poem reflects the sentiment of the scientific article. In the same vein the guru Sathya Sai Baba is credited with the much quoted proverb

"Nature is the best teacher "

We can see that in the literary, scientific and spiritual field there is a strong belief that we must learn from nature and look to nature to solve problems. This is not a new phenomenon as scientists have looked to nature for guidance when solving problems for many years. The design of solutions inspired by nature is known as *Biomimicry*. From the design of tongue and groove which originates from the reproductive organs of humans to the invention of Velcro which is modelled on the cockleburs plant (See Figure 1.1) many of today's inventions originate from nature [1].

We do not need to examine the pickaxe for much time before we see the similarity between this tool and the woodpecker bill. Many more examples of inventions inspired by nature are available and by now we can see that a wide variety of different disciplines in our academic society are open to looking to nature for problem solutions. One such discipline that often looks to nature for problem solving techniques is the area of Computer Science. The field

[1] This information were found on the Designed by Nature slideshow at http://acrodshops.info/resource/eng06.sci.engin.design.biomimicry/

Figure 1.1: The Cockleburs Plant. Image was taken by Franco Folini and is used under the terms of the GNU Free Documentation License (Version 1.2 or later)

of artificial intelligence (AI) by its very nature mirrors the intelligence of human beings in an artificial way. The definition of AI (Poole, Mackworth & Goebel 1998) is said to be

> *"the study and design of intelligent agents "*

These intelligent agents can then perceive their environment. These agents may be able to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.

A sub section of artificial intelligence is *Evolutionary Computation*. Negnevitsky (Negnevitsky 2005)(Page 14) states that:

> *"Evolutionary computation works by simulating a population of individuals, evaluating their performance, generating a new population, and repeating this process a number of times. "*

The four techniques which form evolutionary computation are genetic algorithms, evolutionary strategies, genetic programming and evolutionary programming. Evolutionary algorithms (EA) is the name given to genetic algorithms that do not necessarily use binary representation. These algorithms are ill-suited to dealing with constraints so an adaptation called evolutionary optimisation (EO) was formed. Thus EO was evolved to solve constraint based problems. Solutions to constraint based problems must satisfy a defined list of conditions (or constraints) in order to be valid solutions.

In this thesis I will examine the analogy between biology and evolutionary algorithms(EA) that is often used to describe EA. I will then attempt to extend this existing analogy by updating it with a new biological discovery. This new biological discovery of a repair mechanism in the *Arabidopsis thaliana* plant[1] was first published in 2005 (Lolle, Victor, Young & Pruitt 2005) and we will see further details of this mechanism in Chapter 3. In this thesis I will investigate the use of this repair mechanism when implemented as part of an EA. Our objective is to investigate whether this repair mechanism will enable evolutionary algorithms to solve constraint based problems in a biologically inspired way. Our second objective is to perform stress or reliability tests of the mechanism using biologically inspired parameters in conjunction with this biologically inspired EO.

[1]Also suspected to exist in soybean - Personal communication with Dr Susan Lolle in University of Waterloo, Ontario, Canada, which is part of ongoing communication and collaboration

Figure 1.2: The *Arabidopsis thaliana* - A non-Mendelian repair mechanism was discovered in this plant in 2005

1.2 The Current Field of EA

Later in this thesis I will examine the field of evolutionary algorithms. I will illustrate the different techniques currently available to enable evolutionary algorithms to solve constraint based problems. We will clearly see how many of the techniques available are either problem specific or are unreproducible due to lack of detail in supporting literature. These techniques are also not biologically inspired. While biological inspiration is not generally a concern for computer scientists, this thesis investigates extending the analogy with biology which is often used to describe evolutionary algorithms and so tries to maintain this analogy by using a biologically inspired repair mechanism proposed by biologist Susan Lolle (Lolle *et al.* 2005). This repair mechanism

is highly controversial and, at the time of this publication, not yet supported by other findings although active research is ongoing.

1.3 Analogy between Natural Evolution and Evolutionary Computation

Analogy (sometimes referred to as metaphor) is said to be a cognitive process of transferring information or meaning from a particular subject (the source analogue) to another particular subject (the target) (Veale, O'Donoghue & Keane 1999). In Chapter 3 I will introduce the subject of analogies. I will look at the theory behind analogies using some simple examples to show how complex analogies can be formed. I will describe the existing analogy between natural evolution and EA, which is often used to describe the EA field. I will then show how adding this new natural genetic repair process to the analogy, results in a new computational process of genetic repair.

1.4 Extending the Analogy using the *Arabidopsis thaliana* and GeneRepair

In Chapter 4 I will examine further the non-Mendelian repair process found in the *Arabidopsis thaliana* plant which was first published in 2005 (Lolle *et al.* 2005). I will first attempt to use this newly proposed repair process to extend the analogy between natural evolution and evolutionary algorithms to include a repair mechanism called GeneRepair. I will fully explain GeneRepair using

a wide variety of examples and we will see whether it enables evolutionary algorithms to solve constraint based problems in a natural way in keeping with original analogy. This thesis has two objectives;

1. Improve EO: the first objective of this thesis is to investigate whether non-Mendelian template repair can be used to enable EO to solve constraint based problems and if so, how does it compare to a popular penalty function approach.

2. Perform stress or reliability tests on the mechanism: I will compare the use of Mendelian and non-Mendelian repair mechanisms. In doing this I will investigate whether non-Mendelian GeneRepair outperforms its Mendelian counterpart under more "biologically"inspired parameters as in the controversial theory proposed by Lolle *et al* (Lolle *et al.* 2005).

1.5 Simulated Biological Conditions

It is important to clarify that in this thesis, what we term as "biological experiments"are still computational evaluations and should not be confused with systems biology or related disciplines. These experiments arise from the desire to identify if this computational optimisation can shed any light on ancestral repair when the parameters more closely resemble those found in biology. These experiments were also motivated by feedback from a number of collaborators who wished to find **any** evidence, either positive or negative, that might shed some light on Lolle *et al* (2005).

Three specific modifications to the evolutionary optimisation model pre-

sented lie at the heart of these experiments. Firstly, a much larger problem size is used to reflect the much longer genome of the *Arabidopsis thaliana* plant. Secondly, the far lower mutation rate is experienced under natural evolution. Finally, some cyclic redundancy is removed in the representation of Travelling Salesman Problem(TSP), which is explained in Section 2.3, solutions as this does not appear to have an analog in the biological domain (See Chapter 3).

Thus, throughout this thesis and especially in section 5.6, "biological experiment"refers specifically to these two interventions and their impact on relative fitness of the Mendelian and non-Mendelian repair templates. The primary focus of this section is to assess which of the Mendelian and non-Mendelian templates appear to work most effectively under these new operating conditions. As far as the author is aware, these results presented in Section 5.6 represent the best "in silico"evaluation of Lolles non-Mendelian genetic restoration hypothesis (Lolle *et al.* 2005).

1.6 Implementing the Repair Mechanism

In this thesis I will discuss the field of evolutionary algorithms. I will explain the concept of analogy and illustrate the analogy between natural evolution and computer science that summarises the field of evolutionary computation. I will then extend this analogy to include the recent biological finding of a repair mechanism in the *Arabidopsis thaliana* plant (Lolle *et al.* 2005). Following from this I will write an EO algorithm embodying this extended understanding of inheritance and genetic repair.

In Chapter 5 I will illustrate the experiments I conducted in order to compare GeneRepair to another constraint handling mechanism. I will then show all of the different experiments I carried out to investigate the influence of EO parameters on the effectiveness of GeneRepair. During these experiments I constantly compare the use of different ancestral repair templates in order to explore the analogy with natural repair in the *Arabidopsis thaliana* which has been proposed to be non-Mendelian. Chapter 5 investigates the theory discussed in Chapter 3 and 4 and suggests parameters to use when using this technique.

Lolle's theory of non-Mendelian repair in the *Arabidopsis thaliana* plant is highly controversial and as of this date unconfirmed by any other publications. This first publication of experimental evidence suggesting the existence of a cache of genetic information appeared on March 24th 2005 (Lolle *et al.* 2005). This finding was quickly thrown into the media spotlight and appeared on the front-page of the New York Times and The Washington Post causing widespread debate among the scientific community. Two different labs,(Peng, Chan, Shah & Jacobsen 2006)(Mercier, Pelletier, Jolivet, Drouaud, Durand, Vignard & Nogu 2008) who were unable to reproduce the results, claimed that the original findings were made through experimental error but this did not decrease the interest among the scientific community. In 2008 The Scientist ran the cover story "Mendel Upended"(Gawrylewski 2008) and in 2010, five years after the original publication, the 2005 paper was ranked No. 1 paper in the all time top ten by Faculty of 1000(Hopkins, Chang, Lai, Doerr & Lolle 2011). In this thesis I will investigate whether

this proposed technique is successful when incorporated into EO in the field of Computer Science regardless of its performance in the *Arabidopsis thaliana* plant.

1.7 Addressing the Issues Associated with Current Constraint Handling Mechanisms

In Chapter 2 I examine the current findings in the field of Evolutionary Optimisation (EO) with a focus of constraint handling mechanisms. I will show that the repair mechanisms summarised are not biologically inspired or are impossible to reproduce or problem specific. These traits are disadvantageous as methods which are too problem specific cannot be easily implemented for a wide variety of problems. Methods which are difficult to reproduce due to lack of information in the publication are not easily implemented. Non-biologically inspired methods are suitable when the user is not interested in the analogy between nature and EO. In this thesis I am concentrating on extending this analogy and therefore investigate whether a biologically inspired repair strategy can be used to enable EO to produce valid solutions to constraint based problems. Chapters 3 and 4 show the biological roots of GeneRepair, mirroring the repair mechanism found in the *Arabidopsis thaliana* plant (Lolle *et al.* 2005). In Chapter 5 I will show the experiments which illustrate the different parameters to use with GeneRepair and give full and explicit details to investigate the influence of these parameters on the ancestral repair process. In Chapter 6 I address the final weakness of

many previously used techniques: Problem Specificity. In this chapter I will revisit the experiments from Chapter 5 using a different problem. I will show that the effectiveness of parameters and ancestral templates is not altered by the change in problem and that this technique may be applied to a different constraint based problems. This enables EO to be used on a wide variety of constraint based problems in a biologically inspired way. In this thesis I will focus particularly on permutation problems.

1.8 Conclusion

The old proverb

> *"Nature is the best teacher "*

attributed to the guru Sathya Sai Baba has been supported by poets and scientists alike from many different disciplines. We can see the natural inspiration behind many inventions from Velcro to the pickaxe. In this thesis we present a repair mechanism inspired by nature to enable EO to produce valid solutions to constrained problems. The analogy between natural evolution and EA is often used to describe EA. In the years following the introduction of EA computer scientists made great advances in the world of evolutionary computation to enable this problem solving technique to solve previously problematic constraint based (and other) problems. Some of these advances broke the analogy between natural evolution and EA, while others were problem specific and so not widely usable. This thesis provides a wide review of the advances made in Computer Science and explains how they

enable EO to solve constraint based problems. It then goes on to explain a major and controversial biological advance made in 2005 (Lolle *et al.* 2005). I incorporate this biological advance into the analogy to create GeneRepair in the EO. I investigate whether this GeneRepair mechanism enables EO to produce valid solutions to constraint based problems. I outline how it can be used for "general"constraint type problems. I will specify how it is applied to the permutation problems TSP and CVRP - solving both with the very same repair mechanism. This being explored in great detail. This thesis introduces a biologically inspired repair mechanism for EO and explores various parameters and values that impact the effectiveness of this mechanism.

Chapter 2

Literature Review

2.1 Introduction

This chapter gives an introduction into the subject of Evolutionary Algorithms (EA). We look at the beginning of EA and how they work. I go on to introduce constraint based problems and outline a (standard) sample problem to be used throughout this thesis. I will show the different mechanisms EA currently use to handle these constraint based problems. Finally I will introduce a biological inspired method to handle constraint based problems using EA. It is this biologically inspired method for handling constraints that will be the focus of this thesis. I will use high level analogies in order to illustrate the properties of this method.

2.1.1 Structure of the Chapter

This chapter has four sections. The first section gives an introduction to and an explanation of Evolutionary Algorithms. The second section introduces the sample problem which will be used to illustrate the different operators in this chapter. The third section describes the different methods currently used within EA to handle constraints. This section will review the current methods and show how they differ to the method being explored in this thesis. The final section will give a conclusion on all of the above sections. This thesis introduces a new non-Mendelian repair mechanism and investigates whether it can be used with evolutionary optimisation to solve constraint based problems. This chapter will therefore give an introduction to the field of evolutionary algorithms and the current methods which exist to solve constraint based problems.

2.2 Introduction to Evolutionary Algorithms

Using technology evolutionary algorithms (EA) could be seen to mimic the organic evolutionary process to solve problems. Though the early study of evolution is strongly linked to Charles Darwin (12 February 1809 - 19 April 1882) the word evolution dates back to the 17th century. In the 18th century momentum grew for this idea as natural philosophers Pierre Maupertuis and Erasmus Darwin began to suggest the idea of proto-evolution (Terrall 2002). The next development took place when Jean-Baptiste Lamark put forward the theory of transmutation of species. Radical ideas were being suggested

and the stage was set for an in depth look at evolution. It is not surprising therefore that at this time two separate people were working on the idea of natural selection. These two scientists were Charles Darwin and Alfred Russel Wallace. Charles Darwin continued to develop his hypothesis and in 1859 published the first edition of *"On the Origin of the Species"*. This book explained that natural selection is nature's algorithm for evolution. Natural selection is the natural law that enables evolution by giving the fittest individuals in a population the highest chance of procreation. For example, if a certain animal can run faster than its peers then it is more likely to run away from predators and survive attacks. The chance of this animal procreating is higher than that of a slower peer as that peer is more likely to be killed by a predator. Therefore this ability to run fast will most likely be passed from this generation to the next and this animal may therefore evolve into a fast runner. If environmental conditions meant that a slower run was more favourable (probably through conserving its energy) then the slower of the two animals would be more likely to survive and so over time this breed of animal may evolve to be slower runners. This is a simple example of natural selection. Therefore natural selection is the non-random process by which biological traits become more or less common in a population as a function of differential reproduction of their bearers and is a key mechanism of evolution. *Survival of the fittest* is a simpler idea meaning better adapted for the immediate, local environment. In this case the word *fittest* means most suitable for the given environment.

2.2.1 The Evolutionary Algorithm Family

Evolutionary algorithms could be seen to mimic this evolutionary process to develop solutions to problems. A population of solutions is evolved in a similar way to that of natural evolution. The origins of GA and ES can be attributed to Holland and Reichenberg (Holland 1975), (Rechenberg 1971). Solutions that are fittest in relation to the given environment or problem go forward to produce further generations and eventually a suitable solution is produced. While there has been many adaptations to the canonical EA the standard implementation is based on the Genetic Algorithm . Fogel (Fogel 1994) describes this as

1. The problem to be addressed is defined and captured in an objective function that indicates the fitness of any potential solution

2. A population of candidate solutions is initialised subject to certain constraints

3. Each chromosome in the population is decoded into a form appropriate for evaluation and is then assigned a fitness score

4. Each chromosome is assigned a probability of reproduction

5. A new population is formed and mutated

6. The process is halted if a certain objective is met otherwise the process proceeds from Step (3) where the new individuals are scored and the cycle is repeated

2.2.2 The Basic Evolutionary Algorithm

In the previous section I have shown the six steps outlined by Fogel (Fogel 1994). By studying these steps we can say that once the problem is understood and the fitness function has been defined, 5 steps are followed to implement the algorithm. There 5 main steps are:

1. Generate Initial Population

   In the first step of the EA a population of possible solutions is generated. As this first set of solutions is randomly created they are not necessarily strong (or fit). This population is the first generation of the Evolutionary Algorithm and the number of solutions generated corresponds to the size of the population. This *Generate Initial Population* step is only performed once.

2. Select Fit Solutions

   A number of fit individuals (solutions) in the population are selected and these will be used to create the next generation. There are a vast number of selection methods available and one standard selection method is *tournament selection* (Blickle & Thiele 1996). Using this method a number n of random individuals are selected and compared to each other. The fittest individual is selected for the new population. This is repeated until the population is sufficiently full. This is based on the selection method used by animals competing to breed in the wild where two or more males may fight for the female or vice versa. Another selection method is the *truncation selection* method. Using

this method the population is re-ordered according to the fitness of the individuals. Some proportion $(1 \setminus p)$ of the fittest individuals are selected and reproduced. For example if $p = 2$ then the top of half (the fittest half) of the population is copied into the bottom half of the population resulting in the least fit half of the population being replaced.

3. Crossover or Recombination

   Having identified the breeding population in Step 2 we now produce new individuals. This is the reproductive step of the EA. In the EA crossover is carried out when a point is chosen and the portion of the first individual (Parent1) to the left of the point is inserted to the left most positions of the offspring and the portion of the second individual (Parent2) to the right of the point is inserted into the rest of the offspring. This method produces one offspring from two parents. In order to produce two offspring from two parents the remaining unused part of Parent2 is inserted into the leftmost position of the second offspring and the remaining used part of Parent1 is inserted into the unused part of the second offspring. The point used to subsection the parents can be uniform or chosen randomly. The crossover method described here is single point crossover producing two offspring as described by Negnevitsky (Negnevitsky 2005) (Reference page 226-227).

4. Mutate

   The final step of the EA is mutation. Mutation alters a small pro-

portion of genetic information in some individuals in order to preserve diversity. Mutation is the second chance for new individuals to be introduced into the population. As the name suggests mutation causes an individual to be changed to avoid local maxima/minima, depending on the choice of mutation method being used.

A simple example of mutation in a binary representation involves flipping a random 1 to a 0 or vice versa creating a new mutated individual. There are many different methods of mutation. A simple method of mutation is Swap Mutation, which is often used for permutation problems. This method involves swapping two random bits in the individual to be mutated.

5. Repeat until termination requirement met

   Steps 2 - 4 are repeated until some termination requirement is met. Examples of this requirement are: a set number of generations being reached, a set amount of time or until a certain solution is reached.

The way in which these solutions are monitored and compared is by using a fitness function. Atmar (Atmar 1994) suggested that a singular measure of evolutionary fitness is the appropriateness of the species' behaviour in terms of its ability to anticipate its environment. This fitness function is often the most difficult part for the creator of the EA to write. This function calculates absolute fitness for each of the individuals and they can then be compared upon this value. Sometimes we evaluate fitness from the phenotype, but

as we shall later see, some problems (& representations) allow evaluation directly by examining the genotype.

2.3 The Sample Constraint Problem

We will now look at the application of this canonical GA on a sample problem of the Travelling Salesman Problem (Reinalt 1991). This will serve two purposes. Firstly it will illustrate the operation of the GA. Secondly this constraint problem will highlight how invalid solutions may be inadvertently generated - something that shall concern us for the remainder of this thesis.

The Travelling Salesman Problem originates with a travelling salesman who has a set number (n) of cities to visit. He must visit each city once and at the end of the tour he must return to the first city he visited. The tour length of fitness of the solution is total distance travelled by the salesman. A problem constraint is a condition that a solution must meet. The constraints for this problem are that each city must be visited once with no duplication or omissions and the after the last city the salesman must return to the start city. These are "hard"constraints (Luke 2009) as solutions that disobey these constraints are invalid. (This is opposed "soft"constraints where it is preferable that each solution obeys the "soft"constraints but disobeying does not render the solution invalid.)

A small example of this problem would be a TSP with the cities Castlebar, Waterford, Tralee, Galway, Dublin and Omagh. In Figure 2.1 we can see a sample solution to this problem where the red arrow indicates the start and end city. For simplicity each city is replaced with an integer as follows:

Castlebar - 0

Waterford - 1

Tralee - 2

Galway - 3

Dublin - 4

Omagh 5

So a sample solution which is (Castlebar, Waterford, Tralee, Galway, Dublin, Omagh) would now be represented by (0, 1, 2, 3, 4, 5)

The initial population is filled with these random solutions until it is full and this is the *Generate Initial Population* step from Section 2.2.2. The individuals that will be used to create the next generation are now chosen using a selection method and this is the *Select Fit Solutions* step. These individuals are now crossed over to create the next generation using the *Crossover or Recombination* step from Section 3. Figure 2.2 illustrates the single-point implementation of the crossover operator.

The next generation has now been created with a new population of individuals (or solutions). A set amount of these solutions are then mutated according to the preset mutation rate. There are many different mutation methods that can be used for this *Mutate* step. The method I have used is Swap Mutation as illustrated in Figure 2.3. Swap mutation randomly selects two elements (locii) according to some mutation rate and swaps the values at these locii. In Figure 2.3 we can see that City 4 was swapped with City 2 (as illustrated in green).

The next step in the algorithm is to *Repeat (until termination requirement*

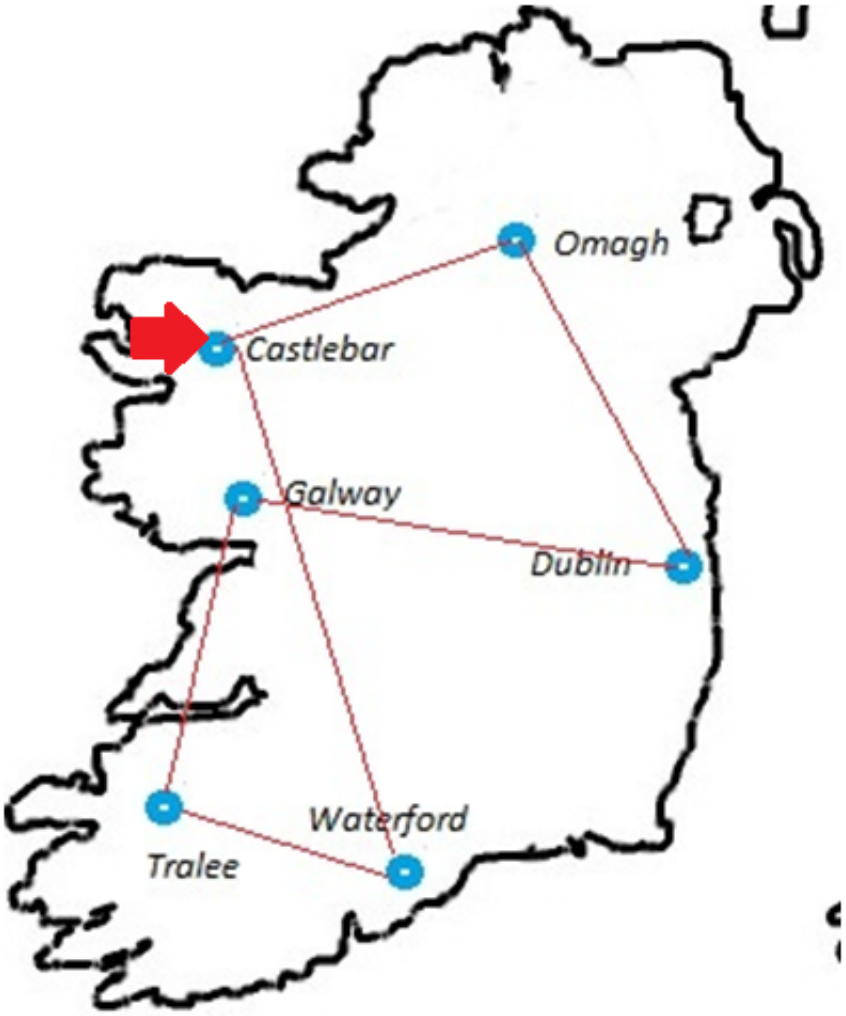

Figure 2.1: TSP Sample Tour

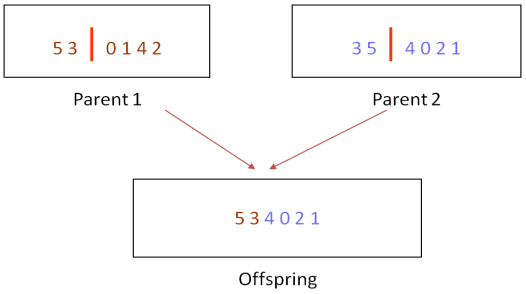

Figure 2.2: Sample Crossover

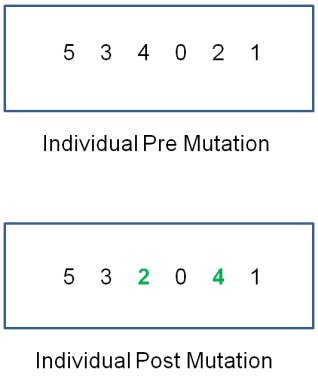

Figure 2.3: Sample Swap Mutation

met) steps 2 to 4 therefore selection, crossover and mutation are repeated until some termination condition is satisfied. This objective may be a set amount of time has passed or a set number of generations have been created.

2.4 Evolutionary Algorithms with constrained problems

2.4.1 The Problem with Evolutionary Algorithms handling constraints

Evolutionary Algorithms are excellent at solving difficult problems that would otherwise take high computing power and prohibitive amounts of time when brute force is used. Where EA are challenged however is when they are used to solve constraint based problems. These problems call for a number of constraints to be satisfied in a given solution and this can cause difficulty for EA. The reason for this difficulty is that EA create a wide and varying range of solutions and lacks a method to ensure that these solutions obey the problem constraints. The canonical EA creates diverse solutions and avoid reaching a local maxima by keeping the population spread across the feasible search space. The fitness function assumes that presented solutions are valid or feasible. With constraint problems some portion of population may represent infeasible solutions which are undesirable.

In order to produce solutions that obey specific problem constraints the canonical EA (as shown in Section 2.2.2) must be adapted. This adaptation

can be done in a variety of ways. At present there are 4 main schools of thought on this subject. The EA can adapt to pareto optimal strategies, can be tweaked in order to use modified operators, it can incorporate a penalty system or it can use a form of repair on invalid solutions. Each of these methods are explained and reviewed in the following subsections.

2.4.2 Pareto Optimal Strategy

The first strategy used to enable EA to find solutions to constraint based problems is called the Pareto Optimal Strategy. This approach separates constraint violation from the fitness function. There are four different constraint handling techniques (Salcedo-Sanz 2009) that fall under the umbrella of the Pareto Optimal Strategy. These are co-evolution, superiority of feasible points, behavioural memory and multi-objective optimisation techniques (Coello Coello 2002). We will now briefly look at each of these in turn.

Co-Evolution

Co-evolution is a technique where two populations are evolved (Coello Coello 2002). The first population is unlike other populations in that it contains all of the problem constraints. The second population contains all of the individuals as normal where the individuals represent valid solutions as well as invalid solutions. An invalid solution disobeys at least one problem constraint. The co-evolutionary strategy is based on the predator-prey analogy where pressure on individuals in one population depends on the fitness of the individuals in the other population. In the second population fitness is cal-

culated by counting the number of constraint violations in each individual. If an individual has few constraint violations they will have a high fitness value but individuals that break many constraints will have a low fitness value. In the first population the fitness of individuals is calculated by counting how many times a constraint is violated. An individual with high fitness in this population is a constraint that is broken by many individuals. *Encounters* will occur between individuals of each population (problem solutions and constraints). Each individual has an encounter history which is stored by that individual and the fitness of each individual is calculated based on the past n encounters where n is a predefined integer. The idea is to concentrate on the constraints that are more difficult to obey by increasing the fitness of these constraints. Paredis (Paredis 1994) theorises that this approach is similar to the self adaptive penalty function, as the relevance of constraints can change over time. An advantage of this approach is that it is efficient because not every constraint needs to be checked at each iteration. However a disadvantage is that stagnation may occur if all of the constraints, or the majority of the constraints are difficult to obey. This approach has not been extended to numerical optimisation problems.

Superiority of Feasible Points

The second pareto-optimal strategy we shall look at is the superiority of feasible points. Powell and Skolnick (Powell & Skolnick 1993) describe the "Superiority of Feasible Points "technique as a map of evaluations of feasible solutions into the interval $(-\infty, 1)$ and infeasible solutions are mapped into

the interval (1, ∞) (also summarised by Coello Coello (Coello Coello 2002). Following this individuals are evaluated using

$$fitness(\vec{x}) = \begin{cases} f(\vec{x}) & \text{if feasible} \\ 1 + r(\sum_{i=1}^{n} g_i(\vec{x}) + \sum_{j=1}^{p} h_j(\vec{x})) & \text{otherwise} \end{cases}$$

$\vec{x}$ is scaled into the interval ($-\infty$, 1), $g_i(\vec{x})$ and $h_j(\vec{x})$ are scaled into the interval (1, ∞), and r is a constant penalty factor. The constraint violation and the objective function remain separate when the individual is invalid. Linear ranking selection (Davis 1991) was used to enable slow convergence in early generations and enforce convergence in later generations by increasing the number of occurrences of the highest ranked individuals. The assumption relied upon by this approach is the superiority of valid individuals over invalid ones. In cases where the ratio between the feasible region and the whole search space is too small this technique will fail unless a feasible point is introduced by the initial population (Michalewicz 1996). This could happen if the constraints are difficult to satisfy.

Powell and Skolnick's approach was changed slightly by Deb (Deb 1998) by evaluating individuals using

$$fitness(\vec{x}) = \begin{cases} f(\vec{x}) & \text{if } g_i(\vec{x}) \geq 0, \forall_i = 1, 2, \ldots, n \\ f_{worst} + \sum_{i=1}^{n} g_i(\vec{x}) & \text{otherwise} \end{cases}$$

where f_{worst} is the objective function of the worst feasible individual solution in the population and $g_i(\vec{x})$ refers to the inequality constraints Deb

has transformed equality constraints to inequality constraints. If there are no feasible solutions in the population then f_{worst} is set to zero. Using binary tournament selection Deb goes on to compare each set of two individuals using the following rules:

- A feasible solution is always preferred over an infeasible one
- Between two feasible solutions; the one having a better objective function value is preferred.
- Between two infeasible solutions; the one having a lower number of constraint violations is preferred.

The fitness of feasible individuals is equal to their objective function value. The use of constraint violation in the comparisons aims to push infeasible individuals towards the feasible region. Deb normalised the constraints to avoid bias as they are usually non-commensurable. This second approach (Deb 1998) does not require a penalty factor due to the pairwise comparisons carried out during selection. This approach instead uses niching which means that the initial focus is on finding feasible solutions and then techniques are employed to maintain diversity as theses solutions approach the optimal.

This approach has problems maintaining diversity in the population and the use of niching and high mutation rates is necessary to combat this. This does mean however a computation cost that can lead to slower execution.

CONGA

The third version of this Pareto Optimal Strategy is called CONGA (COnstraint based Numeric Genetic Algorithm) and was proposed by Hinterding and Michalewicz (Hinterding & Michalewicz 1998). In the first of n phases, this approach concentrates the search on finding feasible individuals and instead of using the the objective function the information about constraint violation is used. The second phase is to fine tune the fittest of the increasing number of feasible solutions. Two selection methods are used - one to select an individual to mutate or to act as the first parent and one to select a mate for that parent. This second selection method selects a mate that will "compliment"the first parent. This means that the second parent should obey the constraints that the first parent does not. This idea came from Ronald (Ronald 1995) but was changed slightly so that the *complimentary* parents will be more likely to produce feasible individuals through crossover. This version relies on the same assumption made by Powell and Skolnick (above) and so shares the same disadvantage that in cases where the ratio between the feasible region and the whole search space is too small this technique will fail unless a feasible point is introduced by the initial population (Michalewicz 1996). This could happen if the constraints are difficult to satisfy. It is also difficult to maintain diversity in the population as the tournament selection strategy may introduce a high selection pressure (Coello Coello 2002).

Behavioural Memory

The technique of behavioural memory was originally proposed for unconstrained optimisation (DeGaris 1990) and was extended by Schaoenauer and Xanthakis (Schoenauer & Xanthakis 1993) to handle constraints. The technique works by handling constraints sequentially using the following algorithm:

1. Start with a random population of individuals

2. Set j = 1 (j is the constraint counter)

3. Evolve this population to minimise the violation of the j-th constraint, until a given percentage of the population (this is called the flip threshold Φ) is feasible for this constraint. In this case

$$fitness(\vec{x}) = M - g_1(\vec{x}) \tag{2.1}$$

   where M is a sufficiently large positive number which is dynamically adjusted at each generation.

4. j = j + 1 (Once 3 above is satisfied)

5. The current population is the starting point for the next phase of evolution, minimising the violation of the j-th constraint,

$$fitness(\vec{x}) = M - g_j(\vec{x}) \tag{2.2}$$

   During this phase, points that do not satisfy at least one of the 1st, 2nd, ... (j-1)-th constraints are eliminated from the population. The

condition required to stop this stage of the algorithm is again the satisfaction of the j-th constraint by the flip threshold percentage Φ of the population.

6. If j $<$ m, repeat the last two steps, otherwise (j = m) optimise the objective function f rejecting infeasible individuals.

When a certain percentage of the population satisfies a constraint an attempt to satisfy the next constraint is made and so on. The last step of the algorithm involves the death penalty, so that invalid individuals are not present in the population (We look at the death penalty in Section 2.4.4). This step is only reached however when the majority (or a certain percentage defined by Φ) of the population consists of valid individuals. The order of the constraints can greatly influence the final solutions produced by this approach. This approach also has a high computational cost which may not be justified when there are many constraints to obey. Another disadvantage of this approach is that the flip-threshold introduces another arbitrary parameter that must be tuned.

Multiobjective Optimisation

We shall now briefly examine Coello Coello's (Coello Coello 2002) review of the four MOO techniques. Multiobjective optimisation (MOO) techniques force EA to produce solutions to constraint based problems by approaching a single objective optimisation as a multiobjective optimisation problem with $m + 1$ objectives where m is the number of constraints. There are many

multiobjective optimisation techniques that can then be applied to the new vector $\vec{v} = (\mathrm{f}(\vec{x}), \mathrm{f}_1(\vec{x}), \ldots, \mathrm{f}_m(\vec{x}))$ where $\mathrm{f}_1(\vec{x}), \ldots, \mathrm{f}_m(\vec{x})$ are the problem constraints. An ideal solution $\vec{x}$ may have $\mathrm{f}_i(\vec{x})=0$ for $1 \leq \mathrm{i} \leq \mathrm{m}$ and $\mathrm{f}(\vec{x}) \leq \mathrm{f}(\vec{y})$ for all feasible $\vec{y}$ assuming minimisation (Coello Coello 2002)

Many MOO techniques have been proposed. COMOGA (Surry & Radcliffe 1997) proposed by Surry *et al* ranked the population based on constraint violations. A portion of the population was selected based on this ranking and the rest of the individuals' fitness. This did not produce better results than the penalty function (Surry & Radcliffe 1997) and requires several extra parameters although it is not very sensitive to their values. This approach is also computationally expensive. Later in this thesis we compare the GeneRepair technique proposed in this thesis to the penalty technique.

Further Assorted Techniques

Parmee & Purchase (Parmee & Purchase 1994) handled constraints as objectives allowing the EA to locate a feasible region in the constrained search space. They used specialised operators to create a variable size hypercube around each feasible point to ensure the EA stays within the feasible region. This approach was proposed for heavily constrained search space and was appropriate for reaching the feasible region but this is not equivalent to reaching the global optimal. The use of special operators also restricts the generality of the approach.

Camponogara & Talukdar (Camponogara & Talukdar 1997) proposed to redefine a single objective optimisation problem to consider two objectives,

the first is to optimise the original objective function and the second is to minimise. Non-dominated solutions with respect to the two new objectives are generated and these define a search direction

$$d = (x_i - x_j)/|x_i - x_j| \tag{2.3}$$

where $x_i \epsilon S_i$, $x_j \epsilon S_j$ and S_i and S_j are Pareto sets. A line search is carried out in this direction (d) so that a solution can be found that is a better compromise than the previous two solutions found. Thus line search replaces the crossover operator in this EA. Half the population is also eliminated at regular intervals by replacing the less fit solutions with randomly generated points. This approach has a problem maintaining diversity and the use of line search space is computationally expensive.

Jiménez & Verdegay (Jimnéz & Verdegay 1999) proposed a min-max MOO technique which followed three simple rules (much like the technique proposed by Deb (Deb 1998) except that no extra method was used to maintain diversity. The rules of this technique for comparing individuals are:

1. If the they are both feasible select based on minimum value of objective function

2. If one is feasible choose it

3. If both are infeasible choose individual with lowest maximum violation wins based on $\max(g_j(\vec{x}))$, for $j = 1, \ldots,$ m where m is the total number of constraints.

This approach is not considered to be multiobjective optimisation by some (Coello Coello 2002) as it only ranks infeasible solutions based on their con-

straint violations. While this approach may sound like an alternative to finding feasible points in a heavily constrained search space it can also get stuck in an infeasible part of the search space, due to its random sampling of the feasible region and time may be wasted with such a blind search.

Another strong contributor to the field of MOO is Coello Coello (Coello Coello & Nacional 1999), (Coello Coello 2002). He proposed the use of population based MOO to handle each of the constraints as a separate objective. The population is split into *m+1* sub populations, where m is the number of constraints, so that part of the population is selected using the objective function as its fitness. If the problem is a minimisation problem then the fitness function is multiplied by -1. The other part of the population uses the first constraint and no penalties are enforced. . The algorithm used is:

if $g_j(\vec{x}) \leq 0.0$ **then** fitness = $g_j(\vec{x})$

else if v $\neq$ 0 **then** fitness = -v

else fitness = f($\vec{x}$)

where $g_j(\vec{x})$ refers to the constraint corresponding to sub-population j+1 and v refers to the number of constraints that are violated ($\leq$m), with the assumption that the first sub-population is assigned to the objective function f($\vec{x}$). This split of the population is carried out at every generation. Each sub-population will try to minimise its constraint violations. If a solution does not violate constraints but is infeasible the algorithm will try to minimise the total number of violations and join with other sub populations to drive the EA to a globally feasible region. The aim is to combine the distance from feasibility with information about the number of constraint violations. If the

solution is feasible it will be merged with the first sub-population where it will be evaluated with the same fitness function (the objective function). One of the main disadvantages of this approach is the number of sub-populations needed in complex problems.

Coello Coello went on to propose another technique based on non-dominance. With this new technique fitness is assigned using the following algorithm:

Let the vector $\vec{x}$ (i = 1,. . . , pop–size) be an individual in the current population with the size = pop–size. The rank of an individual $\vec{x}_i$ is computed using

$$rank(\vec{x}_i) = count(\vec{x}_i) + 1 \tag{2.4}$$

The count function is computed using the rules:

1. $\vec{x}_i$ is compared against every individual in the population where $\vec{x}_j$(j=1, . . . , pop–size and j$\neq$ i) is the other individual

2. Count is initialised to zero

3. If both $\vec{x}_j$ and $\vec{x}_i$ are feasible then both are ranked zero and count is unchanged

4. If $\vec{x}_i$ is infeasible but $\vec{x}_j$ is feasible then one is added to count

5. If both are infeasible but $\vec{x}_i$ violates more constraints then count($\vec{x}_i$) is incremented by one

6. If both are infeasible and violate the same number of constraints but $\vec{x}_i$ has a larger total amount of constraint violations, then count ($\vec{x}_i$) is incremented by one. If any constraints $g_k(\vec{x})$ (k=1,. . . m, where m

is total constraints) is satisfied if ($g_i(\vec{x}) \leq 0$), then the amount of constraint violations for $\vec{x}$ is given by

$$coef(\vec{x}_i) = \sum_{k=1}^{p} g_k(\vec{x}) forall g_k(\vec{x}) > 0 \tag{2.5}$$

Fitness is computed using the following rules

1. If $\vec{x}_i$ is feasible, rank($\vec{x}_i$) = $fitness(\vec{x}_i)$
2. else rank($\vec{x}_i$) = $1/rank(\vec{x}_i)$

Selection of individuals is based on rank($\vec{x}$) and values produced by $fitness(\vec{x}_i)$ must be normalised. This ensures that the rank of valid individuals is always higher than the rank of invalid ones. No special techniques are used to ensure diversity. This approach tends to generate good solutions in highly constrained search spaces but may have difficulties reaching a global optimal.

Ray *et al* (Ray, Kang & Chye 2000) went onto to propose a technique where individuals are ranked separately by their objective function and constraints. Mating rules are applied based on the information held by each individual about its own validity. This was inspired by Hinterding and Michalewicz (Hinterding & Michalewicz 1998) where the global optimal is reached through co-operative learning. This is a promising technique but as with all MOO techniques there are sacrifices made in terms of quality of solutions produced (Coello Coello 2002).

Runarsson & Yao (Runarsson & Yao 2000) proposed a technique where the population is ranked using a stochastic version of bubble sort. Individuals are compared to their adjacent neighbour over a certain number of sweeps.

This aims to find out whether the objective or penalty function is dominating so that a balance can be found and the EA can be guided to the global optimum. This is a MOO function as opposed to a simple penalty function as it addresses the imbalance and corrects it. One of the drawbacks of this technique is the need for a parameter to define the probability of using the objective function for comparison in the ranking process (when lying in the infeasible region).

2.4.3 Modified Representations and Operators

The second method for handling constraints is the use of modified representations and operators. The reason that evolutionary algorithms are unsuited to solving constraint based problems is that blind crossover and mutation operators will inadvertently introduce errors, generating infeasible solutions. As shown in Section 2.4.1 EA operators can cause invalid solutions to be generated.

The second way to enforce constraints in EA involves the use of special representations and operators (Coello Coello 2002). The use of a special representation (tailored for each problem type) in an EA means that the basic standard operators are no longer appropriate, hence the simultaneous introduction of modified operators. Special representations can be used when the normal representation is not appropriate for the problem at hand. The aim of these special representations is to simplify the search process and focus it (possibly exclusively) on the feasible search space. Special operators can be used in such a way that invalid solutions are never created.

An example of modified representations is "Random Key Encoding"proposed by James C. Bean (Bean 1992) for certain sequencing and optimisation problems such as job shop scheduling. This method eliminates the need for special crossover and mutation operators without adding computational overhead. It does this by encoding a solution with random numbers which are then used as keys to decode the solution. For example an n-job m-machine scheduling problem has each allele as a real number with an integer part belonging to the set $\{1,2,\ldots m\}$ but the decimal fraction is randomly generated with the interval (0,1). The integer part of the number is interpreted as the machine assignment for that job and the sorted fractional parts provide the job sequence on each machine (Normal & Bean 1995). This approach is quite problem specific and does not perform well for some other applications tested(Coello Coello 2002).

In a paper written by Yuval Davidor (Davidor 1989) a new technique is proposed which also uses Lamarckian probabilities for crossover and mutation. This introduces an example of a modified operator which is *analogous crossover* that chooses crossover points in the parent strings using phenotypic similarities. Crossover and mutation points were chosen based on the error distribution across the individual. This technique is also non-Mendelian.

Another example of modified representation is GENOCOP (GEnetic algorithm for Numerical Optimisation for COnstrained Problems). This technique was developed by Michalewicz and has since gone on to contribute to the creation of GENOCOPII and GENOCOPIII(Michalewicz & Nazhiyath 1995). GENOCOP gets rid of equality constraints and also eliminates a

number of problem variables. This simplifies the work of the EA as part of the search space has been removed. The EA then searches the newly formed convex set of remaining constraints (which are linear inequalities). An initial feasible solution is searched for and if it is not found the user is prompted to provide a starting point. This starting point will be duplicated to create the initial population. The operators then form linear combinations of individuals and this ensures that an invalid offspring will never be created. One problem with this technique is the requirement that the user of the EA must have a way of generating a starting point as GENOCOP assumes a feasible starting point. This technique is also limited to linear constraints (Dasgupta & Michalewicz 1997). GENOCOPIII incorporates the original GENOCOP(Michalewicz 1996) system, but also extends it by maintaining two separate populations, where a development in one population influences evaluations of individuals in the other population. This is different to the technique proposed in this thesis as it maintains two populations and it does not use non-Mendelian inheritance for repair.

Constraint Consistency was proposed by Kowalczyk (Kowalczyk 1997) to decrease the search space by aborting alleles that are not consistent with the problem constraints. This method ensures that individuals produce only valid solutions. A disadvantage of this approach is the computational cost of propagating constraints which may be more computationally expensive than the optimisation.

Another modification that can be made to EA is locating the boundary of the feasible region. As the name suggests this technique involves iden-

tifying the feasible region and searching the area close to this. This idea was originally proposed as a technique called *strategic oscillation* (Glover & Kochenberger 1995) and can be used in combinatorial and non-linear optimisation problems. It uses adaptive penalties or a similar mechanism to cross back and forth over the feasibility boundary by relaxing or tightening a factor that determines the direction of movement. This search of the boundary between feasible and infeasible regions is justified as in many non-linear optimisation problems at least some of the constraints are active at the global optimum. This approach uses an initialisation procedure to generate feasible points and genetic operators that explore the feasible region. The crossover operator is able to create all points between the parents and small mutation causes small change to the fitness function.

The drawback of this technique is that the genetic operators are highly confined and problem specific. The second disadvantage is that many problems have disjoint feasible regions, of which only one would be searched.

Another example of this family of constraint obeying EA is decoders. This is where the chromosome itself gives instructions on how to build a valid solution (Coello Coello 2002). The job of the decoder is to create a relationship is between a feasible solution and a decoded solution.

All of these approaches carry high computation costs. Another disadvantage of modified operators and representations is the extra work that must be done to maintain diversity. By preventing the creation of invalid solutions and limiting the search space of the EA, the diversity is weakened which can hinder the EA in its search to find a global optimal. This is problem specific

so even small changes to the problem specification can mean the modified operator/representation does not work.

2.4.4 Imposing Penalties

The third strategy that EA use to handle constraints is to penalise invalid solutions by imposing a penalty. This penalty may either prohibit or decrease the chance of the individual contributing to the next generation. There are seven principal ways to penalise these solutions which range in severity of and the way in which the penalty is imposed. The attraction of the penalty method is problem generality. The method does not need to be adapted for different problems. It is also biologically plausible as unfit individuals in nature suffer some kind of survival penalty.

Death Penalty

The first form of penalty we look at is the death penalty. As the name suggests this is the most extreme penalty which can be imposed on an individual. Using this technique invalid solutions are eliminated from the population. These solutions are identified as they break the problems hard constraints, representing infeasible or non-valid solutions. One problem with the death penalty technique is that it assumes that the population contains at least one valid individual as otherwise the EA will not produce a solution. Obviously the death penalty is an extreme punishment on invalid solutions, less extreme methods which impose penalties depending on the level of constraint violation are shown in the following subsections.

Static Penalty

While death penalties eliminate infeasible individuals from the population, static penalties gradually phase invalid solutions out of the population. A static penalty function reduces the fitness of solutions violating hard of soft constraints. These penalised individuals therefore have a reduced chance of contributing to subsequent generations. The penalty imposed increases as the violation reaches higher levels. When the violation is severe then so is the penalty but when the violation is minor the penalty is minimal. This means that the individuals with a major violation are heavily penalised and unlikely to contribute to further generations while individuals with few or smaller violations will have a high probability of still contributing to further generations but with a smaller probability than their valid counterparts.

The main disadvantage with this approach is the intricate formulae which are used to calculate the penalty applied to the fitness of each individual. These formulae require many parameters as seen in the technique proposed by Homaifar *et al*(Homaifar, Qi & Lai 1994) where a problem with m constraints needed $m(2l\ +1)$ parameters.

Dynamic Penalty

This technique involves any penalty function where the current generation number is used in some way to compute the corresponding penalty factors. In general the penalty function increases over time\generations. This means that violations in later generations are penalised in a stronger manner than violations in earlier generations. As the EA closes in on a solutions constraint

violations in this solution are less acceptable than in early generations when the EA is examining all of the possible solutions as it may be more difficult to find valid solutions earlier in the life cycle of the EA.

The problem with this technique is that it is difficult to produce good penalty factors and as with static penalties, if a bad penalty factor is chosen the EA could converge to non-optimal valid solutions or even worse, invalid solutions.

Annealing Penalty

The annealing penalty like dynamic penalty changes the penalty function - but in this case it happens when the algorithm has been trapped in a local optimal. At each generation only active constraints are considered and, similar to dynamic penalties, the penalty increases over time. This means that invalid solutions are highly penalised at later generations.

This approach does not fit with the biological analogy of EA with nature. As with the other penalty techniques the fine tuning of both the penalty function and the fitness function can prove difficult.

Adaptive Penalty

Another dynamic style penalty, this technique, proposed by Bean and Hadj-Alouane (Bean & Hadj-Alouane 1992) allows the penalty to loosen if the EA is producing feasible solutions and to tighten when a high rate of infeasible solutions are being created. The adaptive penalty technique takes feedback from the search process. A parameter used in the penalty calculation formula

is decreased if all individuals in the previous generation were valid. That parameter is increased if all of the individuals were invalid and it is unchanged if there were a mixture.

The problem lies in the choosing of the generational gap in which to feedback this information and as like the other techniques it is difficult to choose suitable values for the parameters involved in the penalty function.

Co-evolutionary Penalty

Coello Coello proposed a technique (Coello Coello 1999) where the penalty is split into two values so the EA has information about the number of violated constraints and also the corresponding amounts of these violations. As with other co-evolutionary techniques there are two different population used in this technique. The first population holds the individuals while the second population encodes the set of weight combinations that would be used to compute the fitness of the individuals in the first population. Therefore the second population contains the penalty factors. The problem with this technique, as with other techniques above, is the number of extra parameters introduced and the difficulty in initialising these parameters with suitable values. Ashish Mani and C. Patvardhan (Mani & Patvardhan 2009) suggest a co-evolutionary algorithm which uses a self determining and regulating penalty factor method as well as feasibility rules for handling constraints however this still requires the computation time and implementation for the upkeep of two populations.

Segregated Genetic Algorithm

This technique proposed by Le Riche *et al* (Riche, Knopf-lenoir & Haftka 1995) uses two penalty parameters for each constraint. The aim of this is to balance heavy and moderate penalties by maintaining two populations of individuals. When individuals merge they are segregated in terms of constraint satisfaction. The problem occurs when you try to choose penalties for each of the two sub-populations. While guidelines are available they are problem specific, which means that this technique does not lend itself to becoming problem independent.

Penalty Approach Summary

Penalty functions are the most common approach in the EA community to handle constraints and, as shown above, there are a wide variety of penalty functions to choose from. One of the disadvantages of the death penalty is that it limits diversity across the population. The main drawback of the static penalty function introduced is the high number of parameters required while when using dynamic penalties it can be difficult to derive a good dynamic penalty function. It is also difficult to produce good penalty factors for static penalties. The annealing penalty function shown is very sensitive to the values of its parameters making it difficult to implement while choosing the generation gap for the adaptive penalty function can also prove to be troublesome. Co-evolutionary penalties introduce additional parameters which adds to the complexity of the implementation as these need to be tuned. Similar to this it is difficult to choose the penalties for each of the

two sub-populations in the segregated genetic algorithm. For a recent survey of penalty functions for constrained optimisation see (Yeniay 2005).

2.4.5 Repair Strategies

Repair is the final technique to enable EA to handle constraint based problems. Relatively little work has been done on repair, so little in fact that in Neumann and Witt's Bioinspired Computation in Combinatorial Optimization (Neumann & Witt 2010) there was no mention of repair. In this section we shall examine the area in greater detail as genetic repair strategies are the focus of this thesis. While there are a variety of repair strategies, the common thread is the repair of invalid individuals to turn these into valid individuals. This means that individuals that disobey the problem constraints are repaired in order to satisfy the constraints. These repaired individuals can then be used in three ways. They can be used only for evaluation purposes (Liepins & Potter 1991), they can replace their invalid counterparts in the population (Nakano & Yamada 1991) or they can replace invalid individuals with some given probability(Orvosh & Davis 1993). Repairing invalid individuals and returning them to the population is sometimes known as Lamarckian evolution. This idea is based on the fact that individuals improve during their lifetime and these improvements are coded back into the chromosome. However, Lamarckism (Houck, Joines, Kay & Wilson 1997) specifically refers to traits that (already valid) individuals accrue during their lifetime. (The repair strategy presented in this thesis is not Lamarckian.) Alternative to the Lamarckian approach, the second heuris-

tic approach to repair is called Baldwinian. Using this approach the EA population does not change after the application of the repair heuristic and only the fitness function is modified (Salcedo-Sanz 2009). Hisao Ishibuchi and Narukawa (Ishibuchi & Narukawa 2005) use a partial Lamarckian repair technique to enable EA to solve the multi-objective knapsack problem. This greedy repair technique removes the heaviest items from the solution in order to repair it. This technique is problem specific to the knapsack problem.

Salcedo-Sanz (Salcedo-Sanz 2009) published a survey paper giving an overview of all of the important repair mechanisms enabling EA to handle constraints in 2009. The first half of the paper examined repair procedures for different representations while the second half of the paper described applications of different repair techniques. Salcedo Sanz divided the representations into four distinct sections. These sections were Repair Procedures in Permutation Encoding, Repair Procedures in Binary Representations, Repair Procedures in Graphs and Trees and Repair Procedures in Grouping GAs. In the repair procedures in binary representations Salcedo-Sanz describes repair heuristics for fixing the number of 1s in binary representations and shows examples of Hopfield Networks as repair heuristics. While describing repair procedures in graphs and trees several issues related to the use of specific representations of graphs and trees (specifically spanning trees) in evolutionary algorithms are discussed, and also how repair algorithms are sometimes needed to improve the EA performance in these problems. As the representation used for the experiments illustrated in this thesis is permutation encoding it is this section that is of great interest.

Repair Procedures in Permutation Encoding

Using a permutation encoding each solution is represented as an ordered list. This is suited to a wide range of problems such as TSP and RNA folding structure problem. The disadvantage of this encoding is that crossover produces invalid individuals so repair must be used if a modified crossover operator is not. Salcedo-Sanz describes two crossover operators which can be used with permutation encodings to provide valid individuals. These crossover operators are the partially mapped crossover operator (PMX) (Goldberg & Lingle 1985) and the tie-breaking crossover (TBX).

To conduct PMX two crossing points are uniformly chosen, at random in the parents. The corresponding genes are swapped between the two crossover points. In each offspring the repeated genes outside of the genes that were exchanged are located, and these are substituted by the corresponding genes within the crossover points in the other individual. The rest of the genes in each individual are maintained as they were before the crossover process. With TBX a standard two-point crossover is performed. A new permutation called crossover map is randomly generated. Each offspring is multiplied by n (the length of the individual) and the corresponding gene of the crossover map is added. The lowest number in each offspring is replaced by 1, the second lowest by 2, etc.

While permutation encoding is used in this thesis no modified crossover operator is used. Operators are therefore independent of representation and problem domain leading to a constraint enforcing technique which will not need to be adapted in a major way for different representations or problem

sets. Instead a GeneRepair technique utilising non-Mendelian inheritance is presented and later we will examine whether this can be used to enable EO to solve constraint based problems.

Salcedo-Sanz went on to describe the applications of repair techniques using six subsections. Some applications of hybrid EA with Repair Techniques were examined as well as Permutations and Binary Fix Encodings (which included the TSP, RNA folding problem and Broadcast Scheduling Problem in ad-hoc networks), Hybrid Hopfield Network Evolution Algorithms, Applications involving Repair Heuristics in Graphs and trees, Applications of Hybrid Grouping Genetic Algorithms and Other Applications. Salcedo-Sanz paper has reviewed (EAs) which use repair techniques for reducing the search space size. These approaches have been called hybrid algorithms, to differentiate them from memetic algorithms, which use local search procedures to improve the objective function value of each individual in the EA. Interestingly, he recommends that this paper could be used as a starting point when investigating new repair techniques.

Davis and Orvosh (Orvosh & Davis 1993) proposed the technique of replacing only 5% of invalid individuals with their repaired counterparts. This highly cited paper uses the rather complex Survivable Network Design Problem and the standard Graph Colouring Problem to compare the results produced by different rates of replacement of original chromosomes with their repaired counterparts. The results produced are the findings when two different problem sets were tested using steady state GAs and are based on 6 CPU months of running data on a Symbolics 3630 Lisp machine. Orvosh

and Davis suggest that replacing at a 5% probability yields better results than either always replacing or never replacing. Orvosh and Davis do not detail the actual repair mechanism. We conjecture that this might involve an heuristic repair process, reducing population diversity and necessitating a low replacement rate.

Liepens *et al* (Liepins & Potter 1991) have used a different approach and never return repaired individuals to the population. Instead, these repaired individuals are used for evaluation and guidance of the EA. This approach could be seen as somewhat co-evolutionary as the repaired individuals are not part of the current population but instead guide the current population towards a feasible search space. The problem used by Liepens *et al* was the complex network design problem. While Orvosh and Davis suggest that this was the correct method for the network design problem and agreed with the findings of Liepens *et al* Orvosh and Davis still suggest that a 5% replacement strategy should be used.

Michalevicz *et al* (Michalewicz 1996) however went on to propose that the probability of repair should be 15% for best results in numerical optimisation problems. The original Genocop (also Michalevicz) (for GEnetic algorithm for Numerical Optimization of COnstrained Problems) system assumes linear constraints only and a feasible starting point (or feasible initial population). A closed set of operators maintains feasibility of solutions. GENOCOPIII (Michalewicz & Nazhiyath 1995) uses repair by having two populations and allowing the evaluation of individuals in one population to be influenced by the results in the other. The first population has search points which

obey the problems constraints while the second population consists of feasible reference points. The reference points are evaluated by the objective function but the search points are repaired for evaluation. It is shown that GenocopIII is a promising tool for constrained nonlinear optimization problems. However, there are many issues which require further attention and experiments. This paper addresses the 5% repair replacement suggestion of Orvosh and Davis and suggests that it would be interesting to check this rule in numerical domains. Handa *et al* (Handa, Watanabe, Katai, Konishi & Baba 1999) also use a co-evolutionary technique whereby a population called "H-GA"stores solutions and another population called "P-GA"stores schemata. This complex co-evolutionary technique is quite complicated and differs to the technique presented in this thesis as it uses two populations and does not repair using non-Mendelian ancestral information. The repair strategy presented in this thesis uses a repair rate of 100% as it is meta-heuristic and allows sufficient diversity in the repair process.

Unfortunately repair algorithms tend to be problem specific and some problems do not lend themselves towards easy repair of individuals which can lead to high computational costs. The repair operator itself can also introduce a bias which can skew the evolutionary process and greatly limit population diversity. As stated earlier repair strategies have not been researched as much as other constraint enforcing strategies for EA as up to now the main disadvantage attributed to genetic repair is its problem dependence(Michalewicz & Fogel 2000). This is because most genetic repair operators are based upon heuristics (Ahn & Ramakrishna 2002) (Arroyo 2002) (Bäck, Schütz & Khuri

1995), (Nakano & Yamada 1991) (Orvosh & Davis 1993), (Walters 1998) that are highly problem dependent.

Ahn and Ramakrishna (Ahn & Ramakrishna 2002) propose a repair mechanism for the Shortest Path Routing Problem which operates by searching for nested loops within the solution and deleting the nested loop. Variable-length chromosomes (strings) and their genes (parameters) were used to encode the problem. The crossover operation exchanges partial chromosomes (partial routes) at positionally independent crossing sites and the mutation operation maintains the genetic diversity of the population. The proposed algorithm can cure all the infeasible chromosomes with a simple repair function. Computer simulations show that the proposed algorithm exhibits a much better quality of solution (route optimality) and a much higher rate of convergence than other algorithms. Among the authors of this paper it is felt that the presented algorithm can be used for determining an adequate population size (for a desired quality of solution) in that routing problem. The drawback of this mechanism is that it is problem specific. It is also different to the repair strategy presented in this thesis as it repairs the individual by removing nested loops while the repair strategy presented in this thesis uses an archived ancestor as a template for repair.

Arroyo (Arroyo 2002) presents a heuristic repair mechanism to solve the Unit Commitment Problem. This optimisation problem is large-scale, combinatorial, mixed-integer, and nonlinear. At the time of publication of this paper (2002) exact solution techniques to solve it were not available. Aurora presented a novel genetic algorithm using repair, which employed heuristics

in order to achieve a near optimal solution to this problem. This optimisation technique is directly parallelizable, with three different parallel approaches being developed. The paper showed that the developed genetic algorithm has been successfully applied to realistic case studies. While this mechanism avoids high computation cost by exploiting parallel computing it is problem specific. This means that in order to apply this approach to a different problem domain a large amount of modification would be required.

Back *et al* (Bäck *et al.* 1995) uses the non-unicost set-covering problem to illustrate their repair technique. They compared the effectiveness of using various stochastic operators: four different crossover operators are compared to using a repair heuristic. Their repair heuristic transforms infeasible strings into feasible ones. The stochastic operators are incorporated into GENEsYs (Bäck 1992). This uses a simple fitness function that has a graded penalty term to penalise invalid solutions. GENEsYs does not have any prior knowledge of the problem except for the fitness function which means that this strategy is less problem specific than other strategies we have examine so far. When their greedy heuristic repair was compared to the death penalty, the death penalty was far outperformed. They even went on to say that while the solutions produced by the death penalty strategy were all feasible they were nowhere near the quality of those produced by the greedy repair heuristic. They also suggested that incorporating a simple repair method into the evaluation of solutions improved the solution quality and caused the GA to encounter better solutions than the greedy heuristic in most cases. The paper, however, could not give a definitive answer on the rate of repair

to use and suggested that further research is required to determine which portion of infeasible solutions should be repaired.

Nakano and Yamada (Nakano & Yamada 1991) introduce a heuristic repair mechanism to solve the job shop scheduling problem. The first serious application of GAs to solve the JSSP was proposed by Nakano and Yamada using a bit string representation and conventional genetic operators. Although this approach is simple and straightforward, it was not very powerful. Early approaches by the authors were active schedule-based GAs which are suitable for middle-size problems. It became apparent, however, that it was necessary to combine each with other heuristics such as the shifting bottleneck or local search to solve larger-size problems. The multi-step crossover fusion (MSXF) was proposed by Yamada and Nakano as a unified operator of a local search method and a recombination operator in genetic local search. This paper illustrated that the MSXF-GA outperforms other GA methods in terms of the MT benchmark and is able to find near-optimal solutions for the ten benchmark problems, including optimal solutions for five of them. While this method produces excellent results (sometimes optimal) it is unfortunately tailored to this specific problem.

Walters (Walters 1998) presents a repair technique for the TSP which incorporates *Soft Brood Selection*. In brood selection, a pair of parents will generate several children (a brood) and only the best one or two children will be selected as offspring for the rest of the genetic algorithm. ("Soft"selection refers to the fact that children are selected based on their fitness values relative to each other). Walters has called this algorithm Directed Edge Repair

(DER) and it uses Nearest Neighbour Notation. The repair approach taken in this paper assigns as many edges as possible from the original chromosome. If an edge from a particular city cannot be assigned to the tour, a new directed edge from the same city is found, with a distance as close as possible to the one specified in the original gene value. By attempting to assign edges that are close in distance to the original values specified in the chromosome, it is hoped that the final tour is close to the one originally described in the genes. While this is a successful algorithm for solving the TSP it is problem specific and further work is required to calculate the optimal brood size, while identifying the required nearest neighbour can also be extremely expensive for large problems.

Kuk-Hyun Han and Jong-Hwan Kim (hyun Han & Kim 2000) present a novel evolutionary computing method called a genetic quantum algorithm (GQA). This is based on the concept of principles of quantum computing. The effectiveness and applicability of GQA are illustrated using the knapsack problem as the problem domain. The authors suggest that this GQA method is superior to GAs using penalty functions, repair methods and decoders. GQA can represent a linear superposition of states and there is no need to include many individuals. While this GQA technique does outperform penalty functions and some repair methods, it is moving away from the analogy between natural evolution and EA which this thesis will extend.

Weimer *et al*(Weimer, Forrest, Le Goues & Nguyen 2010) combine evolutionary computation with program analysis methods to automatically repair bugs in off-the-shelf legacy C programs. The input to this technique is the

buggy C code, a failed test case and a small number of other test cases that encode the required functionality of the problem. Looking at this technique from an abstract view, the input of test cases that encode the required functionality of the problem could be seen as the template. This repair technique does not rely on formal specifications. This paper is more focussed on completing research which will eventually lead to the dream of automatic programming, that is evolving new programs from scratch than the improvement of EA to solve constraint based problems, but it does give us an insight into repair techniques used to evolve legacy software.

Kimbrough and Wood (Kimbrough & Wood 2007) present a repair-by-interpolation mechanism whereby a genetic operator takes as input two individuals with differing feasibility and outputs a pair (feasible or infeasible) that differ by only a single bit. The authors view this as a valuable technique as any optimal solution is within a single bit of transitioning between infeasibility and feasibility unless the constraints are irrelevant. This technique falls into the binary representation category described by Salcedo-Sanz (Salcedo-Sanz 2009) and is implemented by a binary search along a path connecting the two inputs. This path is seen as a portion of randomly selected gray code. The authors report that their exploratory research indicates that this operator can find additional good and sometimes better solutions from a randomly generated initial population. If the GA is run to maturity however the technique is not particularly productive for the illustrated examples but is said to be more useful than the GA on difficult problems. This technique is representation specific and, as with other techniques described, does not

build on the existing analogy between natural evolution and EA. However, the repair technique presented in this thesis aims to extend the existing analogy and build upon the advances made in biology to enable the EA to solve constraint based problems.

Craenen *et al* (Craenen, Eiben & Marchiori 2001) provide a broad description of the field of EA that address constraint based problems. Much like Salcedo-Sanz (2009) this paper is an excellent starting point when trying to research the field of EA and constraint based problems.

Eiben (Eiben 2001) also gives a description of definitions, research directions and methodology which is a valuable resource when conducting research in this field. Eiben *et al* (Eiben 2001) describe EA for constraint satisfaction problems as falling into two categories. The first category is heuristics that can be incorporated in almost any EA component and the second category is formed by adaptive features modified during a run. Eiben *et al* describe repair as falling into the first category as repair techniques (as shown in throughout this section) tend to rely on heuristics. The biologically inspired repair strategy proposed in this thesis is different to the repair strategies mentioned as it is a meta-heuristic which means that it is concerned with the representation space as opposed to the problem space. Eiben *et al* go on to give three guidelines for future research into constraint handling mechanisms for EA. These are as follows:

1. Use, possibly existing, heuristics to estimate the quality of sub-individual entities in the components of the EA: fitness function, mutation and recombination operators, selection and repair mechanism.

2. Exploit the composite nature of the fitness function and change its composition over time.

3. Try small populations and mutation only schemes.

The repair mechanism presented in this thesis used existing quality estimate for the fitness function which in this case was the tour length as the TSP problem domain was used as a testbed. While the composite nature of the fitness function was not exploited and mutation only schemes would not be suitable as the thesis presents a new repair technique, small populations were examined.

Previous Work on GeneRepair

This thesis builds on previous work on GeneRepair. This work is described in (Mitchell, O'Donoghue & Trenaman 2000), (Mitchell, O'Donoghue, Barnes & McCarville 2003), (Mitchell 2005*a*), (Mitchell 2005*b*) and (Mitchell 2007). In 2007 Dr. George Mitchell presented his PhD thesis (Mitchell 2007) entitled Evolutionary Computation Applied to Optimisation Problems. In this thesis Mitchell introduced the Genetic Repair technique which repaired invalid individuals in a population and incorporated GeneRepair results with his work on Quality Time Tradeoff (QTT). In this thesis GeneRepair was described as being a form of template driven repair. Template driven repair uses a template to support the correction of an invalid individual in the EO. In Figure 2.4 we can see that the *Invalid Individual* disobeys the TSP constraints by having a duplicate City 2 and by omitting City 3.

GeneRepair repairs invalid individuals by using the corrective template to repair any invalidity identified in the invalid individual. The *Repair Template* is shown on the right of the Figure 2.4. This is simply a sequence of all genes required to solve this permutation problem. The template is valid as it does not have any duplicate or missing cities. GeneRepair consists of two distinct phases. The first phase of GeneRepair is called *error detection*. This phase identifies genetic defects such as duplicate cities in the solution. The second phase of GeneRepair is called *Error Correction*. This replaces the duplicate City 2 which is highlighted with a red box with the missing City 3 which is highlighted with an orange box. The repaired individual is produced by using the repair template to repair any invalidity found to exist in the invalid individual. The *Repaired Individual* is shown at the bottom of Figure 2.4 with the newly inserted City 3 highlighted using a green box. This individual is valid and can therefore rejoin the population and may be used to produce further generations. The fact that this individual has been repaired does not guarantee that it will contribute to the next generation. It simply gives the individual an equal chance to that of every other valid individual of the population, depending upon the selection operator being used.

Mitchell describes the GeneRepair technique as being a follow up to the original *CleanUp* operator (Mitchell *et al.* 2000). This was a repair technique originally designed to improve a genetic algorithm used to find solutions for the TSP only. It was designed to replace the penalty function and improve the computational efficiency of the GA. Following on from the original implementation of the *CleanUp* repair operator, further work was initiated to

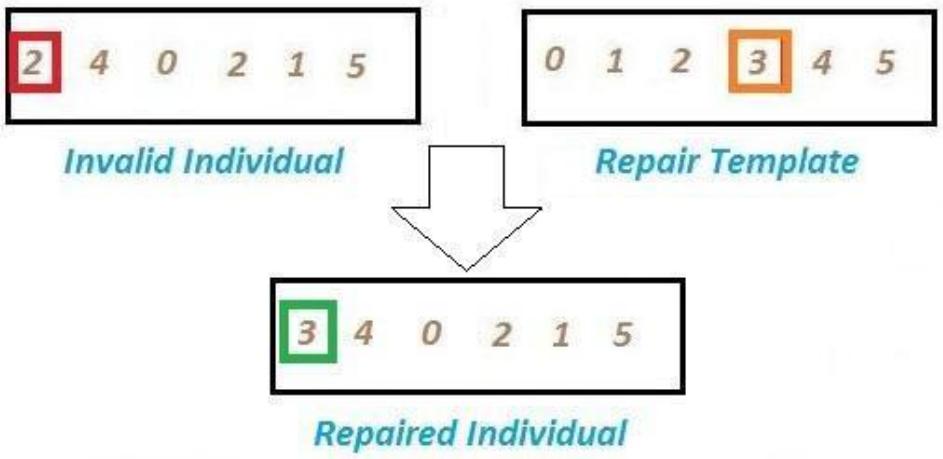

Figure 2.4: Basic Template Repair

implement a GA which relied solely on the repair operator to ensure solution validity (Mitchell 2007). This early repair operator was not related to a natural process of repair but instead was a corrective function which transformed invalid tours in the population to valid tours. In the investigation of the template repair method Mitchell was focused on the results produced by the repair templates rather than investigating the ease of use of the repair templates themselves. Mitchell compared the results produced by seven different techniques for generating repair templates. These were:

- Static Random Template: A random template is generated at the start of the GA search and remains fixed thereafter
- Dynamic Random Template: A random template is generated for each

invalid individual, thus each infeasible individual is repaired using a different template

- Best Solution in Previous Generation
- Best Solution so Far: This is the best tour found in the GA search
- Fitter Parent
- Least Fit Parent
- Varying Combination on Each Generation: Randomly use any of the foregoing template methodologies

As well as exploring two Mendelian forms of repair, Mitchell *et al* explored several strategies that are not biologically inspired. Overall his results indicate that the "dynamic random template" produced the best result, echoing the results of Lichtblau (2002). However, even better results were produced by a random selection of all his techniques.

Mitchell *et al* (2003) extended the previous research (Mitchell *et al.* 2000) on the repair operator and changed the name to GeneRepair. This method could be seen to be a genetic repair approach as two of the seven templates investigated use the parent as a repair template. This is directly inherited information relating the invalid individual and so could be seen to be analogous with ancestral genetic information.

Mitchell describes three techniques for applying the templates (or three scanning methods) which are Left-to-Right, Right-to-Left and Random Direction. In this thesis I will also compare these three techniques for applying

the repair templates. The pseudo code for the implementation of GeneRepair is shown in Algorithm 2.1

Mitchell *et al* (Mitchell *et al.* 2003) the GeneRepair technique was presented and the problem sets used were the TSP and the VRP. The findings suggested that a random template should be used in conjunction with GeneRepair and also went on to investigate why GeneRepair is called upon to a greater extent in early evolution as opposed to later in the evolutionary cycle. The authors also point out that GeneRepair is not an alternative to standard mutation. While Mitchell did carry out experiments and publish findings (Mitchell 2005*b*) on this GeneRepair technique much of his research focused on the cost benefit trade-off function. In (Mitchell 2005*a*) presented a Distributed Parameter-Less GA which was used in conjunction with GeneRepair and allowed its users also to achieve economically viable results in a shorter space of time as a direct result of the GATermination operator. This GATermination operator is a *quality time tradeoff* (QTT) operator and was successfully been tested on the TSPLIB benchmark problem set where significant computation time could be saved. QTT simply stops when it reasons there is little advantage in waiting for a better solution by signalling to the EA when to quit. It does not produce superior solutions and is simply a time-saving technique.

Mitchell concludes that a GA with Generepair consistently outperforms all other GA approaches tested. He also concludes that a cost/benefit function (QTT Tradeoff) was developed which allowed the user to set his own criteria in terms of quality and cost for the TSP GA configuration setting

```
1  repeat
2      Select candidate from population P(i) based on order in
       population (i, j = 0);
3      repeat
4          Generate random tour of cities to form the template;
5          repeat
6              Compare candidate city[j] with template, if city[j] in
               template and is un-flagged then flag city[j] in candidate
               solution and template;
7          until until all cities have been checked;
8          If all cities in template and candidate flagged tour valid;
9          Else if unflagged cities in template;
10         repeat
11             Replace first unflagged city in candidate solution with first
               unflagged city in template ;
12             Proceed in left to right manner replacing unflagged cities
               with unflagged counterparts from template;
13         until until all cities in candidate solution are updated;
14         Else if unflagged cities in candidate and template all flagged
           repeat
15             Delete first unflagged city in candidate solution;
16             Proceed in a left to right manner deleting unflagged cities
17         until until all only flagged cities remain;
18     until until candidate solution validated;
19 until until all candidate solutions validated;
```


Algorithm 2.1: GeneRepair Pseudo code

problem. His third finding was that optimal configuration settings (genetic operators and parameters) for three TSP problem sizes were established from a possible 250 million configurations. Mitchell proposed that the best single template of those GeneRepair techniques investigated was the use of a dynamic random template. He showed that this technique produced superior results to those produced when the parent was used as a repair template. While the use of the fittest and least fit parent were compared as repair template within GeneRepair the use of a random parent was not investigated. While Mitchell did mention the exciting biological finding of non-Mendelian inheritance proposed by Lolle *et al* (Lolle *et al.* 2005) his thesis did ***not*** explore the use of a non-Mendelian ancestral repair template with GeneRepair. Perhaps this is due to the fact that this biological discovery was made close to the end of his PhD research. Curiously, Mitchell found that the best results were produced by the final strategy - a random selection of the listed strategies. This is most interesting as the "dynamic random template"did not produce the best result - thus "true"randomness must be excessive and this reduced version of randomness outperformed true randomness. Therefore Mitchell suggested that RTR is not best but something less than completely random is better and he found this to be a random choice of the templates compared. This finding provides some of the motivation for this thesis.

As far as the author is aware the non-Mendelian repair template mechanism presented in this thesis has not been investigated before, and non-Mendelian inheritance has not been used for repair. During ongoing communication with Dr. Lolle (University of Waterloo) it has become clear that

such research in the computer science field is of great interest to those involved in the research of non-Mendelian inheritance in the academic field of biology.

2.5 Review

In this chapter I have explained how EA are used and how they are implemented. We saw that the standard EA is ill-suited to solving constraint based problems. For this reason there has been a large amount of research into adaptations of the canonical EA which enable it solve constraint based problems. This Chapter reviewed many of these techniques. While these techniques are hugely advantageous by enabling the EA to solve constraint based problems, they are also often problem specific and sometimes difficult to implement. Many of the adaptations are not based on biological findings and so do not belong to the analogy which is often used to describe EA. This sometimes causes these techniques to be difficult to understand and recreate. One common trait shared by the above reviewed papers is that they dispose of a population when producing the next generation - in contrast we archive some solutions for possible later inclusion in the population. Thus, a small number of reasonably fit solutions may have a second chance of contributing at least some portion of their archived genetic material to (much) later generations.

Chapter 3

The Analogy Between Evolutionary Algorithms and Biological Evolution

3.1 Introduction

Evolution is the method by which a species is updated in order to exist within its environment. Our understanding of evolution is based on the Darwinian idea of Survival of the Fittest (Darwin 1872) (6th Edition). An individual most suited to the environment will have a greater chance of survival and so will have a higher probability of contributing towards the next generation. This is a seemingly simple and successful algorithm. In this chapter I will show the existing analogy between natural evolution and EA and how it underpins our understanding of EA. I will then show my novel extension to

this analogy which updates the analogy to reflect recent biological research findings.

Analogies can be the foundation for inventions, a teaching tool and even a problem solving algorithm. By making an inference based on analogy we can expand our knowledge of a subject in a reliable way. The design of the camera can be seen to resemble the design of the human eye. When people are being taught about computers the CPU is often referred to as the brain of the computer. In computer science analogies based on nature can be used to solve problems. In this thesis I examine the analogy between nature and Evolutionary Algorithms (EA).

By bridging the gap between EA and biology this chapter further examines the links between both disciplines and shows where each of the experimental parameters originate. This is of interest because knowing the parameters biological counterpart may allow us to fine-tune these parameters which can cause the EA to produce better results. The motivation for examining the analogy is so that our EA can be extended in the correct manner. To do this the current analogy between biology and EA needs to be explored. This exploration can then extend our biological understanding of Genetic Repair so that this information can be transferred to the EA in accordance with this extended analogy. This correctly extended analogy and computational model (EA + GeneRepair) will later be reviewed which will generate some tentative inferences about the likely viability of Lolle's repair strategy in the biological setting. This extended understanding of EA will be evaluated from both perspectives - the computations as well as the

biologically inspired perspective.

3.2 Analogy - The Power of Comparing like with Like

Analogy is a cognitive process of transferring information or meaning from a particular subject (called the *source*) to another subject (called the *target*) (Gentner 1983). In this thesis I use the analogy between nature and computer science, specifically the analogy that can be drawn between the repair mechanism of the *Arabidopsis thaliana* plant and the corresponding repair mechanism that we add to the Evolutionary Algorithms.

In order to understand this analogy we must first have a brief understanding of what forms an analogical comparison between the *source* and the *target*. For the sake of an illustrative example the human eye will be the source and the basic camera will be the target. On the left of Figure 3.1 we see a basic diagram of the structure of the human eye. The eye is made up of an opening or iris where light, that has bounced off the image, which the eye is focusing on, enters the eye. This light is then passed through a lens and reflected onto the light sensitive area at the back of the eye. The eye lid serves as a protective layer over this sensitive lens. This is a basic explanation of how the eye functions omitting several details such as focus, aperture and the more biological details such as the function of the optical nerve. If we now focus our attention on the right of Figure 3.1 we can see a basic diagram for the standard camera. The camera has a shutter which

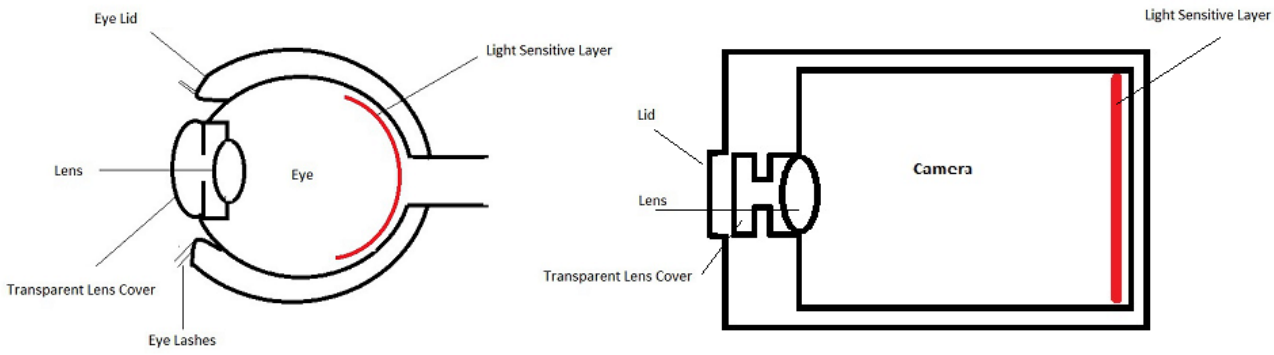

Figure 3.1: The Structure of Camera and The Structure of the Human Eye

opens and closes. It also has a transparent protection layer which surrounds the delicate lens. Light bounces off the subject and passes through the open shutter onto the lens. The lens then refracts the image on the light-sensitive film at the back of the camera. Again, this model is basic and does not include specific details about focus, aperture or image storage. We shall now explore the details of the analogical comparison between the human eye and the camera.The purpose of this section is to understand the analogy itself rather that the structure of the camera *per se*. In Figure 3.1 the source (the eye) and the target (the camera) of the analogy are pictured alongside each other. In this figure we can see that various items in the source form a 1-to-1 correspondence to items in the target.

The art of analogy links the target to the source. This means that we link the eye to the camera. To identify the individual comparisons that form the overall analogy See Table 3.1

Table 3.1: Analogy Between The Structure of the Eye and the Structure of the Camera

The Eye (the source)	**The Camera (the target)**
Eyelid	Shutter
Iris	Lens Cover
Lens	Lens
Light Sensitive Layer	Light Sensitive Film

Figure 3.2 shows the analogy between the eye and the camera. We can see from this that the analogy has equated (or aligned) the pairs of objects, with one object from the source aligned to one from the target.

The blue text in Figure 3.2 represents the parts of the eye in the analogy while the black text represents the corresponding parts of the camera. The purple text represents the relationship between the parts of the eye/camera while the arrows show the direction of the relationship. For example the iris (source) and the transparent lens cover (target) *protect* the lens of the eye (source) and camera (target) respectively. Figure 3.2 therefore shows the analogy between the eye and the camera as it might currently stand.

People often talk about wine tasting in terms of other concepts, while deeper analogies underlie our conceptualisation of time. One interpretation sees us as stationary and time moving past while another sees us actively moving through time. Analogies enable us to gain as good an understanding of a target subject as we have of the source. Analogies allow us to use our

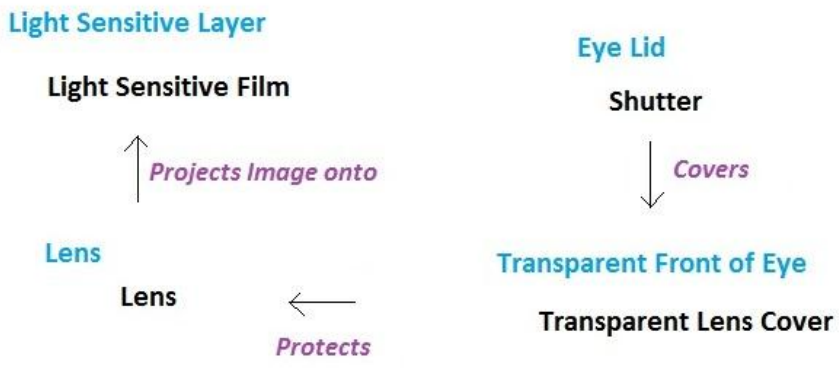

Figure 3.2: The Camera and The Eye Analogy

knowledge of one subject to quickly learn about a new subject. This is the reason that analogies are often used as a teaching aid (Aubusson, Harrison & Ritchie 1996). Analogies even allow us to reason about complex abstract ideas in terms of more physical ones. In this thesis we attempt to extend the analogy between EA and biology by using the complex information we have about the biological side of the analogy to improve our understanding and extend the EA side of the analogy. There are many more uses of analogies but one major advantage is that analogies suggest inferences about the target and often these can be useful, novel and even creative inferences (O'Donoghue 2007).

3.3 Extending an Analogy

In Figure 3.2 I showed a mapping from the source to the target. In that analogy we note that there were no isolated unmapped elements in the source (the eye). Analogical inference is the practise of extending the mapping between the source and the target by transferring unmapped source information to the target thus creating new inferences about the target. To conduct analogical inference the "Copy with Substitution and Generation"(Holyoak, Novick & Melz 1994) technique is used:

1. Identify extra information in the source which cannot be mapped to the current information in the target

2. Copy the verbs from the source to create extra information in the target and map the identified information in the source to this newly created information in the target

3. Copy the nouns (connected to these verbs) to the target - adapting them so that they are appropriate to the target

This method of analogical inference can be used to extend the analogy. In our example, we know that glasses can be used to improve the ability of the lens in the human eye. We can use this information as an analogical inference to transfer this information from the source (the eye) to the target (the camera). Glasses (an extra lens) can be used to **improve the ability** of the **lens** in the **eye**. When we transfer this to the target we can say that an extra lens can be used to **improve the ability** of the **lens** in the **camera**. We can see that the piece of apparatus used in the target would be a zoom

lens and can therefore extend the analogy using analogical inference, to state that zoom lens corresponds to glasses. In Figure 3.3 we can see the extra piece of information in the source, which is the glasses, and can then see the inferred extra piece of information in the target, which is the newly added zoom lens. Analogical inference has allowed us to extend our knowledge of the target using extra knowledge of the source.

This extended analogy is illustrated in Figure 3.4. This diagram shows the mapping from the source (the eye) to the target (the camera) including the extra information which has been added.

This example shows the power of analogies. We began with a source and a target, mapping the information from the source to the target. We went on to identify extra information in the source. Using analogical inference this information was transferred to the target. Creating this analogy has allowed us to extend our knowledge of the target by using additional knowledge of the source.

3.4 Analogy Between EA and Biological Evolution

In this thesis I will extend the analogy between nature (source) and evolutionary optimisation (target). A mapping between these two subjects already exists as natural evolution has often been used to describe evolutionary optimisation, however new information has been identified in the source that is not included in this analogy. Specifically we focus on the new information re-

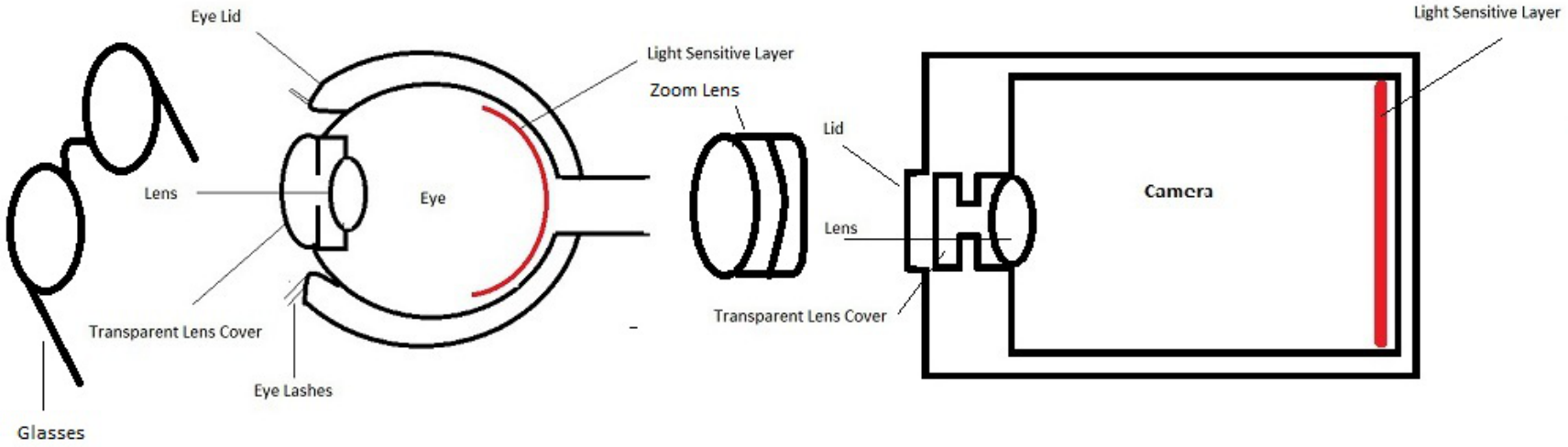

Figure 3.3: The Camera and The Eye with zoom apparatus

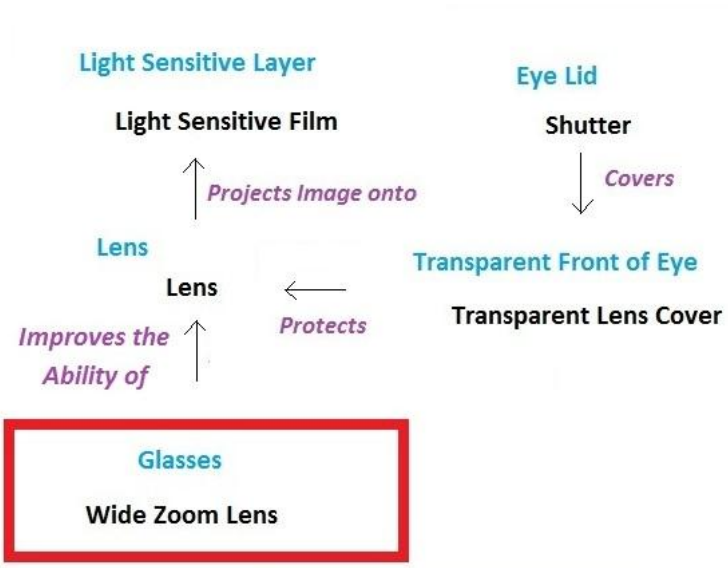

Figure 3.4: The Camera and The Eye Analogy with Analogical Inference

lating to the repair mechanism found in the *Arabidopsis thaliana* plant. I will use the analogical inference approach to extend the analogy by incorporating this repair mechanism found in the source into the target. By incorporating this repair mechanism prior knowledge of the source is being used to extend knowledge of the target in the same way as shown in the previous example.

3.5 The Existing Analogy Between EA and Natural Evolution

Having seen the process of analogical inference applied to the eye, we shall now see how this analogical techniques can be applied to the analogy between natural and simulated evolution. This thesis investigates the new repair

mechanism in EO resulting from the process of analogical inference. In the same way as with the analogy between the eye and the camera we will now examine the original analogy between the biological evolutionary process (the source) and the EA (the target) - before the inclusion of the repair mechanism. The first step in biological evolution can be seen to be the first living organisms (or individuals) of the evolved species. In the EA the *Generate* step corresponds to this as it is the first set of individuals (solutions) created. In biological evolution some of these individuals will go on to contribute to the next generation while others will not and in the EA (target) this is the *Select* step. In the source the individuals mate or outcross while in the target this is called *Crossover*. In biological evolution a tiny portion of these individuals are mutated and this corresponds to the *Mutate* step in the EA. This cycle is repeated in both the source and the target. We can now see how each of the steps described in Section 2.2.2 correspond to biological evolution (the source). In Table 3.2 we can see each of the items in the source and the corresponding item in the target.

While this analogy is broad and the resulting algorithm can be used in many different situations (Fogel 1994) there are inherent limitations with this existing analogy and so the existing structure of EA.

3.5.1 Drawbacks of Existing Analogy

The existing analogy which defines the structure of the EA does not exploit recent advances in biological research. The original analogy can be viewed as being based on evolution and how beings evolve in a natural environment.

Table 3.2: Analogy between EA & Biological Evolution

Biological Evolution	**Evolutionary Algorithm**
Gene	City
Genotype	Candidate solution (List of Cities)
Phenotype	Solution
Variable Size population	Fixed size Population
Fitness	Optimality
Selection	Selection
Outcrossing	Crossover
Mutation	Mutation
Homozygous Plant	Single instance of each gene in solution
DNA	Representation
Initial Population	Generate Step

Adaptations to the analogy (Coello Coello 2002) (See Chapter 2) have been broadly based on improving the results produced and enabling EA to handle constraints rather than starting by strengthening the analogy with nature. Computer scientists have understandably focused on improving the algorithm whereby the improvement is measured in its ability to solve problems.

EA were first introduced as a problem solving strategy mirroring how nature solves problems. The reason for this is that natural evolution is a most robust yet efficient problem-solving technique (Fogel 1995). The concentrated efforts to improve EA focused on one side of the analogy rather than improving EA by building on the analogy using information known about the source. This tangent unfortunately has led to many EA becoming problem specific. EA can be difficult to adapt to solve a new constraint problem - even if a similar problem has been previously solved.

My approach is to update the analogy to reflect biological advances since the first EA development and then to use this updated analogy to produce an EA which is not problem specific and is more suited to constraint based problems than the current set of algorithms available. For a recent survey on constraint handling in EA see (Salcedo-Sanz 2009) and Chapter 2.

3.6 Ancestral Repair in *Arabidopsis thaliana*

We will now look at the biological advances which, in the following section, will be used to extend the analogy between nature and EO. Since the introduction of the field of evolutionary optimisation there have been many variations on the original idea (Coello Coello 2002) (Michalewicz 1995). While

the variations may enable the algorithm to solve different problem types or to go about solving problems in a slightly different manner the fundamentals of the algorithm (Fogel 1994) remain the same.

Since these steps were developed research has also continued on the other side of the analogy. Biologists have made ground breaking advances from sequencing the human genome (1000 Human Genome Project) to cloning a sheep (Campbell, McWhir, Ritchie & Wilmut 1996). In 2005 one such ground breaking advance was described in a peer reviewed research paper published in Nature (Lolle *et al.* 2005). This paper described the investigation into non-Mendelian repair in the *Arabidopsis thaliana* (*A. thaliana*) plant. It is called non-Mendelian as it disagrees with the findings laid out by Gregor Mendel (also known as the father of modern genetics) in 1865 (Mendel 1865). This paper outlined Mendel's experiments (which were carried out over eight years) tracking seven different characteristics in over 33,500 pea plants. Mendel showed that some parent plants could produce progeny that did not resemble either parent. Mendel predicted that this was due to traits which were recessive (or hidden) behind the dominant round trait. He also determined that each parent had to have two copies of each trait, one of which was passed down to the offspring. This became known as the Principle of Independent Segregation (or Mendel's Law of Segregation) and states that each allele, or copy of a trait, has an equal likelihood of being passed on to the progeny. Therefore, when both parents have only dominant genes, the progeny will all be dominant and when both parents have only recessive genes, the progeny will all be recessive but if both parents

Figure 3.5: *Arabidopsis thaliana*

have both types of genes, then the progeny will have varying combinations of these genes in different proportions.

A. thaliana is a model plant widely used in genetic studies, having a number of qualities that make it attractive for genetic study. It has a relatively short genome with about 157 million base-pairs encoding 27,000 genes and was the first plant to have its entire genome sequenced in 2000. Additionally its very fast life-cycle of around 6 weeks from germination to mature seed makes it ideal to longitudinal study and allows comparison of multiple genomes from the same species.

In the 2005 article in Nature it was suggested that some offspring of this plant inherit genetic material from individuals *other than those of the direct parents* (Lolle et al, 2005). This paper attracted a great deal of attention, with letters expressing both supporting (Weigel & Jürgens 2005),(Chaudhury 2005), (Ray 2005) and contrary opinions (Peng *et al.* 2006) (Mercier *et al.*

2008).

The findings that lie at the centre of this paper relate to the *HOTHEAD (HTH)* gene of *A. thaliana*, which impacts on formation of the flower and epidermis (Figure 3.5). Mutated forms of this *HOTHEAD gene (labelled hth)* result in a malformed flower (Figure 3.6). Individuals were studied that had the hothead mutation *(hth)* on both of their DNA strands (these plants were homozygous for this recessive mutant allele). When these *hth* mutants were allowed to self-fertilise, amazingly around 10% of the progeny were of the wild type *(HTH)* (See Figure 3.7) - even though this genetic information was not detected in either parent!

In Figure 3.8 the crossover of a wildtype (*HTH*) and mutant (*hth*) *A. thaliana* is shown. The three possible offspring are *HTH/HTH*, *HTH/hth* and *hth/hth* where *hth/hth* is the mutant plant as shown in Figure 3.6. When two of these pure mutants (*hth/hth*) are crossed over the result is shown in Figure 3.9. Following Mendels theories we can predict that since each parent has only the mutant copy of the gene, all progeny should inherit those mutant copies. In this Figure it is shown that approximately 90% of the time a mutant plant (*hth/hth*) is produced as expected. Surprisingly, as illustrated in the Figure, approximately 10% of the time the wildtype flowering (See Figure 3.7) *A. thaliana* (*HTH/hth*) is produced. This illustrates the plant using the genetic "cache "of information to revert back to the preferred state. This use of information not present in the parent illustrates non-Mendelian inheritance. This 10% rate of correction is a far higher rate than can be accounted for by random point mutations, which would generally occur with

Figure 3.6: *Arabidopsis thaliana* - *hth* Mutant

Figure 3.7: *Arabidopsis thaliana* - *HTH* Wildtype

a frequency of the order of 1 per billions per allele per generation (Weigel and Jrgens, 2005). These findings are not consistent with the standard Mendelian model of inheritance (as shown in Figure 3.8) and led to the controversial non-Mendelian theory - for an accessible overview of these findings see (Agrawal, 2005).

Lolle's explanation centres on a genetic repair process that restores the normal *HTH* allele using an ancestral repair template. Lolle posits that each

plant carries an additional "cache"of genetic information derived from each individual's ancestors. This cache is then used to drive a template-directed repair process, which restores the validity of these mutant individuals. While most explanations for this non-Mendelian inheritance rely on RNA mediated inheritance (Rassoulzadegan, 2006), our model is akin to the archival DNA explanation offered by (Ray, 2005). Ray suggests a form of encrypted DNA records the ancestral genomic data that supports the repair process. This archival DNA is not responsive to the techniques such as Southern blotting or PCR that are used to process DNA[1].

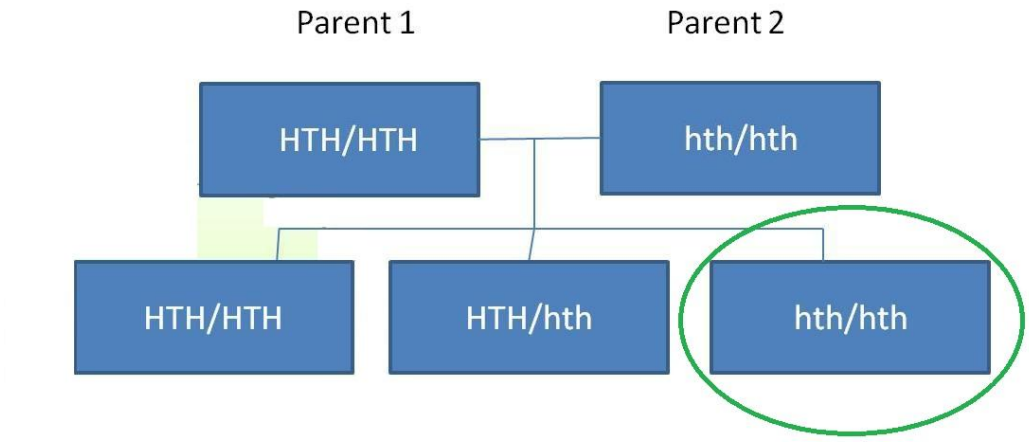

Figure 3.8: Normal Mendelian Inheritance

This advance in biological research suggests one method which has been proposed to be used in nature (Lolle *et al.* 2005) for correcting genetic defects or errors. This method allows many erroneous individuals to fix themselves

[1]This information is result of knowledge gained while working in Dr. Susan Lolle's lab during Summer 2009 under a scholarship awarded by ICUF

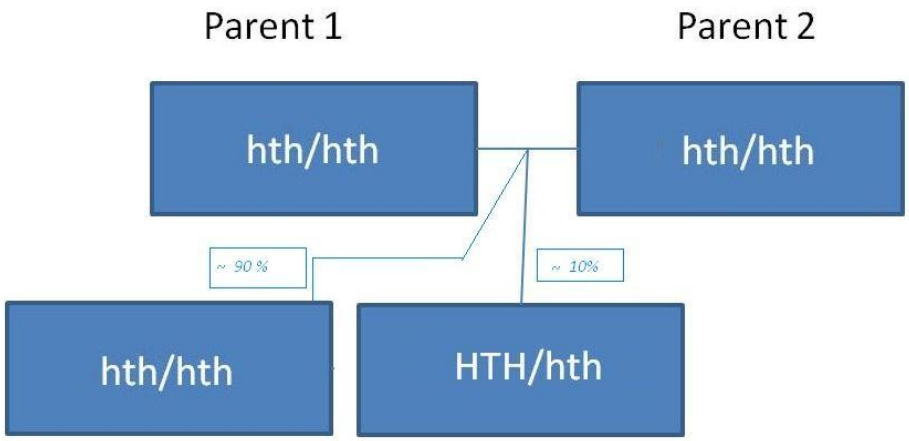

Figure 3.9: Non Mendelian Inheritance

and return to the population.

This advance however was widely disputed in the academic field of biology (Peng *et al.* 2006) (Mercier *et al.* 2008). Until such a time as supporting results are produced this theory can be viewed as a tentative hypothesis.

This thesis investigates whether this proposal, when implemented as part of the overall analogy, can enable EA to produce solutions to constraint based problems. I investigate whether non-Mendelian ancestral repair templates are useful in the field of Computer Science when compared to Mendelian templates. This thesis is concerned with the impact of ancestral repair on the field of EO regardless of whether the proposal is further supported or in fact disproved.

In the overall analogy this proposed repair mechanism in the *A. thaliana* could be viewed as a repair mechanism to enable individuals to repair themselves in order to continue in the environment. This method may allow the

EA to produce valid solutions to constraint based problems by fixing invalid solutions.

In the *Arabidopsis thaliana* plant a known mutation is a glue like substance on the leaves which prevents the plant from blooming fully. This mutation is called "organ fusion"and allows pollen to germinate on all plant surfaces as opposed to the usual case where it can only germinate on the stigma. Another mutation spotted in this plant is double flowering. Neither of these mutations in the *A. thaliana* cause the plant to be invalid though they do make it more difficult to propagate. For this reason the mutation method used in the EO presented is a validity preserving mutation method (swap mutation). This method mutates a small percentage of individuals to maintain diversity but does not cause individuals to become invalid.

I have adapted the standard evolutionary algorithm to include this repair method or GeneRepair which is modelled on (but not identical to) that found in nature in the *A. thaliana* plant and is now found in the EA (FitzGerald & O'Donoghue 2008). Our EA will now maintain the normal population as well as an ancestral archive for each individual. This archive will be used to carry out GeneRepair on invalid individuals.

Thus far we have seen how the original analogy between EA and natural evolution was created and developed over time. Unfortunately this EA is unsuited to solving constraint based problems and has not been updated to mirror biological research in the same way that it mirrors advances in computer science research. I will now look at how the original analogy can be updated to incorporate this new exciting advance in biology. In order to

adapt this new feature the links in the analogy between nature and the EA must be extended. I can now clearly identify a link between constraints in the EA and in nature. I am going to investigate if this repair mechanism can be adapted and applied to the task of handling constraints with an EA. I will now explore the possibility of this expanded analogy by examining both sides of the analogy.

3.7 Extending the Evolutionary Analogy to Include Genetic Repair

While EA are efficient at solving large complex problems they are ill-suited to solving constraint based problems (Eiben 2001). When constraints are broken EA do not have an innate repair mechanism. This thesis presents a novel approach which adapts the standard EA which we hope will allow it to produce valid solutions to constrained problems efficiently by mirroring nature. The EA can now incorporate the natural repair mechanism found in the *A. thaliana* plant and this further strengthens the analogy between EA and biological evolution. The extended analogy can be seen in Table 3.3.

The constraints facing our sample problem domain (the TSP - See Section 2.3) are prohibition of duplicate cities as well as the inclusion of all cities. This means that each city must appear in a tour once and once only. If these constraints are broken the individual is no longer valid. A constraint violation of this type could be introduced during the crossover phase of evolution or during the mutation phase. The constraints that we see on the *A.*

Table 3.3: Extended Analogy between EA with GeneRepair& Biological Evolution

Biological Evolution	**Evolutionary Algorithm**
Genetic Error	Missing Sequence
hothead hth mutants	Invalid Solution (tour)
Non-Parental Sequence Restoration	City Re-insertion
Genetic Repair by restoration	GeneRepair
DNA/RNA Cache	Complete Solution Cache

thaliana is that it produces a wildtype offspring rather than a mutant. That is, *A. thaliana* always tries to generate a feasible plant. While the mutant offspring is still valid the wildtype is seen to be fitter biologically as it has a much higher chance of reproducing. The mutant offspring (as described above), also known as a "fusion mutant"allows germination to occur on the full plant surface rather than restricting it to the stigma. This type of mutation breaks the validity constraints and causes the individual organisms to die during construction as this is a non-viable phenotype construction. Typically infeasible individuals are marked in nature by "abortion events"where there is a failure to construct a valid phenotype from the genotype. Genetic repair appears to be one means of correcting some of these infeasible individuals.

3.8 Dealing with Constraint Violations

In the previous section I have mentioned that constraint violations can be easily introduced to the population of the EA. Similar to this, constraint violations can also be easily introduced into the *A. thaliana* causing it to produce a fusion mutant offspring. The way in which the *A. thaliana* repairs this constraint violation is to use information from a generation previous to the parent as a template. Repair in the *A. thaliana* plant appears to occur when the mutant creates offspring while repair in the EA will occur the instant the mutant is generated BEFORE it enters the population. This means that there is a generation of delay allowing the mutants to form in the *A thaliana* life cycle but the mutants are prohibited from forming in the EA.

This proposed repair information of the *A thaliana* is inherited through a Non-Mendelian inheritance process which means that it is not inherited from the parents as you might suspect. This means that the erroneous information was inherited as normal in the construction of the offspring while the genetic information used to carry out the repair was not inherited in the same Mendelian way. The genetic information used to carry out the repair is seen to exist in the grandparent of the individual but may have originated in a generation previous to this.[1] This template is used as a type of cache to repair the violation. The plant then produces an offspring which has reverted back to the non-mutant form and does not possess the fused organs(As shown in Figure 3.5. The details of how exactly this is performed in the *A. thaliana*

[1]Personal communication with Dr Susan J. Lolle

are not completely clear as research is ongoing.

The novel approach addressed in this thesis to enforce constraints in EA is to use a previous generation as a repair template to correct constraint violations. In order to do this I store the ancestors of each individual in the population. In this thesis I compare their use as repair templates and examine which template should be used for various conditions. The *A. thaliana* plant was shown to have a *parallel path of inheritance*. Robert Pruitt used this term in an interview with the Washington Post which appeared on March 23rd 2005 to describe what appears to occur in addition to standard Mendelian inheritance, where information from generations previous to the parent is used to ensure that constraints are obeyed in the newest generation. This has been mirrored in the EO presented which uses generations previous to the parent as templates to repair constraint violations in the newest generation. This strengthens the analogy between EA and biological evolution by enforcing the same method of constraint violation repair. This analogy gave us an inference and motivated us to investigate whether the ancestral repair generates strong solutions.

3.8.1 Representation

Representation is the word given to describe how individuals are depicted in the environment. This *representation* is part of the overall analogy so in this subsection I will show the representation of the *A. thaliana* and the corresponding representation of the individuals in the presented EA. *A. thaliana* is made up of approximately 157 million base pairs and 5 chromosomes encod-

ing 27,000 genes. Each individual in the EA represents a tour or a possible solution to the TSP in the case of experiments illustrated in Chapter 5. This means that each individual contains all of the cities in the order that they are visited. The cities are represented by integers for ease of computation (as suggested by (Luger 2002) on p 476).

3.8.2 Fitness

Fitness is a measure of the health or strength of an individual. The formula to determine the fitness of an individual (fitness formula) is completely dependent on the environment where the individual exists. The fitness of the A. *thaliana* is measured by testing if the plant is alive or dead and how well it survives. The fitness of living plants could be equated to one while the fitness of dead plants could be zero. It could also be said that when the mutant *A. thaliana* is compared to the wildtype, wildtype is seen to be fitter. Similarly, the fitness of each individual in the EA is the calculation of its tour length. Shorter tours represent higher fitness while longer tours have lower fitness. This problem (the TSP) can therefore be described as a minimisation problem as the goal is to minimise the cost function. In the EA the fitness function adds the distance between each set of neighbouring cities, including the distance from the last city back to the starting city to give the tour length or fitness of the individual.

3.8.3 Optimisation

Optimisation is the term given to the improvement of the population in an EA. In biological evolution this would be the improvement of the *A. thaliana* plant in comparison to previous generations competing for the same resources and within the same space. This is the fundamental idea behind evolution - there are occasional large improvements with small genetic drifts in between over a large number of generations. As explained previously the theory of *natural selection* and *survival of the fittest* suggests that as a species evolves, it becomes stronger in relation to the environment in which it exists. The evolution of the EA can be shown by Holland's Schema Theorem which states that short, low-order, schemata with above-average fitness increase exponentially in successive generations. This is the EA counterpart to the notion of survival of the fittest.

3.9 Conclusion

Many Computer Science inventions and theories are based on analogies with biology. The design of the SLR camera can be seen as being closely related to the design of the human eye. The motherboard in a standard desktop computer can be seen to be the brain while the hard-drive is often described as memory. Evolutionary computation is no different in that it is analogically linked with natural evolution. In this chapter I have shown the pre-existing analogy between evolution in nature and EA. While the computer science and biological side of the analogy have been researched in great detail the

central analogy has remained the same. This means that the drawbacks of EA remain although the EA may be adapted to enable them to produce valid solutions to constraint based problems. I have updated the analogy to reflect advances in biological research with the introduction of a natural non-Mendelian repair mechanism (Lolle *et al.* 2005) proposed in the *Arabidopsis thaliana* plant. In this chapter I have presented the two sides of the analogy in the form of representation, constraints and basic evolution between the EA and the *Arabidopsis thaliana* plant. I have shown how many properties of the EA have a *real-life* counterpart in the *Arabidopsis thaliana* plant. The implications of this analogy will be investigated and tested throughout the thesis to show how it enables EA to handle constraints in a similar manner to that of the *Arabidopsis thaliana* plant. In order to concrete this analogy with the reader I have created a simple reference table (See Table 3.4) to show the computer science and biological evolution counterparts to terms continuously mentioned throughout the thesis.

In Chapter 4 I will show how this new addition to the analogy is implemented. I will explain the process of GeneRepair and investigate whether it can adapt EA to produce valid solutions to constraint based problems. In this following chapters I will investigate a central tenet of Lolles controversial hypothesis - whether repair using a non-Mendelian template outperforms the Mendelian alternative, for computational problems.

Table 3.4: Analogy between EA & Biological Evolution

Term	Biological Evolution	Evolutionary Strategy
Individual	*Arabidopsis thaliana*	TSP tour
Generate	First creation of plant	Creation of population of individuals
Select	Natural Selection or Survival of the Fittest	Truncation or Tournament
Crossover	Reproduction	Single Point Crossover
Mutate	Organ Fusion	Swap Mutation
Constraint	Wildtype is Preferred	No duplication
Constraint Violation	Organ Fusion	Duplication of Cities
Repair	Revertant Wildtype through non-Mendelian Inheritance	GeneRepair using non-Mendelian ancestral template

Chapter 4

The GeneRepair Operator

4.1 Introduction

Evolutionary algorithms (EA) are problem solving techniques often viewed as being analogous with natural evolution. EA are used successfully by computer scientists to solve a wide range of different problem sets but are ill-suited to constraint based problems (Eiben 2001). A recent biological suggestion has enabled the extension of the analogy between nature and biology which might enable EA to handle constraint based problems. In Section 3.6 I explained the biological research which has proposed a new and controversial genetic repair hypothesis in the *A. thaliana* plant. I will now introduce the repair operator which attempts to mirror this repair process within the EA. This repair mechanism enables the EA to produce valid solutions to constraint based problems as errors introduced to the population are no longer a critical issue.

4.2 Introduction of Validity Errors

An analogy exists between natural evolution and artificial computer based evolution. In Chapter 3 we saw that since EA were created a repair mechanism was not generally incorporated. This means that EA did not incorporate a way to repair errors in individuals. An error is a constraint violation in the individual thus causing the solution to be invalid. These constraint breakages or invalid individuals are easily produced as I shall now illustrate.

Figure 4.1 (top) shows two individuals in a population that will be crossed over to create two new offspring. Each of these individuals represents a six city solution to a TSP problem. The constraint of interest for this problem is that each city must appear once and only once in each solution.

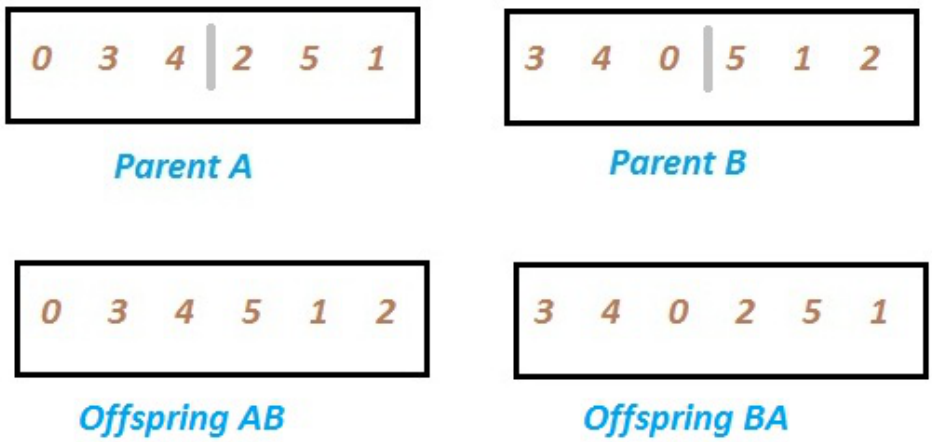

Figure 4.1: Parents A and B with Offspring AB and BA

Figure 4.1 (bottom) shows the two offspring produced by Parent A and

Parent B, using single point crossover, where the crossover point in the parents is illustrated by a grey line. These offspring do not break any of the problem constraints and so it would appear that this algorithm is able to successfully produce solutions to a constraint based problem.

In the next example we will see that the validity of the solutions produced by crossover cannot be guaranteed. If we look at Figure 4.2 we see Parent C and Parent D produce two offspring with single point crossover. The offspring in Figure 4.2 break the problem constraint as City 2 appears more than once in offspring CD and also City 3 appears more than once in offspring DC. CD also breaks the problem constraint as City 3 does not appear and DC breaks the problem constraint by not containing City 2. It should be noted that with this fixed length representation, for each *duplicate city* constraint violation there is also a *missing city* constraint violation. The individuals CD and DC can therefore not be used as their fitness cannot be assessed, nor can they produce feasible offspring.

A number of techniques have been introduced to counteract this problem and allow EA to handle this invalidity, as discussed in Chapter 2. The technique developed and explored in this thesis is not only suitable for many problems it is also based on nature. It extends the underlying analogy by incorporating new biological information into the analogy and producing a new extended and cohesive model for EO.

This section has shown how constraint violations are introduced using standard single point crossover. The second way for errors to be introduced to the population is through point mutation. Holland said of mutation that

Figure 4.2: Parents C and D with Offspring CD and DC

"*Though mutation is one of the most familiar of the genetic operators, its role in adaptation is frequently misinterpreted. In genetics, mutation is a process wherein one allele of a gene is randomly replaced by (or modified to) another to yield a new structure.*"(Holland 1975) Point mutation, which is a specific type of mutation, can introduce errors to the individual. Point mutation replaces an arbitrarily chosen allele with a different allele. In a binary representation point mutation flips the identified bit so if it is a 1 it would change to a 0 and vise versa. In the representation used in this thesis where the individual is an ordered list of integers the identified allele would be replaced with a different integer (such as a positive integer within the required range). This mutation would frequently cause a duplication error which is a constraint violation. In the experiments described in this thesis I have not used point mutation but instead have used swap mutation. As the

name suggests swap mutation swaps two alleles in the individual which does not break the problem constraints as no errors are introduced though this method (Luger 2002). The reason that swap mutation was chosen over point mutation is so that the focus is on repairing errors arising from crossover. Swap mutation also fits into the underlying analogy.

4.3 Impact of Errors on Fitness

Each individual in the EO population has an associated fitness value. The fitness of the individual is a rating depending on the suitability of the individual to the environment. If Individual A has a better or stronger fitness than Individual B this means that Individual A is a superior solution to the given problem and is more suitable to the current environment. To illustrate this we can compare the fitness of two individuals in our given TSP. The fitness of each of these individuals is a calculation of their tour length. Therefore the fittest individual will be the individual with the smallest "cost"for this minimisation problem (FitzGerald & O'Donoghue 2008). In a maximisation problem the individual with the highest fitness value would be the fittest individual in the population.

In Figure 4.3 a 6 City TSP is shown. The objective is to find the shortest route that visits each of the cities once and once only. In Figure 4.4 a possible solution to this problem is illustrated. In Figure 4.5 the optimal route is shown. The individual representing the tour shown in 4.4 would have a larger cost than the optimal tour shown in Figure 4.5. The reason for this is that the fitness for this problem represents tour length and the tour

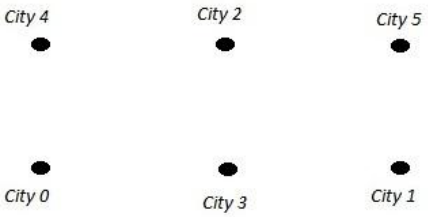

Figure 4.3: Travelling Salesman Problem - Sample 6 City Problem

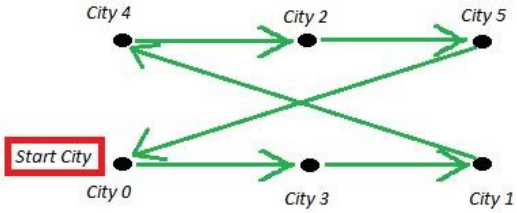

Figure 4.4: Travelling Salesman Problem - Sample 6 City Tour

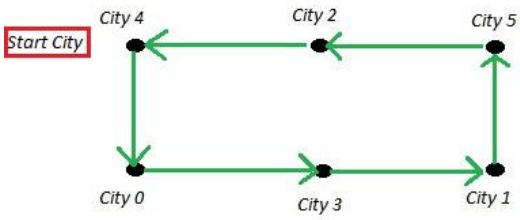

Figure 4.5: 6 City TSP Optimal Route

in Figure 4.4 is longer than the tour illustrated in Figure 4.5.

If a constraint was broken in one of these solutions and a city appeared twice, this would often lower the distance travelled and so produce a low tour length. This is particularly true when a city and its duplicate appear adja-

cent to one another on the tour as one inter-city distance will have the null cost of 0. This would make the invalid individual appear to be fitter than the other valid individuals potentially increasing its impact on subsequent generations. This is one of the reasons that repair (or another constraint handling mechanism) is extremely important. It prohibits false positive fitness values from being created.

4.4 The GeneRepair Adjunct Operator

GeneRepair is the biologically inspired technique used in conjunction with EA to repair invalid solutions in the population allowing them to be reintroduced to the population of feasible solutions. The technique is based on the Non-Mendelian repair technique proposed for the *A. thaliana* plant (Lolle *et al.* 2005).

This technique builds on previous research into template directed repair by Mitchell *et al* (See Section 2.4.5) in 2000 which introduced a cleanup operator to repair invalid solutions. GeneRepair uses ancestral information to replace invalid alleles in an individual. In its simplest form, ancestral repair simply replaces any erroneous alleles with a corresponding allele from an (ancestral) repair template. We know that it has been proposed that the *A. thaliana* uses non-Mendelian ancestral information to repair invalid genes so GeneRepair also explores the use of non-Mendelian information as a template for repair. When an invalid (duplicate) allele is detected it can be replaced by another allele from the repair template. As shown previously the standard EO has five steps (Fogel 1994), three of which are repeated

for some number of generations. GeneRepair fits into these steps to allow a standard EO to handle problem constraints:

1. Generate
2. Select
3. Crossover
4. Mutate
5. **GENEREPAIR**
6. Repeat Steps 2 - 5

We decompose this new adjunct genetic operator (GeneRepair) into two distinct phases of *error detection* followed by *error correction*.

4.4.1 Error Detection

Error detection occurs whenever the genotype cannot generate a valid phenotype. That is, the solution generated is not a valid, complete or usable solution to the given problem. In our TSP representation erroneous alleles are identified as duplicate cities. However different representations may result in different invalidity signatures: missing cities or cities not within the range of the given problem. This process may also identify the specific alleles (or sequences of alleles) that will undergo repair. Error detection can be seen as metaheuristic as it makes few direct assumptions about the underlying problem and could be based on "phenotype "construction. In Figure 2.4

(Section 2.4.5 of Chapter 2) error detection identifies a constraint violation because there is a duplicate of City 2 in the invalid individual. Which of these will be replaced is arbitrated by the direction of detection and this will be explained in Section 4.7.

4.4.2 Error Correction

In Section 2.4.5 we saw how Template Repair uses a repair template to support the correction of an invalid individual in the EO. In Figure 2.4 (Section 2.4.5 of Chapter 2) a template was used to repair an invalid individual. This template was simply a valid individual. GeneRepair is a form of template repair in that it uses a template to advise the EO on the properties of a valid individual. In this thesis we modify template repair so that it does not use a static sequence as the repair template. Instead the template is a direct ancestor of the individual being repaired. This means that the template can change from one generation to the next. The order of items in the GeneRepair template (the location of each element within the template) also dictates the repaired individual. Different templates and corrective strategies yield different solutions. Using the alternative constraint handling mechanisms individuals with a small constraint violation could be lost forever but GeneRepair restores these individuals to full validity enabling them to participate in the population. Now we will look at how this dynamic version of GeneRepair repairs invalid individuals. GeneRepair archives the ancestral data of the individual and uses this as the repair template. This thesis explores and compares the relative efficiency of different templates.

4.5 Repair using the Parent template

We have seen how repair templates can be used to enforce constraints on individuals. This section shows an example of how an ancestral repair template can be used in a similar way to repair errors in an invalid individual. Previously, in Figure 4.2 it was illustrated how two parents could produce invalid offspring during crossover. So, to repair *Offspring DC* from this Figure we can use one parent as a template for repair. However, there is a choice of two parents, *Parent C* or *Parent D*, to act as a template. For this example I shall randomly choose *Parent D*. (Later in this thesis I will explore this choice by comparing the use of different ancestors and investigating the results produced.) Figure 4.6 illustrates how *Parent D* and *Offspring DC* (from Figure 4.2) have been identified as the Repair Template and Invalid Individual respectively.

GeneRepair first implements *Error Detection* to detect the constraint violations in the individual. This detection phase identifies that City 3 is a duplicate and therefore an error. GeneRepair then implements the *Error Correction* phase. This step replaces the error City 3 with the missing City 2 using the *Repair Template* to dictate the order of the elements repaired. In Figure 4.7 we can see that City 3 is identified as an extra city in the *Invalid Individual* (highlighted with a red box). As City 3 is a duplicated there is two City 3s that could be detected as the error. The direction of error detection arbitrates which City 3 is the error and this will be explained in Section 4.7. As we will see in later chapters the "direction"of GeneRepair impacts directly on the final solution and shall be explained and explored in later

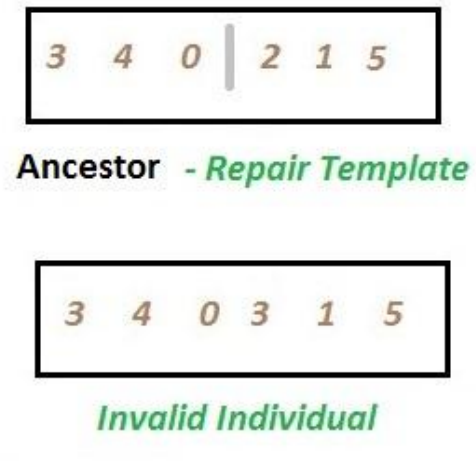

Figure 4.6: Parent Template Repair - Template and Individual prior to Repair

sections. The identified duplicate is then replaced with City 2 from the *Repair Template* (the Parent) which is highlighted with an orange box. We can see the previously missing City 2 is now present in the *Repaired Individual* as highlighted with a green box. The *Repaired Individual* now obeys the problem constraint. This repaired individual has been transformed using GeneRepair. The GeneRepair technique used the *Parent D* as a repair template to correct the invalidity on the *Offspring DC*. This *Repaired Individual* now replaces the *Invalid Individual* in the population.

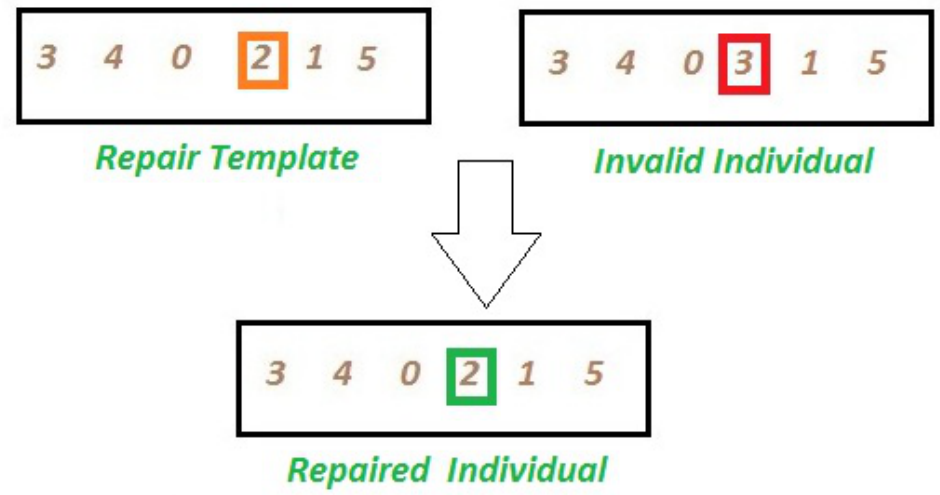

Figure 4.7: Parent Template Repair

4.6 Ancestry

A. thaliana uses non-Mendelian information to repair invalidity found in the offspring. The information used to repair this invalidity, also known as the "cache", is the plant's repair template. It is a central claim of Lolle's hypothesis that this repair template information does not originate in the parent but is present in the grandparent (Lolle *et al.* 2005). In order to incorporate an analogous strategy into the EO, the GeneRepair technique can use the parents, grandparents or great-grandparents as a repair template. Each of these generations are stored by the EO along with the present generation. These templates can then be used as repair templates in the same way as the parent was used as a template in Section 4.5 so that the efficiency of each template can be compared.

In Figure 4.8 we can see how the parent, grandparent or great-grandparent can be used as repair templates to repair the invalid individual. In this example there is only one error and the choice of template does not affect the resultant *Repaired Individual*. In reality the invalid individual being repaired would often have more than one error and so this thesis investigates whether the choice of template will directly affect the resultant repaired individual.

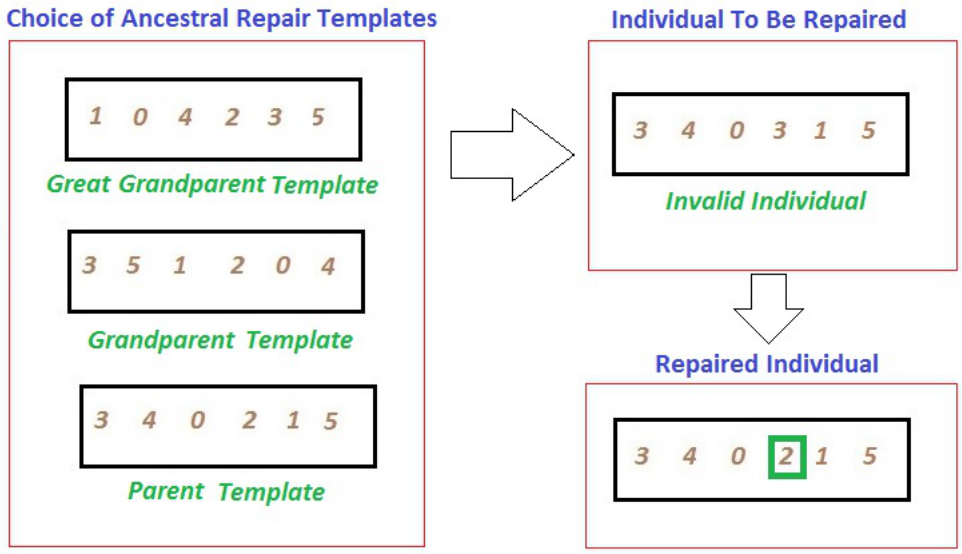

Figure 4.8: GeneRepair - A choice of Ancestral Repair Templates

4.7 Direction of Error Detection

As shown in Section 4.4 GeneRepair has two phases, error detection and error correction. When the errors in the individual are repaired the resultant

individual will depend on a number of different variables such as the template used. Another such variable that impacts on the resultant individual is the direction of repair. For example in Figure 4.9 we can see that the City 2 appears twice. This has been detected as an error during the error detection phase as the problem constraint states that each city must appear once only. Whether the red City 2 or the blue City 2 in the diagram are replaced depends on the direction of repair, specifically, on the direction of the error detection process. The GeneRepair model always starts at one end (City 0 or City (n-1)) and works either from left-to-right or from right-to-left. The only exception to this is when Reduced Redundancy Representation is used (See Section 5.6.1) in which case it starts from City 1 or City (n-1). During traversal of the solution, it is always the 1st instance of a city that is encountered which is tagged as the duplicate.

If error detection is carried out in a left to right direction, the resultant individual will be the *Repaired Individual A* illustrated in Figure 4.9. This is because City 2 (in red) has been identified as a duplicate city during the *error detection* phase of GeneRepair.

If the repair however is carried out in a right to left direction the resultant repaired individual will be the *Repaired Individual B* as illustrated in Figure 4.9. This is because GeneRepair scans the *Invalid Individual* from right to left and replaces the first instance of the identified duplicate, in this case City 2 (blue), found. Figure 4.9 illustrates the effect of repair direction where one error is being repaired. As the number of errors increases the difference between the repaired individuals would be much greater. In Figure

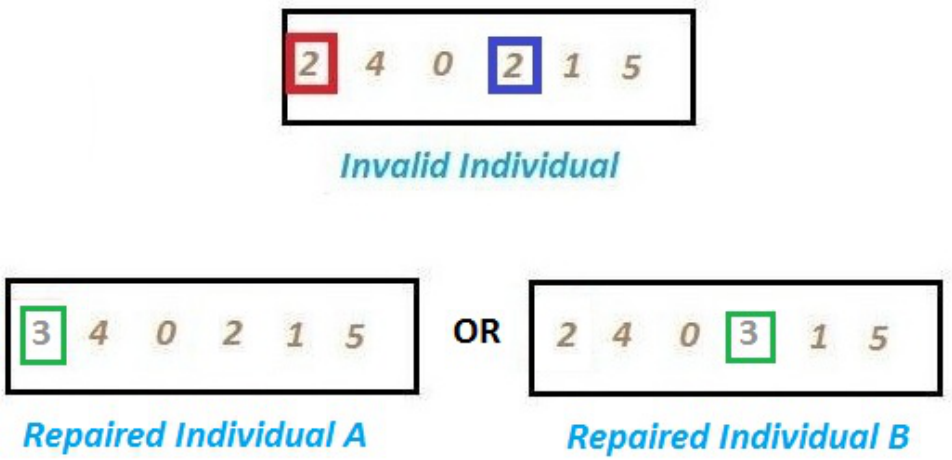

Figure 4.9: GeneRepair - Effect of Repair Direction

4.10 we can see an individual with two City 3s and two City 5s. Whether the cities surrounded by the blue boxes or the cities surrounded by the red boxes will be replaced depends on the direction of detection. Each identified duplicate is replaced with the first *Missing City* found in the repair template. This is illustrated in Figure 4.10. We can see from this Figure that the resultant individuals would be drastically different depending on which alleles are repaired.

In this section I have described the parameter of repair direction and illustrated the effect of two repair directions on resultant individuals. There is however a third repair direction in addition to the right to left and left to right repair directions previously described (FitzGerald & O'Donoghue 2008). Individuals can be repaired in a random and varying direction. For each

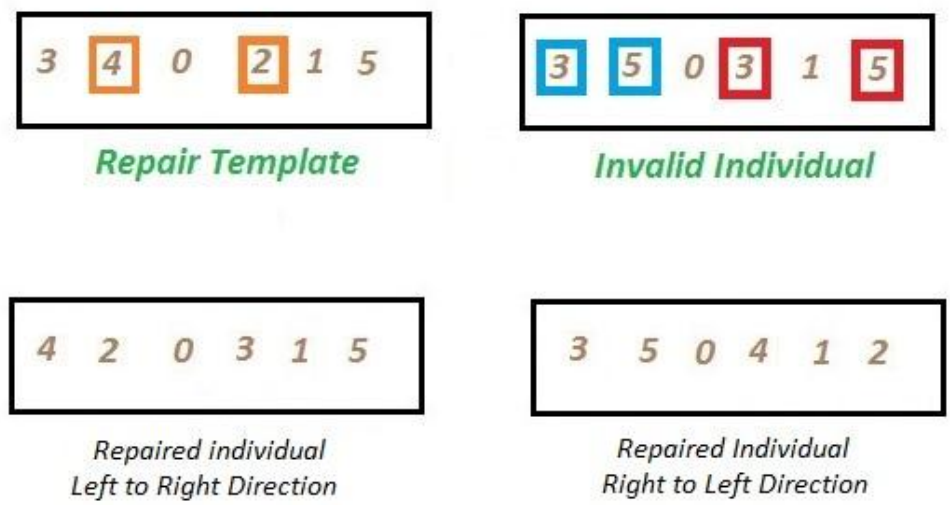

Figure 4.10: Individual with More than One Constraint Violation

individual this method chooses randomly between right to left direction and left to right direction. This choice is made at the start of repairing each individual. Then, the entire solution is repaired in that direction. This thesis will examine the three different repair direction choices and compare the results produced by each in order to identify a superior repair direction to be used by GeneRepair. In Chapter 5 the effects of direction (and varying direction) on the results produced will be examined.

4.8 Template Fitness

In Section 4.3 we looked at the definition of fitness in an EA. Each individual in the EA has an associated fitness. When choosing the repair template,

GeneRepair may use the fitness level of the template to decide which template to store or to use for the repair process. The three choices would be to use the fittest template for repair, the least fit template or a random template from the templates available. In Chapter 5 we will explore the influence of each of these choices on the resultant repaired individuals.

4.9 Implementation of GeneRepair

GeneRepair was implemented as part of an EO using the Java programming language. As shown in the class diagram (See Figure A.1 in Appendix) the package was made up of 8 classes and utilises one imported package. The only other resource used is the problem data, in this case this is the TSP text files, the benchmark TSPLIB problem data. Further information on the implementation can be found in the Appendix (A).

4.9.1 Implementation Discussion

In the Appendix (See A) I have shown the computational implementation of the GeneRepair technique presented in this thesis. While this GeneRepair technique was presented in a previous thesis (Mitchell 2007) the implementation and research in this thesis are novel. The implementation described above differs from previous descriptions. The design of the `EvolutionaryOptimisation` was originally based on Mitchell's model but has been changed significantly. All of the other classes are new and designed for this implementation of GeneRepair. To overcome some bottlenecks in Mitchell's model, the `Map` class

was introduced to alleviate unnecessary complexity of previous implementations where the *(x,y)* co-ordinates of the TSP were scattered throughout the EO disallowing separation between representation and problem definition. The tour manager class allows the EO to carry out calculations on the problem data while allowing a significant degree of separation between the problem representation and the repair technique employed. The implementation used in this thesis only requires the use of an ordered list of integers, thereby making minimal assumptions about the underlying problem. The implementation of the GeneRepair technique also differs from (Mitchell 2007) as the invalid individual is scanned for constraint violations, the template is scanned for violation repairs and the final scan repairs the first violation of each constraint breakage found rather than using a flag system and repairing un-flagged elements of the individual. For integrity of results the MersenneTwister package is used as this is more reliable than the java.util.Random method (Luke 2009). The principal item to note in the implementation described in this section is that the EO stores the parent, grandparent and great-grandparent of every individual and ensures that regardless of mutation or selection mechanisms used, the correct ancestors are associated with the individual in the current population.

4.10 General EO Parameters and GeneRepair

There are other parameters which are not specific to the GeneRepair technique that impact on the quality of the EO results. These parameters are used both when GeneRepair is used and when it is not. The parameters in question are population size, number of generations and mutation rate.

4.10.1 Population Size

The size of the population can directly affect the diversity maintained across the population as shown by C. Ahn and R. S. Ramakrishna who investigate a method to create appropriate population size (Ahn & Ramakrishna 2002). If a population is small it may not be diverse enough, while large populations may be too diverse. The techniques for finding the correct population size to use are limited to specific problems and can only ever act as a guide rather than a rule. The *A. thaliana* reproduces mostly by selfing leading to little diversity in a population. Selfing is the main process for propagation - common to many crop plants(Hopkins *et al.* 2011). The word selfing refers to the act of reproduction without the need for another plant, that is the plant reproduces independently or by the acting of "selfing". The *A. thaliana* reproducing mostly by selfing may suggest that a smaller population should be used with the corresponding repair method (GeneRepair) on the EO side of the analogy. We will investigate the effect of population size on GeneRepair in Section 5.5.7.

4.10.2 Number of Generations

The length of execution of an EO is also not specific to the GeneRepair technique but is an EO parameter. This parameter specifies how long the experiment will run. It can range from number of generations to a set time or until the some milestone has been reached. There has been research (Mitchell 2007) into the results produced by of letting the EO run for a longer execution time and if the difference in results outweigh the computation cost. In Chapter 5 we will investigate the effects of running the experiments for varying numbers of generations. I will show the influence of this parameter on the resulting repaired individuals and on the ancestral template choice. Among the questions that motivate this thesis, it would be beneficial to know if GeneRepair works well in populations containing a lot of homogeneity - such as in *A.thaliana*.

4.10.3 Mutation Rate

The mutation rate specifies the number of individuals that will be mutated per generation. Evolutionary algorithms can benefit from both high mutation and low mutation depending on the problem size and/or representation. There is also findings that adaptive mutation (Smith & Fogarty 1996) is preferred for specific problems. While much research has been carried out on mutation and which type and rate to use there is a suggestion to use 1/l where l denotes the bit string length in a genetic algorithm, but there are also contrasting views and no agreed upon discipline-wide guideline (Tate & Smith 1993)(Bäck 1993). In the following Chapter I will investigate the ef-

fect of mutation rate on the choice of ancestral repair template for a number of TSPs and show how the repair templates behave near the optimal (from those investigated) mutation rates found.

4.11 GeneRepair Parameter Choices

Previously in this chapter the GeneRepair technique has been introduced. This technique, which is based on the repair mechanism of the *Arabidopsis thaliana* plant, offers a novel approach to allowing EO to produce valid solutions to constraint based problems. The technique uses an invalid individual's ancestor as a repair template to correct errors in that invalid individual. There are a number of parameters which influence the resultant repaired individual. Each parameter can have a huge influence on the fitness of the repaired individual. In the following chapters I will investigate the effects of each parameter and suggest the parameter values to use in conjunction with GeneRepair for two different constraint based problems.

The first choice of parameter is which generation of ancestor to use as a repair template. I investigate and compare the use of *parent*, *grandparent* and *great-grandparent* templates in this thesis. The GeneRepair technique also allows you to use a more removed ancestor from further back in the lineage than the great-grandparent but this calls for more storage space and longer computation times and is not presented in this thesis.

At the time this thesis was undertaken, Lolle and others were primarily interested in recent generations - especially the grandparent generation. In line with this work, this thesis has also focused on the grandparent generation

as well as its immediate neighbours. This tactic of course, focuses on the central contention claim of Lolle *et al* - that the proposed genetic repair process is non-Mendelian. That is, the repaired individual contains genetic information that does not originate in the parent generation. Such a repair mechanism suggests that this genetic repair process offers the competitive advantage over competing (ie Mendelian) hypotheses.

The next choice of parameter is which of the ancestors to use. Natural beings have two parents, four grandparents and eight great-grandparents. The GeneRepair model stores two ancestors for each generation. This means that there is a choice of two parents, two grandparents or two great-grandparents. The choice of which two ancestors are stored from the grandparent and great-grandparent generation is arbitrary.

4.12 Conclusion

EO are excellent at solving difficult problems and are widely used in Computer Science. They solve problems by evolving a solution much like the evolution of any species in nature. EO are often viewed as being analogous with natural evolution to solve problems in Computer Science. When evolutionary algorithms and more specifically genetic algorithms were first created no constraint handling technique was included on the nature side (source) of the analogy. For this reason EO constraint handling mechanisms often use modified operators. These operators break the analogy with nature. These modified operators are often not biologically inspired and can be very problem specific (See Chapter 2). In Chapter 3 I showed how this thesis

extends the analogy between natural and simulated evolution by attempting to mirror the repair mechanism found in the *Arabidopsis thaliana* plant. This thesis presents a repair technique called GeneRepair. GeneRepair allows EO to repair invalid individuals in the population. This ability allows the EO to handle constraints without modifying any other part of the EO. GeneRepair makes use of recent ancestors to bootstrap individuals with somewhat minor genetic defects, that is defects that can be repaired with ancestral genomic data. Unlike many existing genetic repair operators, this technique is not heuristic but is metaheuristic - repairing the representation rather than addressing the underlying problem specifically. Furthermore, repair cannot solve the given problem *ab initio*, whereas many heuristic techniques are often used to solve these problems without the use of EO.

In this chapter I showed how individuals may break the problem constraints and become invalid through simple crossover or mutation. I described Template Repair and showed how this can be used as a template to repair invalid individuals. I went on to introduce GeneRepair which uses the individual's ancestor as the repair template. I showed how the parent, grandparent or great-grandparent can be used to repair an invalid individual in the population. I described how GeneRepair operates in two distinct phases: *error detection* followed by ancestor driven *error correction*. I explained the GeneRepair parameters of ancestor, generation, repair direction and fitness and illustrated how each of these can dictate the properties of the repaired individual.

In the next chapter I will conduct experiments to investigate GeneRepair

and the influence of each of its parameters. The goal when deciding on all of these properties is to try to find the optimal solution by using GeneRepair in conjunction with an EO. In order to do this diversity must be maintained across the population. If diversity is not maintained the EO will quickly reach a local maxima but will never reach the global optimal. The reason for this is that all individuals in the population will tend towards the best individual. Soon the population will consist of duplicates of the best individual. By maintaining diversity across the population the individuals will evolve in different directions allowing the EO to reach a global optimal rather than local maxima.

For the first collection of experiments the Travelling Salesman Problem will be used as a sample problem. While this technique of constraint handling is not the most suited technique to use when solving the TSP, I have chosen the TSP as it clearly illustrates the use of repair in the experiments and is widely known and accepted as a sample constraint based problem. As it is a *de facto* standard constraint satisfaction problem, it also affords us the possibility to explore the effectiveness of Lolles hypothesised repair strategy on a standard problem domain. Experimental results (See Chapter 5) may help shed some light on the likely efficiency of this controversial non-Mendelian inheritance theory.

Each of the experiments conducted will illustrate the effect of the different parameters on the individual repaired by GeneRepair. Using the results produced by these experiments suggestions will be proposed on the optimal parameters to use. In order to support these experimental results I will use

a number of different datasets within the TSP. I will also show how problem size affects GeneRepair and make suggestions based on this trend for the possible result of even larger untested problems.

Chapter 5

Testing the Theory

5.1 Introduction

There are two primary objectives of this thesis. The first concerns the field of computer science. This objective is to investigate whether non-Mendelian template repair can be used to enable EO to produce valid solutions to constraint based problems. I will investigate this non-Mendelian template repair thoroughly and compare it to the alternative Mendelian repair where the parent is used as the repair template. The results produced when achieving this objective are illustrated in Section 5.5. The second objective of this thesis is to find out if the computer science side of the EO analogy can support or undermine the findings of Lolle *et al* that suggest that non-Mendelian repair can be used to repair individuals in the current population. The results produced when achieving this objective are explored throughout these results but are the central focus in Section 5.6. The use of GeneRepair in conjunc-

tion with EO was investigated thoroughly and the results are illustrated in this chapter.

5.1.1 Structure of Chapter

This chapter begins by explaining the problem set. It then goes on to illustrate the experimental setup for all experiments described in this chapter. The results are divided in two in terms of the two objectives described above. The first section begins by establishing a base line for the results. This is done by comparing the death penalty to parent template GeneRepair (PTR). We go on to see PTR compared to its non-Mendelian counterparts. The effect of mutation rate on the repair strategy is then investigated followed by the effect of the templates fitness. We then go on to compare different population sizes and also examine how GeneRepair behaves during early evolution and at various generational milestones. Following this we introduce Random Template Repair and compare this to the proposed GeneRepair strategy. We examine the effect of problem size on ancestral repair template efficiency. We will investigate the impact of the direction of repair on the repair strategy and finish by showing how the storage of ancestors is conducted by the repair strategy.

The second objective of this thesis is to investigate whether the CS side of the EO analogy can support or undermine the findings of Lolle *et al.* In this set of results we will look at the effect of reduced redundancy representation on the choice of ancestral repair template. We will also investigate the use of self-crossover as well as low mutation rates. The results in this second section

aim to use biologically inspired parameters to explore the proposed repair strategy, however (as explained in Section 1.5) what we term as "biological experiments"are still computational evaluations and should not be confused with systems biology or related disciplines.

5.2 Explanation of the Problem

The Travelling Salesman Problem (TSP), as the name suggests, originates from the optimisation problem facing a travelling salesman who has a number of cities to visit and wants to visit each one exactly once and return to the first city using the shortest possible route. The salesman wants to cover as little distance as possible but still visit each city once and once only returning to the first city at the end. The distance travelled is called the tour length and the order of the cities visited is called the tour.

Previously, in Figure 4.3 I illustrated a sample 6 city TSP. In Figure 4.4 a possible solution to this TSP is illustrated while in Figure 4.5 an optimal solution is shown. This 6 city tour has 6! possible solutions as it has been previously proved that a

$$Number of Possible Solutions(n\, City\, TSP) = n! \tag{5.1}$$

possible solutions. This 6 City problem therefore has 720 possible solutions. This is a small problem used for illustrative purposes only. In our experiments we will be using the 51 City (eil51), 101 City (eil101), 532 City (att532) and the 18512 City (d18512) TSPs. For the smallest of these prob-

lems, the 51 city problem, there are

$$1.55111875 \times 10^{66} \tag{5.2}$$

possible solutions.

In this thesis the use of GeneRepair is investigated and the effect of a variety of impacting parameters are examined. In order to carry out these experiments a standard travelling salesman problem dataset was used. GeneRepair is not necessarily the best technique for TSP (Lin & Kernighan 1973) but GeneRepair can in principle be applied to many constraint problems. As mentioned previously, the reason that the TSP is used in this thesis is that it is a standard example of a constraint based problem thus there is no confusion about the problem and all focus is on the GeneRepair technique. There is also a wide and varying number of datasets that can be used.

5.2.1 Datasets

During the investigation of GeneRepair I will conduct experiments using the TSP library (Reinalt 1991). From this library I will use a number of different datasets to guarantee integrity. The use of different datasets will also eliminate the possibility that results depend on accidental artefacts of a particular problem and allow me to investigate effect related to problem size. In Chapter 6 we will look at the effectiveness of genetic repair on some of the CVRP datasets from TSPLIB repository (Reinalt 1991).

5.3 Experimental Setup

The previously described model of Genetic Repair was implemented in a Java model in order to test its efficiency. As the biological origins are of central significance to this work, many experiments investigate the impact of biological factors as much as factors of pure computational significance.

For the experiments in this chapter the problem used is the TSP so the fitness is calculated as the tour length of the individual. This fitness value is calculated by using the Euclidean Distance Formula between each of the cities in the individual including the distance from the last city back to the first city. The experiments are repeated for a number of times under identical conditions to ensure that the results are reproducible and reliable. In order to obtain reliable results that could be statistically analysed each experiment was repeated for a number of iterations.

5.3.1 Repetition of Experiments

In the implementation the experiments were labelled from a to z and so the number of iterations was either 26 or occasionally 52. Truncation selection is the selection method used throughout this chapter. The experimental parameters for each experiment in this Chapter are listed in Table 5.1.

5.3.2 Population Size

As shown in Table 5.1 population sizes of 4, 10, 50 and 100 were used. In order to investigate the effect of population size on the results produced I

Table 5.1: Overview of Results Presented

Figure	TSP Size	Population	Mutation Rate (%)	Generations	Iterations
5.1	101	100	0.5, 2, 5 and 10	500k	26
5.2	101	50	2 and 10	500k	26
5.3	532	50	2	500k	26
5.4	18512	10	0.001	500k	16
5.5	101	50	2	500k	26
5.6	51	50	2	500k	52
5.7	76	50	2	500k	26
5.8	532	50	2	500k	26
5.9	18512	10	0.001	500k	16
5.10	51	50	2	500k	26
5.11	76	50	2	500k	26
5.12	51	50	2	500k	52
5.13	101	50	2, 1.75, 1.5, 1.25, 1, 0.75, 0.5, 0.25 and 0.1	500k	52
5.14	51	50	2, 1.75, 1.5, 1.25, 1, 0.75, 0.5, 0.25 and 0.1	500k	52
5.15	101	50	2	500k	26
5.16	101	50	2	500k	26
5.17	101	50	2	500k	26
5.18	101	50	2	500k	26
5.19	101	100	2	500k	26
5.20	101	4	2	500k	26
5.21	101	100	2	500k	26
5.22	18512	10	0.01	50k	26
5.23	76	100	2, 1.75, 1.5, 1.25, 1, 0.75, 0.5, 0.25 and 0.1	500k	26
5.24	532	50	2	500k	26
5.25	51	100	2	500k	52
5.26	51	100	2	500k	52
5.27	51	100	2	500k	52
5.28	51	100	1.75	500k	52
5.29	101	50	2	500k	26
5.30	101	50	2	500k	26
5.31	101	50	2	500k	26
5.33	101	50	2	500k	26
5.34	101	50	2	500k	26
5.35	101	50	2	500k	26
5.36	1379	100	0.1	10k	26
5.37	1379	100	0.1	10k	26
5.38	101	50	2	500k	26
5.39	101	50	2	500k	26
5.40	101	50	2	500k	26
5.41	101	50	2	500k	26
5.42	101	50	2	500k	50
5.43	101	50	2	500k	50

amended the population size. While it may be argued that 50 and 100 are too close to similar to show the effect of population size on the results I argue that by doubling the population size (from 50 to 100) this is a drastic change to the parameter. Using a population of 10 in conjunction with the 18512 city TSP attempts to reproduce a more "biologically"plausible parameter. The *A. thaliana* is predominantly a population of clones with little diversity so this experiment with a small population of 10 and a large problem size (18512 cities) attempts to simulate this biological environment in the EO.

5.3.3 Mutation Rate

The mutation rate represents the % of change that is inflicted on the population during each generation. The rate is

$$MutationRate\% * (populationsize * numberofCities) \tag{5.3}$$

Therefore if the mutation rate is 2.0%, the population is 50 and there are 51 Cities the number of Cities mutated is equal to:

$$0.02 * (50 * 51) = 51 \tag{5.4}$$

In the experiments described in this thesis swap mutation was used which means that for the above equation 51 swap mutations would be carried out which equates to 102 alleles (51 identified by the swap mutation technique and the 51 alleles that these are swapped with) in the entire population being affected by the mutation. Using this mutation technique the city selected for

TSP Dataset	Known Optimal
eil 51	426
eil 76	538
eil 101	629
att 532	27686
nrw 1379	56638
d18512	[645092,645300]

Table 5.2: TSP datasets and their optimal tour length or interval of upper and lower bound

mutation is swapped randomly with another city (also known as reciprocal exchange mutation).

5.3.4 Solutions Produced by Alternative Methods

In this thesis I compare the use of different ancestral templates as repair templates to use with GeneRepair. The results are compared to each other as opposed to benchmark results produced by other EA. For reference, Table 5.2 shows the TSPs used in this Chapter along with their known optimal tour length (or interval in the case of d18512 TSP). This shows the performance of other EA on these problems. This thesis compares ancestral repair templates to each other and to the use of the death penalty as opposed to comparing them to the performance of other EA.

5.3.5 Computational Effort

In order to multi-task and save time some of the experiments presented were run on the The Irish Centre for High-End Computing (ICHEC) and CREIG (which is the NUI Maynooth cluster) [1] servers. The majority of the experiments, however, were run on a standard PC as they did not have any special computational requirements. One of the merits of the GeneRepair method is the low computational running cost. For example a 500,000 generation eil101 TSP with a population of 100 and great-grandparent template GeneRepair took less than six minutes to run on a standard unmodified PC running Windows XP. The experiments which took the longest time were those using the d18512 city TSP dataset but with a population of 10 and 10,000 generations each iteration of this experiment took less than 45 minutes to complete.

5.4 Presentation of Results

To illustrate the results produced by the experiments I have used boxplots, graphs and tables, accompanied by statistics as appropriate. For the majority of the results presented I chose to use the "quartile graph" (boxplot) which details the minimum, maximum (max/min.avg). Where graphs are used in this thesis, the results displayed in each graph have been pre-sorted from smallest to largest for convenience and ease of comparison. The statistical analysis tools that I have used are the Kruskal-Wallis test and the Mann-

[1] This is the NUI Maynooth cluster. More information available at available at http://creig.cs.nuim.ie/wordpress/

Whitney U test.

The Kruskal-Wallis test is a nonparametric test that compares three or more unpaired or unmatched groups. This one-way analysis of variance by ranks is a non-parametric method for testing whether samples originate from the same distribution. When the Kruskal-Wallis test leads to significant results this shows that at least one of the samples is different from the other samples. The test does not identify where the differences occur or how many differences actually occur. It is an extension of the MannWhitney U test to 3 or more groups. The Mann-Whitney U test can provide more information as it analyses the specific sample pairs for significant differences.

The MannWhitney U test (also called the MannWhitneyWilcoxon (MWW) or Wilcoxon rank-sum test) is a non-parametric statistical hypothesis test for assessing whether two independent samples of observations have equally large values. It is one of the most well-known non-parametric significance tests. For presented statistics the sample size (n) and the p value are provided. This statistical analysis method first ranks all of the values from low to high. If two values are the same, then they both get the average of the two ranks for which they tie. The smallest number gets a rank of 1. The largest number gets a rank of N, where N is the total number of values in the two groups. The ranks are then summed in each group, and the two sums are reported. If the sums of the ranks are very different, the P value will be small. If the P value is small, you can reject the idea that the difference is a coincidence, and conclude instead that the populations have different medians.

The TSP is a minimisation problem so the lower the result in the ta-

ble/graph the better the result or the lower the tour length the stronger the result. In this chapter the boxplots/graphs shown have tour length on the Y-axis. The X-axis is labelled and this generally represents the type of ancestral GeneRepair carried out. Each graph contains a legend describing which repair methods are being compared.

5.5 Objective 1 - Computationally Focused Investigation of GeneRepair

As stated in Chapter 1 the first objective of this thesis is to investigate whether non-Mendelian template repair can be used to enable EO to produce valid solutions to constraint based problems. In this Section I will compare the use of non-Mendelian template repair to the use of Mendelian template repair in a number of various situations using different parameters and problem sets. But before moving onto this central issue of non-Mendelian repair, we shall first explore a version of GeneRepair that uses a Mendelian (ie parent based) repair template. Our objective is to establish a baseline for comparison, by comparing Mendelian repair to another biologically inspired constraint handling method of the Death Penalty (as described in Section 2.4.4).

5.5.1 Death Penalty

The Death Penalty is a method that can be used to enable EO to solve constraint based problems. As the name suggests this method "kills"or *elim-*

inates invalid individuals from the population. This constraint enforcing mechanism is part of the family of mechanisms known as Penalties (See Section 2.4.4). In this Section the death penalty technique is used as an initial benchmark to illustrate one technique EO may use to solve constraint based problems. In the next Section I will compare this technique to GeneRepair using the parent of the individual as the repair template. To investigate the effect of death penalty as a constraint handling mechanism an EO was run on the 101 City TSP (eil101) for 500,000 generations and mutations of 0.5%, 2%, 5% and 10% on a population of 100. This mutation rate choice will be explained thoroughly in Section 5.5.5.

In Figure 5.1 we can see the results produced by this experiment. The known optimal result for this problem set is a tour length of 629. Figure 5.1 shows that none of the mutation rates enable the death penalty technique to produce results close to the optimal of 629 with the best result produced having a tour length of over 3,000. This Figure also shows that the mutation rates of 10% and 2% produced the strongest results. It was expected that the 10% mutation would perform well as death penalty drastically reduces diversity and high mutation brings back some of that lost diversity. Mutation at 2% may have produced strong results as it introduces just enough diversity to keep the algorithm evolving as opposed to 0.5% which may not introduce enough diversity.

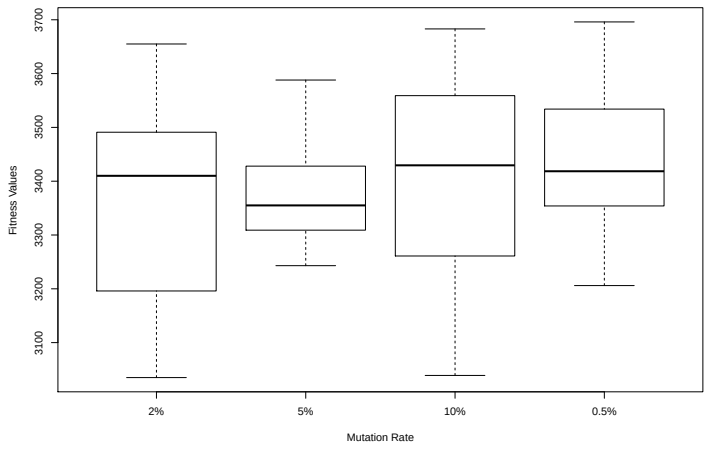

Figure 5.1: Death Penalty Results

5.5.2 Parent Template Repair

Thus far we have seen the performance of the death penalty to enable EO to solve a standard constraint based problem. This technique shall now be used as a benchmark to measure the efficiency of GeneRepair using the parent as a repair template to produce valid solutions to the same constraint based problem.

Hypothesis: *Parent Driven GeneRepair provides a more efficient technique to enable EO to handle constraints than Death Penalty approaches*
As described in Section 4.5 Parent Template Repair(abbreviated to PTR) is the use of the parent as a repair template in the GeneRepair mechanism. This section compares the use of Death Penalty with the use of Parent Template GeneRepair on a number of different TSPs. The population is set to 50. The experiment is run for a standard of 500,000 generations. The mutation

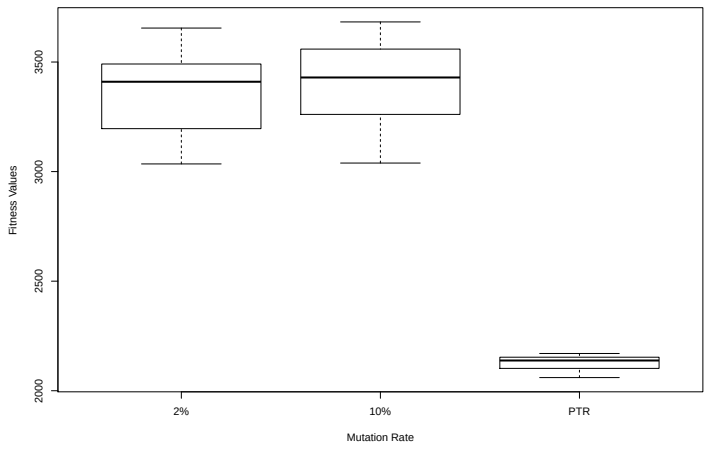

Figure 5.2: Death Penalty vs Parent Template Repair for 101 City TSP

rate has been set to 2.0% and the problem is the 101 City TSP (eil101).

Figure 5.2 shows that the parent template can be used by GeneRepair to enable EO to produce valid solutions to a standard constraint based problem. As shown in Figure 5.2 the GeneRepair technique using a parent repair template greatly outperforms the death penalty at the best mutation rate found in Section 5.5.1 when the results of the three (Death Penalty at two mutation rates and PTR) experiment sets are compared. Using all 26 repetitions of this experiment, Kruskal-Wallis analysis showed that there was a significant different among the three groups (PTR, DPTR at 2% and DPTR at 10% mutation) with $H = 51.71$, $df = 2$ and $p < 0.0001$. Mann Whitney statistical analysis found that PTR <DP with ($p < 0.0001$). This provides support for the hypothesis that parent template GeneRepair provides a more efficient technique to enable EO to handle constraints than the death penalty for the given conditions. This result is in agreement with previously published

work (FitzGerald & O'Donoghue 2008). Many similar results were produced showing weakness of death penalty and so further results are not included for this reason.

5.5.3 Non-Mendelian Template Repair

Thus far in this thesis we have seen that GeneRepair can successfully use the parent as a repair template to enable EO to produce valid solutions to a standard constraint based problem. We have also seen that this method produces superior results to the death penalty approach. In this Section we will investigate whether non-Mendelian repair templates are successful at enabling GeneRepair to produce valid solutions to the same constraint based problem and if they produce superior results to that of the parent template.

Hypothesis: *Non-Mendelian repair templates are more effective than Mendelian Repair Template*

In Section 4.6 the use of the Grandparent as a repair template was explained and this was shown to be the central contentious claim of Lolle *et al*(Lolle *et al.* 2005). This repair mechanism is tested in this section for a standard of 500,000 generations with a population of 50 and a mutation rate of 2.0% using the larger 532 City TSP. For reasons that shall soon become clear we will not commence by looking at the 101 city problem, but shall start by looking a the 532 city problem instead. In this Section Grandparent Repair Template GeneRepair (GPTR) is compared to the use of Parent Repair Template GeneRepair (PTR) which was previously described in Section 5.5.2. The results produced by this experiment are illustrated in Figure 5.3.

This illustrates that the grandparent was more effective as a repair template than the parent as it produced results with a shorter tour length. There is a clear difference between the two lines showing that across 26 separate experiments the grandparent and parent produced reliably different results with the grandparent repair template producing better results. This experiment was run 52 times, as opposed to 26, to clearly show the results produced by running the experiment 26 times are as reliable as those produced when it is run 52 times. Using the Mann Whitney statistical analysis with a sample size 52 it was shown that GPTR <PTR with ($p < 0.0001$). This strong probability shows that the difference between the results is more than coincidence and we can confidently say that the grandparent repair template used with GeneRepair produced stronger results than the parent template.

This is a surprising and startlingly strong finding that may have implications for researchers of Mendelian and non-Mendelian inheritance in the Biological Research Community [1]. It also represents the first piece of supporting evidence (albeit highly indirect) for Lolle's non-Mendelian inheritance theory. While Figure 5.3 illustrates that the use of parent and grandparent as repair templates produce different results with grandparent outperforming parent, neither of these result sets are close to the known optimal for this solution which is 27,686 as convergence was still proceeding and I am confident that later in this thesis we will see instance where GeneRepair gets extremely close to the known optimal for a problem. In this Section the hypothesis that non-Mendelian repair templates can prove more effective than their Mendelian

[1]Personal communication with Dr Susan Lolle in University of Waterloo, Ontario, Canada. Biological considerations will be revisited later in this thesis.

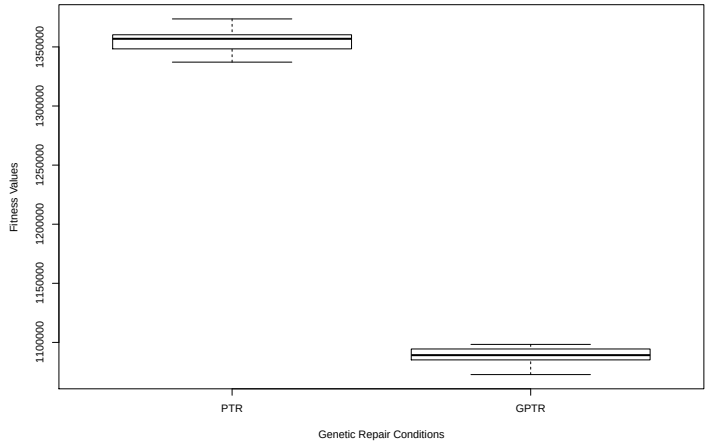

Figure 5.3: Parent vs Grandparent Results - 532 City TSP

counterparts was supported(FitzGerald & O'Donoghue 2008).

While the statistical analysis for this set of results (See Figure 5.3) shows that the grandparent repair template outperforms the parent repair template. I will now strengthen this result by showing the findings again show the superiority of the non-Mendelian template repair when the 18512 City TSP is used. In Figure 5.4 the results are shown when the parent template GeneRepair is compared to the Grandparent template GeneRepair for the 18512 City TSP with 0.001% mutation and a population of 10. These new parameters have been used to reduce the computation time for such a large problem. High mutation of 2% is not necessary for such a large problem size as the diversity will be maintained at lower mutation due to the length of the individual (number of cities). The direction of repair was a random and varying direction which will be discussed in Section 5.5.12. We can see from Figure 5.4 that once again the grandparent template outperforms the

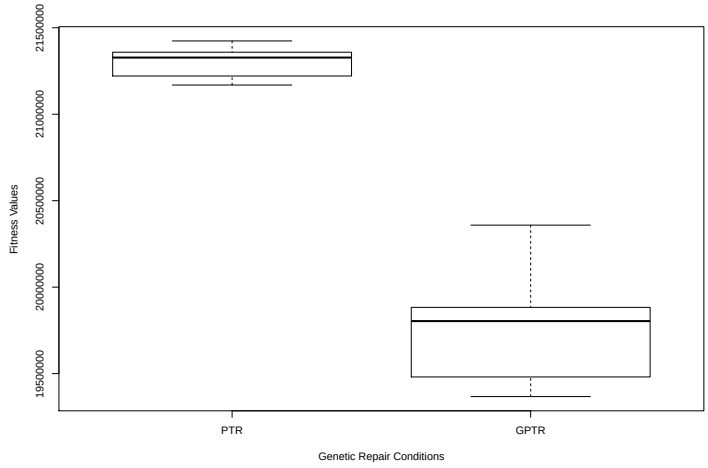

Figure 5.4: Parent vs Grandparent Results - 18512 City TSP

parent template when used with GeneRepair. Comparing the use of the parent and grandparent repair templates on the 532 and 18512 City problems we have seen that the grandparent outperforms the parent when used with GeneRepair to enforce the problem constraints.

Now I will examine a smaller problem which is the 101 City TSP. In Figure 5.5 we can see that on this smaller problem size surprisingly the parent template outperforms the grandparent template. We see the same superiority of parent template when we compare the use of parent template and grandparent template GeneRepair on the 51 and 76 City TSP (See Figures 5.6 and 5.7) problems. Thus far we have seen that grandparent template GeneRepair outperforms parent template GeneRepair for the (large) 532 and 18512 City problems but when the problem size is reduced to 51, 76 and 101 Cities the parent outperforms the grandparent repair template. In Section 5.5.4 I shall explain this seemingly contradictory result.

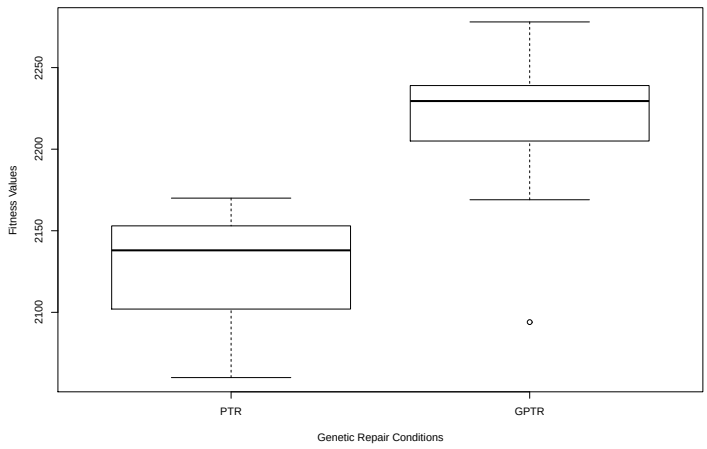

Figure 5.5: Parent vs Grandparent Results 101 City TSP

5.5.4 Great-Grandparent Template Repair

Following from this finding I will now examine the great-grandparent repair template to examine whether this outperforms the grandparent template for larger (18512 and 532 City) and smaller (101, 76 and 51 City) problems.

In Section 4.6 the use of Great-Grandparent Template in conjunction with GeneRepair was discussed. This Great-grandparent repair (GGPTR) is investigated in this section. As with the experiments described previously in this chapter (See Section 5.5.2 and 5.5.3) the population is set to 50 and the experiment is run for 500,000 generations with a mutation rate of 2.0% (as was used in Section 5.5.2 and 5.5.3).

In Figure 5.8 we can see that the great-grandparent repair template outperforms the parent template but fails to outperform the grandparent template. In this set of results both non-Mendelian templates yet again out-

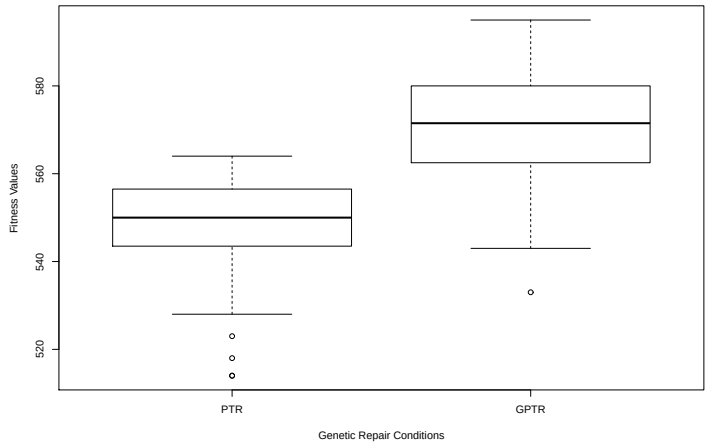

Figure 5.6: Parent vs Grandparent Results - 51 City TSP

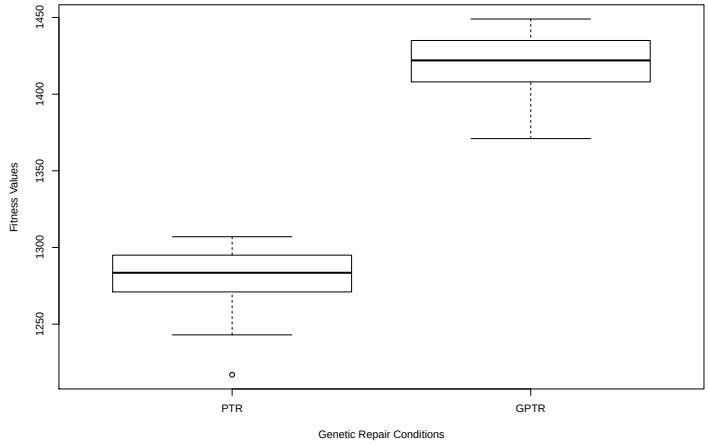

Figure 5.7: Parent vs Grandparent Results - 76 City TSP

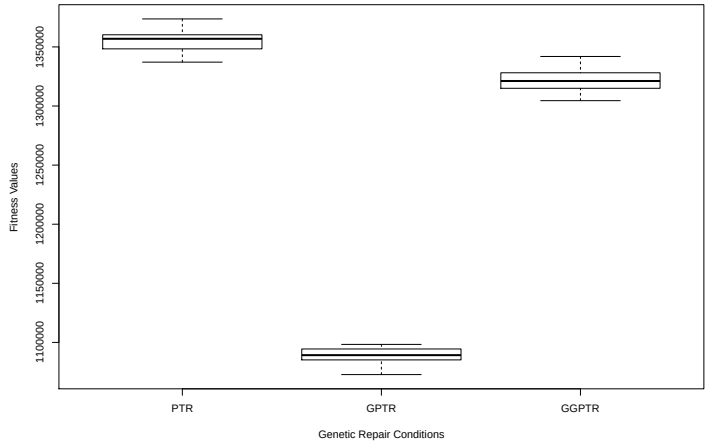

Figure 5.8: Parent vs Grandparent vs Great Grand Parent Results - 532 City TSP

perform their Mendelian counterparts. Kruskal-Wallis analysis showed that there was a significant different among the three groups (PTR, GPTR and GGPTR) with H = 68.25, df = 2 and p <0.0001. The Mann Whitney statistical analysis for this set of results showed that GPTR <GGPTR with a (p <0.0001) and GGPTR <PTR with (p <0.0001).

In Figure 5.9 we can see that the great-grandparent repair template outperforms the parent template but fails to outperform the grandparent template which is the same as when the other large problem (532 City TSP) was investigated. This again supports the hypothesis that non-Mendelian ancestral repair templates outperform their Mendelian counterparts. Kruskal-Wallis analysis on this set of showed a significant difference in the three groups with H = 36.15, df = 2 and p <0.0001.

In Figure 5.10 we can see the results produced when the 101 City TSP was

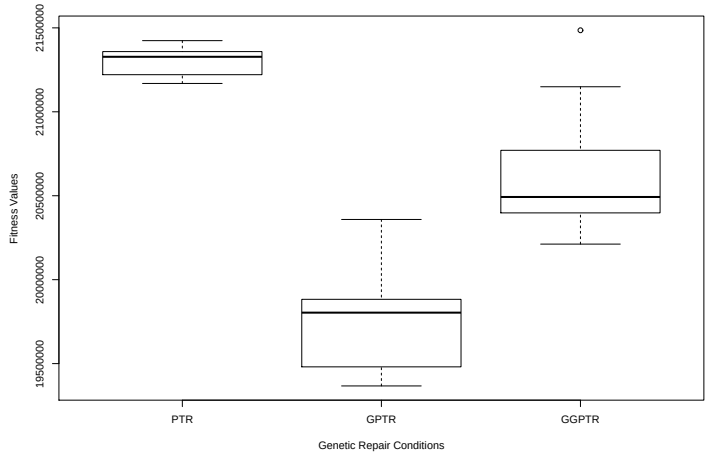

Figure 5.9: Parent vs Grandparent vs Great Grand Parent Results - 18512 City TSP

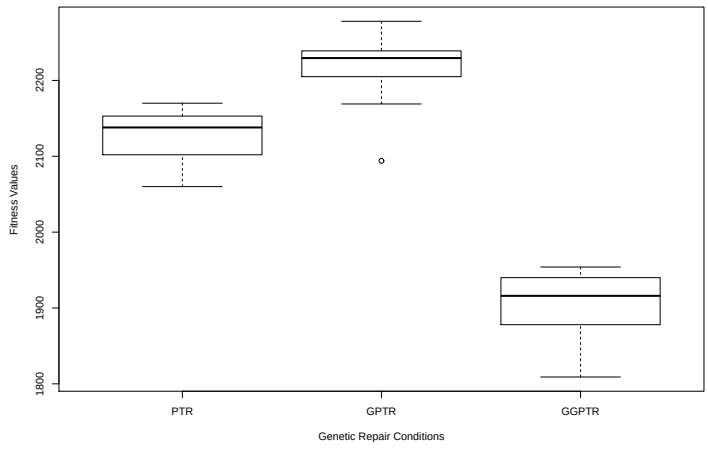

Figure 5.10: Parent vs Grandparent vs Great Grand Parent Results for 101 City TSP

used to compare parent, grandparent and great-grandparent repair templates with GeneRepair. In Figure 5.10 we can see that the use of the great grandparent as a repair template is more effective than using either the parent or the grandparent. This again supports the hypothesis that non-Mendelian repair templates can prove more effective than their Mendelian counterparts. We must still remember that the known optimal for this problem is a tour length of 629 so this Section suggests the effectiveness of the repair templates in comparison to each other as opposed to other possible techniques (FitzGerald, O'Donoghue & Liu 2009). With greatly increasing the number of generations we are confident that GeneRepair will generate more competitive solutions and it should also be noted that we have not used adaptive mutation, elitism or other techniques to improve the results obtained in this chapter. Kruskal-Wallis analysis on this set of results showed a significant difference in the three sets of results with $H = 66.25$, $df = 2$ and $p < 0.0001$. If we look at the Mann Whitney statistics for these results we find that GGPTR <GPTR with ($p < 0.0001$) and GGPTR <PTR with ($p < 0.0001$). We can therefore confidently state that non-Mendelian repair templates outperform the Mendelian template GeneRepair.

Thus, on the larger problems, both non-Mendelian repair templates yield results that outperform the Mendelian alternative. We now re-visit the results produced on the smaller problems that appear to contradict this finding. We saw in Figure 5.6 and Figure 5.7 that for the smaller 51 and 76 City problems parent template GeneRepair outperformed grandparent template repair apparently contradicting the results suggesting that superior results

are produced using the non-Mendelian templates. These results are examined further by comparing them to great-grandparent template repair. In Figures 5.12 and 5.11 we can see that while parent template GeneRepair outperforms grandparent template GeneRepair, great-grandparent template GeneRepair outperforms both of these options with a confidence level of ($p<0.0001$). Kruskal-Wallis analysis showed significant difference in the sets of results produced with $H = 68.46$, $df = 2$ and $p < 0.0001$ for the data shown in Figure 5.11 and $H = 111.93$, $df = 2$ and $p < 0.0001$ for Figure 5.12.

This supports the hypothesis that non-Mendelian repair templates are more effective than Mendelian repair. For smaller problems the great-grandparent template seems to outperform the parent template while for larger problems both the grandparent template and the great-grandparent template outperform their Mendelian counterpart. We can hypothesise that this effectiveness of repair template may be linked to problem size. The problem size dictates the size of the individual. Using an older ancestral template may produce more diversity than a closer (in ancestral terms) template as it would be more diverse to the individual being repaired. This high level of diversity may be advantageous for the smaller individual which achieved a greater degree of convergence in the given 500,000 generations. In contrast we expect the slightly less diverse template (grandparent) may suit the larger individual which would have a slower convergence. It should also be pointed out that the larger problems were further from their known optimal solutions when these experiments were terminated (at 500,000 generations). This of course was also related to the populations diversity. (For an example of GeneRepair

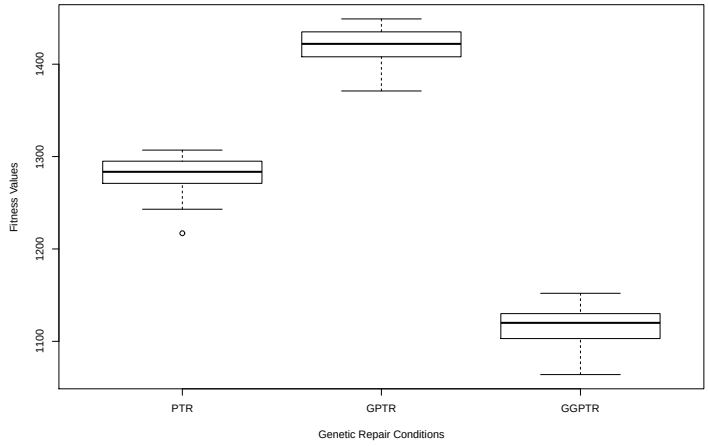

Figure 5.11: Parent vs Grandparent vs Great Grand Parent Results - 76 City TSP

producing near optimal solutions see Section 5.5.10). In this section it has been shown that when the three ancestral repair templates were compared one of the non-Mendelian templates always outperformed the PTR. This is a central finding to this body of research as it suggests that the storage of non-Mendelian ancestral templates to be used with the GeneRepair strategy provides a superior repair strategy to that using Mendelian repair templates.

5.5.5 Mutation Rate

In the experiments carried out so far in this chapter a standard mutation rate of 2.0% has been used throughout with the exception of the 18512 City TSP experiment where a lower mutation was used. In this Section a wide variety of different mutation rates are investigated to examine the efficiency

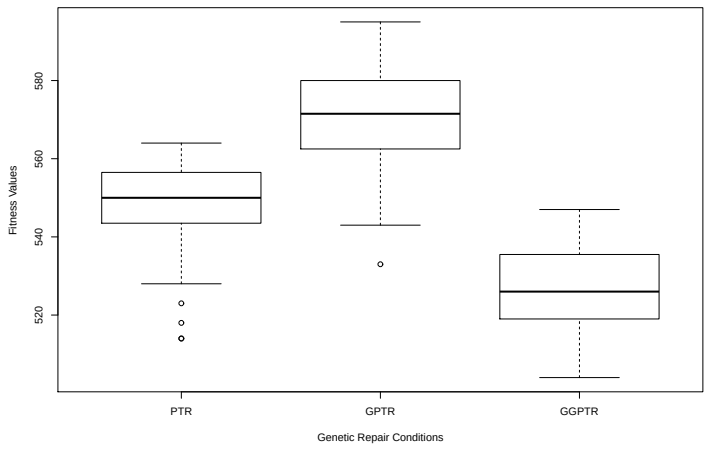

Figure 5.12: Parent vs Grandparent vs Great Grand Parent Results - 51 City TSP

of Mendelian and non-Mendelian repair templates at different mutation rates.

Hypothesis: *The choice of template to use with GeneRepair template does not affect the optimal mutation rate for the specific problem*

In Section 5.5.6 I investigated the effect of the fitness of the repair template used. I found that choosing a random template as opposed to the fittest or least fit available produced superior results. In this section I will incorporate that finding by choosing a random repair template from the two available as opposed to the fittest or least fit. Repair is carried out in a random and varying direction (See Section 5.5.12). In Figure 5.13 the results of the experiment illustrate that regardless of the repair template used the same optimal (of those compared) mutation rate is found at 0.75%. Each mutation rate was run for 52 experiments to ensure integrity of results. When Mann

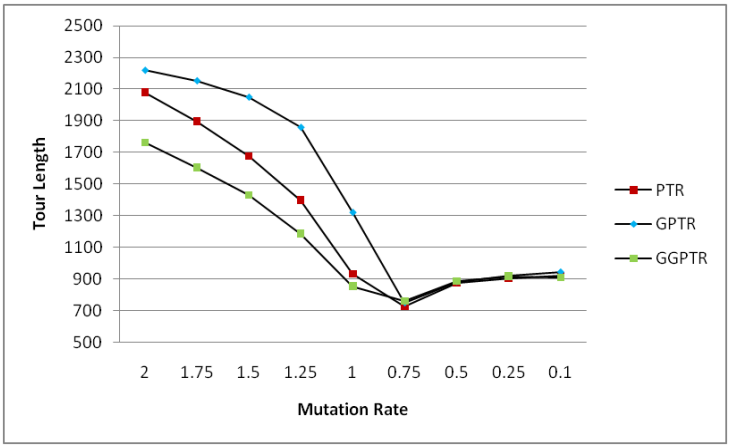

Figure 5.13: Comparing the Efficiency of Ancestral Repair Templates Across a Variety of Mutation Rates using the 101 City TSP

Whitney statistical analysis was carried out on these results, with all 52 samples included, the mutation rate of 0.75% was found to be most efficient when great-grandparents of each mutation rate were compared with ($p < 0.0001$). To carry out the analysis the great-grandparent results of each mutation rate were compared to those at 0.75%. The reason great-grandparent was used for the analysis was that it seemed to produce the best overall result, especially at high mutation rates. We can also note from this Figure that at very low mutation rates the particular repair template has a smaller effect on the results making it apparently less clear which ancestral repair template is most efficient. This property will be investigated later in Section 5.6.3. Figure 5.13 supports the hypothesis that the choice of repair template does not affect the optimal mutation rate for the specific problem (FitzGerald & O'Donoghue 2010).

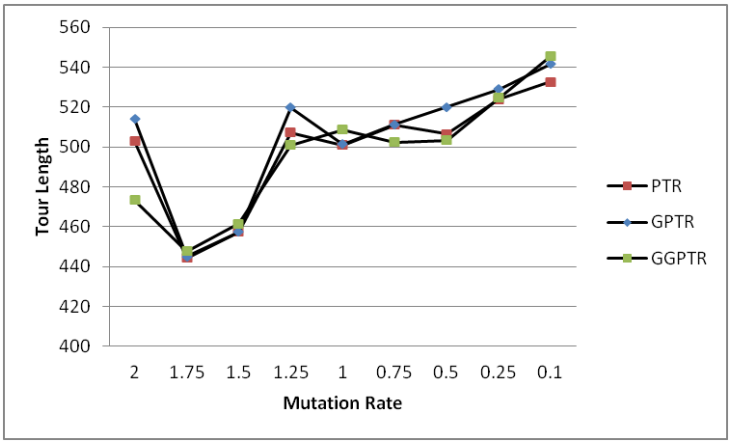

Figure 5.14: 51 City TSP Effect of Repair Template on Mutation Rate Effectiveness

In Figure 5.13 the 101 City TSP was used and the optimal mutation rate for all three of the ancestral repair templates was 0.75%. Figure 5.13 and 5.14 show the average tour length across the experiment iterations at each mutation rate. Figure 5.14 shows the effect of varying the mutation rate across the three different ancestral repair templates as in Section 5.5.5 except that the 51 City TSP is used in Figure 5.14. The optimal mutation rate was found at the same value, 1.75%, for each of the three templates . As you can see in Figure 5.14 the repair template used does not affect which mutation rate produces the best results.

5.5.6 Fitness

Thus far in this thesis we have seen how the parent can be used by GeneRepair as a repair template to enable EO to produce valid solutions to a stan-

dard constraint based problem in a more efficient manner than the death penalty approach. We went on to investigate the use on non-Mendelian repair templates (the grandparent and great-grandparent) and found that non-Mendelian repair templates outperform the Mendelian parent template. As these experiments were for a standard mutation rate of 2% an investigation into the behaviour of GeneRepair at different mutation rates was carried out and it was found that regardless of the template used, the optimal mutation rate found remains the same. To further analyse the efficiency of GeneRepair using Mendelian and non-Mendelian repair templates we will now examine the choice of specific template after the ancestor is chosen by using fitness as a parameter.

Hypothesis: *Non-Mendelian Repair templates outperform their Mendelian counterpart regardless of the fitness of the chosen template*
Fitness is a calculation of the quality of the individual in the EO (See Section 4.3). As previously explained the TSP is the problem set used for these experiments so the fitness refers to the tour length of each individual and as this is a minimisation problem, the lower the fitness - the stronger the result.

When choosing a repair template there are three variations of fitness to choose from. Therefore the choice of repair template in the experiments described in Section 5.5.2 was between two parents while the choice of repair template in the experiments described in Section 5.5.3 was between the two previously stored grandparents. For each of these choices an associated property with each of the repair templates is the fitness value. In this Section I will investigate the impact of using template fitness as a factor in selecting

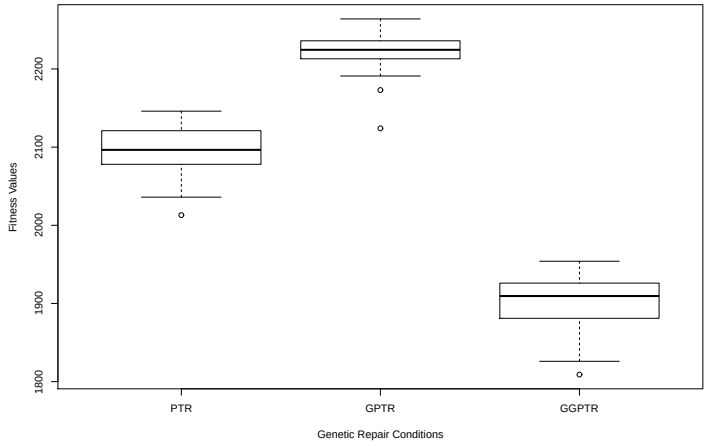

Figure 5.15: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair Using the **Fittest** Template

which ancestor to store as the ancestral template. I will compare the three template selection strategies: the fittest template and the least fit template (of the two ancestors) to a random chosen template (of the two ancestors) and assess how this affects the repair template efficiency. The population will be set to 50 and the experiments are run for 500,000 generations with a mutation rate of 2.0% using the 101 City TSP.

In Figures 5.15, 5.16 and 5.17 we can see the effect of choosing the fittest, least fit and randomly chosen ancestral template (respectively) on the effectiveness of each ancestral repair template. In each of these Figures it is clear to see that the order of effectiveness of the ancestral repair templates is identical with the great-grandparent repair template producing the best results (lowest tour length). The parent repair template is second in the order of results with the great-grandparent repair template producing the worst

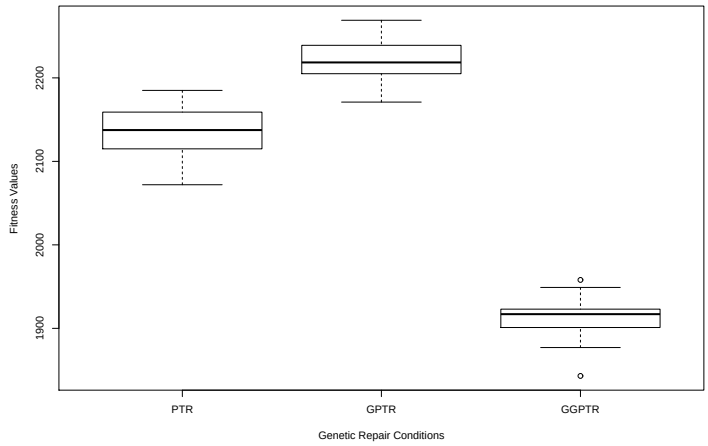

Figure 5.16: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair Using the **Least Fit** Template

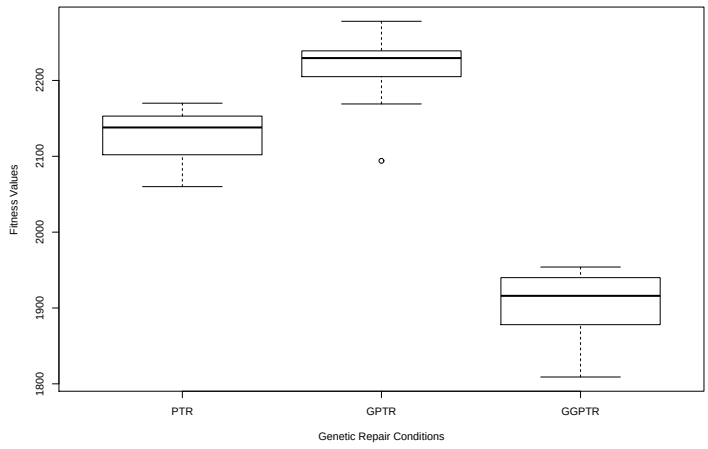

Figure 5.17: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair Using a **Randomly Chosen** Template

Minimum			
Ancestor	Fittest	Least Fit	Randomly Chosen
PTR	2013	2072	2060
GPTR	2124	2171	2094
GGPTR	1809	1843	1809

Table 5.3: Comparing the use of the Fittest Template, Least Fit Template and Randomly Chosen Template - Minimum Tour Length Obtained

results for each of the template fitness options compared. Figures 5.15, 5.16 and 5.17 also illustrate that the fitness of the template chosen from the determined ancestry (parent, grandparent or great-grandparent) does not affect the choice of ancestor to use. Therefore when choosing an ancestral repair template for the 101 City TSP the grandparent should be chosen regardless of which fitness parameter (fittest, least fit or random) is used. Kruskal-Wallis analysis for the results shown in Figure 5.15 showed a significant difference in the three sets of results with $H = 68$, $df = 2$ and $p < 0.0001$.The same data was produced by Kruskal-Wallis analysis for the results shown in Figure 5.16 while the results in Figure 5.17 produced $H = 66.25$, $df = 2$ and $p < 0.0001$.

Returning to the hypothesis posed at the beginning of this section, in Table 5.3 and Table 5.4 the fittest repair template is compared to both the least fit template and the randomly chosen repair template (from the specific ancestral level). (Note: Randomly chosen repair template of two parents or grandparents or great-grandparents of the individual as opposed to a random template chosen from the population. Randomly chosen template is where we

Average			
Ancestor	Fittest Template	Least Fit Template	Randomly Chosen Template
PTR	2096	2134	2129
GPTR	2221	2221	2219
GGPTR	1901	1913	1909

Table 5.4: Comparing the use of the Fittest Template, Least Fit Template and Randomly Chosen Template - Average Tour Length Obtained

are (randomly) selecting an ancestor to store as a repair template irrespective of its fitness value. In contrast a random template is a template formed directly by a random number generator as discussed in Section 5.5.9). Table 5.3 shows the minimum result obtained by the experiments while Table 5.4 shows the average result across the 26 repetitions of the experiment.

Looking at the strongest results produced, as illustrated in Table 5.3, we can see that the lowest tour length produced was 1809 and this was produced in separate experiments where the fittest parent repair template was used and where the randomly chosen template was used. We can also see from Table 5.3 and Table 5.4 that the least fit template never produced the strongest results and so will not be used in future experiments for this thesis. If we look at the results produced by the fittest great-grandparent template compared to those produced by the random great-grandparent template we can see that they are stronger using Mann Whitney statistical analysis with a sample size of 26 and ($p = 0.2389$). This is not as strong a confidence value as we have previously seen in the comparisons in this chapter and suggests that

these results are not as radically different to each other as those previously presented (See Section 5.5.3). In previously published work (FitzGerald & O'Donoghue 2008) we have shown that using a randomly chosen template outperforms the use of the fittest or least fit template. For this reason we shall be using a randomly chosen template in the following experiments.

The strong result that emerges from this set of experiments is that regardless of the fitness of the template chosen non-Mendelian repair (in the case of this problem size that is the great-grandparent) templates outperform the standard Mendelian parent template. While this does not relate to the hypothesis of this section it is one of the core contributions of this thesis.

5.5.7 Effect of Population Size on Choice of Ancestral Repair Template

In the majority of cases the *A. thaliana* breeds by selfing, leading to a very homogenous population. This may suggest that ancestral repair is only needed in populations with little diversity, that is small populations. This section examines this suggestion.

Hypothesis: *The choice of GeneRepair template is independent of Population Size*

In this set of experiments I will compare the use of different population sizes. In the experiments thus far a standard size of 50 for the population has been used. In this section this is compared to the use of populations of size 100.

In Figure 5.18 the 101 City TSP was used to compare the efficiency of three ancestral repair templates on a population of 50 with the experiments

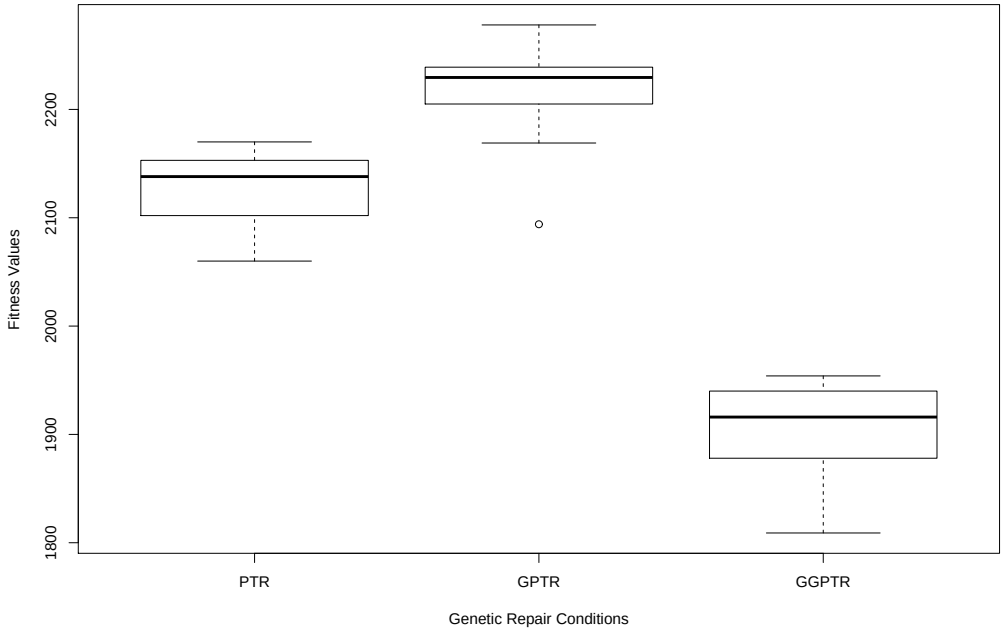

Figure 5.18: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair with Population Size 50

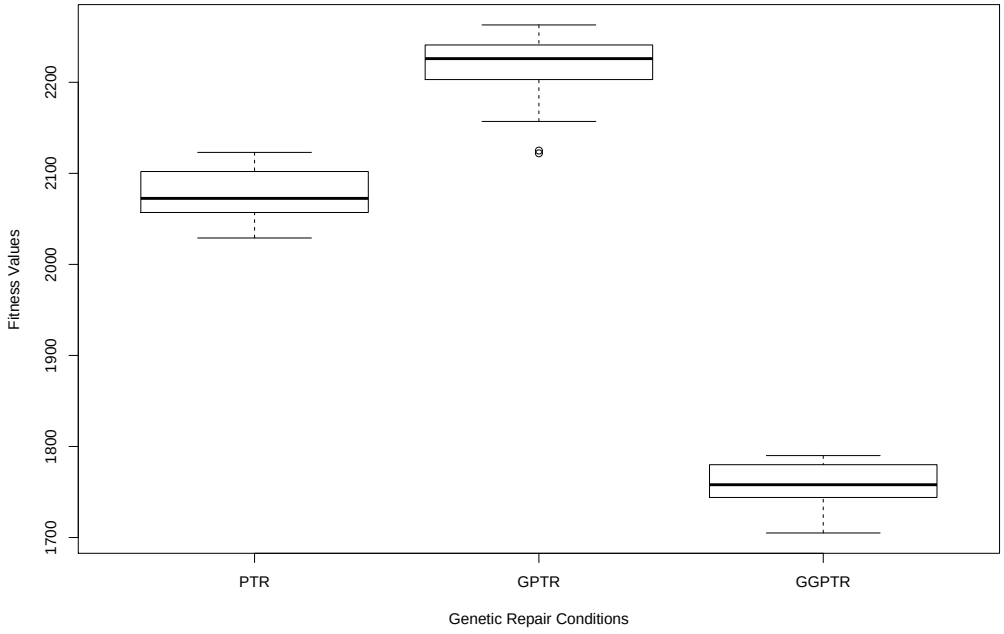

Figure 5.19: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair with Population Size 100

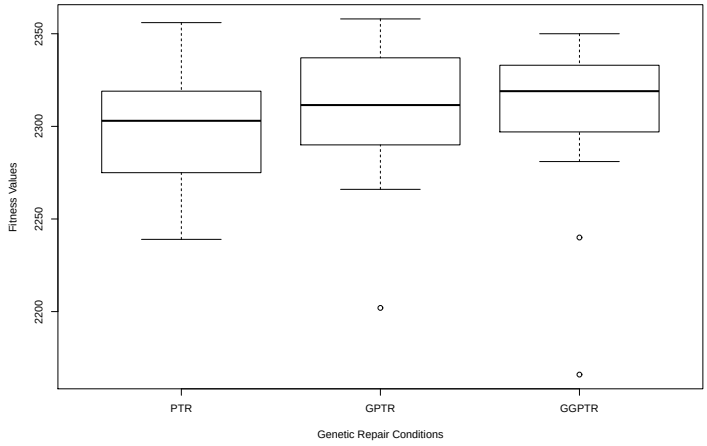

Figure 5.20: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair with Micro-Population of 4 Individuals

running for 500,000 generations. We can compare this graph to Figure 5.19 where similar experiments were carried out using a population of 100. It is clear from the results in these two graphs that population size does not appear to impact the relative efficiency of the repair templates. Therefore the grandparent remains the most effective with the great-grandparent least efficient for this problem size.

The specific tour lengths produced by the experiments illustrated in Figures 5.18 and 5.19 are illustrated in Table 5.5. In this table we can see that for both population sizes the Great Grand Parent repair template (GGPTR) produces the best results with the Parent Repair Template ranking second and the Grand Parent Repair template ranking third. When Mann Whitney statistical analysis was carried out with a sample size of 26 it concluded that great-grandparent template repair with population of 100 was more effi-

cient (produces shorter tour lengths) than great-grandparent repair template with population of 50 with ($p < 0.0001$). This strong confidence level tells us that this is a firm finding and using great-grandparent template repair with a population of 100 is more efficient than with a population of 50. When grandparent template repair was compared for populations of 100 and 50 with a sample size of 26 a population of 50 was shown to produce more efficient results with a p value of 0.4052. This is not a strong p value and therefore can not be seen as a conclusive finding. When the parent template repair was compared for both populations a population of 100 was found to produce better results than a population of 50 with a strong p value of 0.0001. While fitter results were expected from the larger population, the lack of significant difference on the GP template was quite a surprise.

In Figure 5.20 a micro population of just four individuals was used with the same experimental parameters as for Table 5.5. When this micro population was used the relative efficiency of the repair templates did not follow the pattern as for populations of 50 and 100. In Figure 5.20 parent outperforms both grandparent and great-grandparent with p values of 0.063 and 0.0107 respectively with a sample size of 26. Kruskal-Wallis analysis on these results shows a slight difference in the result sets produced with $H = 5.18$, $df = 2$ with $p < 0.075$. Perhaps for this tiny population a mutation rate of 2% was too high and non-Mendelian repair templates may have thus introduced an excessive amount of diversity. Further research could be carried out into the efficiency of ancestral repair templates when used with micro-populations as this has proved to be an interesting result.

Comparison of Population Size				
Population	Statistic	PTR	GPTR	GGPTR
50	Average	2129	2219	1909
	Minimum	2060	2094	1809
100	Average	2077	2119	1758
	Minimum	2029	2122	1705

Table 5.5: Effect of Population Size on Choice of GeneRepair Template

Looking at Table 5.5 and taking into account the described statistical analysis we can say, with the exception of the micro-population experiment, that for parent and great-grandparent repair a population of 100 is more efficient than a population of 50. We can also say that the population size does not affect the order of efficiency of the ancestral repair templates with the great-grandparent repair template once again proving to be most efficient for this problem size followed by the parent and finally the grandparent. For this reason the choice of population size (between 50 and 100) is not important for further experiments when comparing the effectiveness of ancestral repair templates (FitzGerald & O'Donoghue 2010).

5.5.8 The Effect of GeneRepair at Different Generational Milestones

The differing results produced on the larger (50, 100) and micro-population (4) suggest that population diversity may be a factor influencing ancestral genetic repair. This also echoes some of the more recent findings on *Ara-*

bidopsis thaliana (Hopkins *et al.* 2011). With this in mind, we now explore the performance of GeneRepair under situations of greater population diversity (during early evolution) with the performance of GeneRepair in the presence of less population diversity (late evolution).

Hypothesis: *The most effective template is not the most efficient at every generational milestone*

This Section investigates at whether the effectiveness of the repair templates changes as the generations continue to evolve. Each of the experiments above were run for 500,000 generations but only the final results have been presented and discussed. In this Section I will show the intermediate results produced at 10, 100, 10000, 100000 and 500000 generations. We can therefore identify where GeneRepair carries out most of its changes. By conducting further experiments into each of the generational milestones perhaps an ideal number of generations could be found at the point where GeneRepair is found to be most active.

Table 5.6 illustrates the results produced at each generational milestone for the 101 City TSP at the optimal mutation rate (of those compared) of 0.75%. This is only the second time in this thesis that the parent template has been more efficient than its non-Mendelian counterpart. At 500,000 generations the parent template outperforms the non-Mendelian template with ($p < 0.0001$) when Mann Whitney statistical analysis is conducted. For mutation rates of 1%, 1.25%, 1,5%, 1.75% and 2% the great-grandparent repair template was more efficient than the parent template with a sample size of 52 and ($p < 0.0001$). For mutation rate of 0.1% the great-grandparent

	10			100			1,000			10,000			100,000			500,000		
	PTR	GPTR	GGPTR	PTR	GPTR	GGPTR	PTR	GPTR	GGPTR	PTR	GPTR	GGPTR	PTR	GPTR	GGPTR	PTR	GPTR	GGPTR
Average	2835.3	2806	2822.2	2167.9	2222.0	2128.8	1076	1196	1144	866	931	926	775	810	813	728	747	759
Minimum	2693	2634	2686	1938	1984	1786	949	1047	985	773	829	779	715	746	721	694	693	690

Table 5.6: Comparison of Ancestral GeneRepair at Six Generation Milestones

repair template again outperformed the parent template with a p value of 0.2389 for a sample size of 52. However, as at the optimal mutation rate, with the mutation rate of 0.75% the parent template outperformed the great-grandparent template with a p value of 0.0918 and also at the mutation rate of 0.5% parent was more efficient than great-grandparent repair with a p value of 0.2483. From this we can see that at the optimal mutation rate of 0.75% and slightly lower mutation than optimal, 0.25% and 0.5% the parent template outperforms the great-grandparent. But at all other mutation rates the great-grandparent is most efficient. Perhaps near this optimal rate of mutation only slight diversity is needed from the repair template.

Table 5.6 shows that the repair template is sensitive to number of generations when looking at the average result. This is in support of the hypothesis that the most effective template is not the most efficient at every generational milestone. If we look at the minimum result produced at each milestone we can see that the PTR is best at 1,000, 10,000 and 100,000 generations while non-Mendelian repair templates are best at all other milestones. While the average result shows the parent template to be most efficient (See statistical analysis is previous paragraph) the best result was produced by the great-grandparent template.

5.5.9 Random Template Repair

In the above experiments the use of different ancestral repair templates within the GeneRepair technique was compared. The experimental results indicate that the choice of repair template should be selected independently of its fit-

ness relative to other ancestors of that generation (as opposed to the fittest template) and that repair should be carried out in a random and varying direction. This may suggest that the use of random template repair (RTR) (Lichtblau 2002) as opposed to an ancestral template may improve the experimental results even more. RTR is a form of template driven repair that uses templates that are randomly generated for each invalid individual. This offers the maximum diversity in the contents of the repaired alleles. RTR identifies duplicates and produces a list of missing information. RTR then randomly chooses a piece of missing information from the list and replaces a duplicate. In this way the order of the repaired genes is completely random as opposed to their order being derived from some ancestral template. It should be noted that RTR is a form of inheritance that is both non-Mendelian and also non-Darwinian. Errors in the invalid individual are repaired using a completely random template. This method is used by Mathematica for general purpose combinatoric and discrete optimisation problems (Lichtblau 2002).

In this section of experiments I compared the use of a random repair template (RTR) to each of the three ancestral templates investigated so far. In contrast to RTR the GPTR and GGPTR strategies are non-Mendelian but are fully Darwinian, with only PTR being both Mendelian and Darwinian.

Hypothesis: *Using an ancestral repair template is superior to using random template repair(RTR)*

However, this situation looks different when we examine a larger problem. Figure 5.21 compares the results produced by RTR against the three ancestral

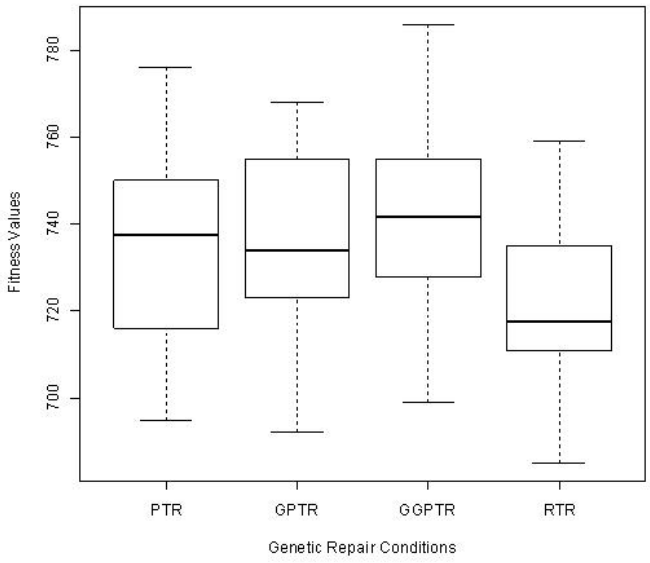

Figure 5.21: Comparison of Ancestral Repair Templates and RTR for the 101 City TSP

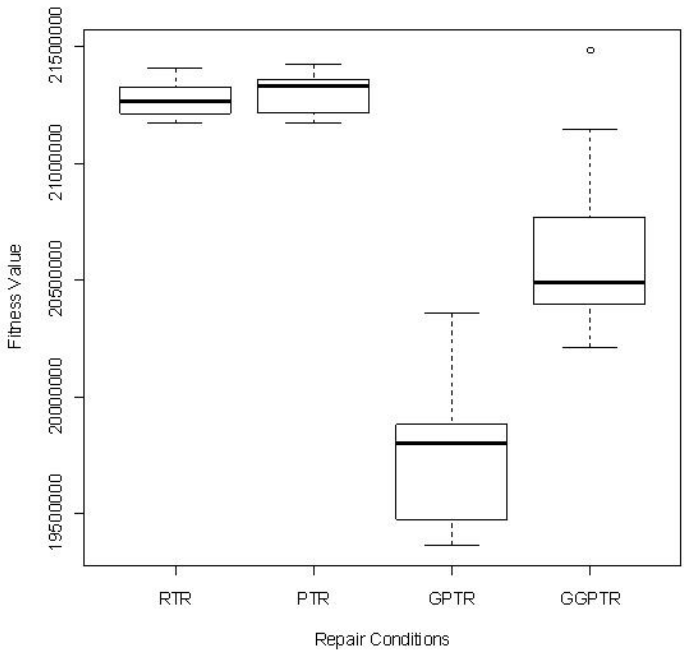

Figure 5.22: Comparison of Ancestral Repair Templates and RTR for the 18512 City TSP

strategies, for a small 101 city problem. The RTR performs very well on this initial small problem, producing the lowest average and minimum tour lengths. This can be attributed to RTR exploring the most diverse solution space as it is operating as a form of blind search, modifying the given search space provided by the population. Unlike ancestral repair the error detection and error correction is completely random so which duplicate is replaced with which extra piece of information is random.

Figure 5.22 compares the results produced by RTR against the three an-

cestral strategies, for a larger 18512 city (Reinelt 1991) problem. The box plot shows the results after 50,000 generations, using a population of 10 and mutation set at 0.01%. This boxplot shows that the non-Mendelian repair strategies are far more effective than either RTR or PTR on the larger 18512 city problem. This illustrates the advantage to be gained from using non-Mendelian repair, because the repaired alleles have already been evaluated in a previous generation and have surviving ancestors to attest to their quality (FitzGerald & O'Donoghue *in preparation*). RTR performs well on small problems as its blind search strategy is effective on smaller search spaces. However when RTR is applied to larger problems, these random explorations generally prove fruitless. On these problems, the ancestral repair strategies have a far greater likelihood of producing a reasonably fit individual (FitzGerald & O'Donoghue *in preparation*).

5.5.10 The Effect of Problem Size on Ancestral Repair Template Efficiency

In the above Sections the 51, 101 and 18512 City TSP were used to illustrate a number of experimental results in order to find support for a number of hypotheses. In this Section I will show the above experiments using the 532 City Tour and the 76 City Tour in order to illustrate that the results produced are not problem size specific. The details of each of these TSPs can be found online by accessing the TSP library (TSPLIB).

Hypothesis: *A Non-Mendelian Repair Template will Outperform the Mendelian Counterpart regardless of Problem Size*

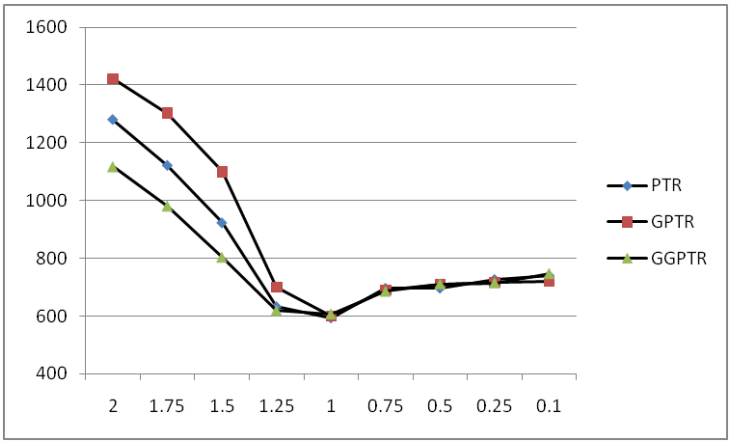

Figure 5.23: 76 City TSP Repair Template and Mutation Rate Effectiveness

As in Figure 5.14 Figure 5.23 shows the same finding. Each of the repair templates investigated has the same optimal mutation rate of those rates tested. At the optimal mutation rate, 1%, the best repair template to use differs slightly than at the other mutation rates (as previously illustrated in Section 5.5.8. This graph supports the hypothesis that the repair templates do not have different optimal mutation rates.

In Figure 5.24 the results found when the att532 TSP was used to compare three different repair templates are illustrated. In this experiment we can see that the Great-Grandparent repair template produces the fittest results followed by the Grandparent and lastly the Parent (FitzGerald & O'Donoghue 2010) at each generational milestone. This Figure supports the hypothesis that the most effective template is most efficient at every generational milestone.

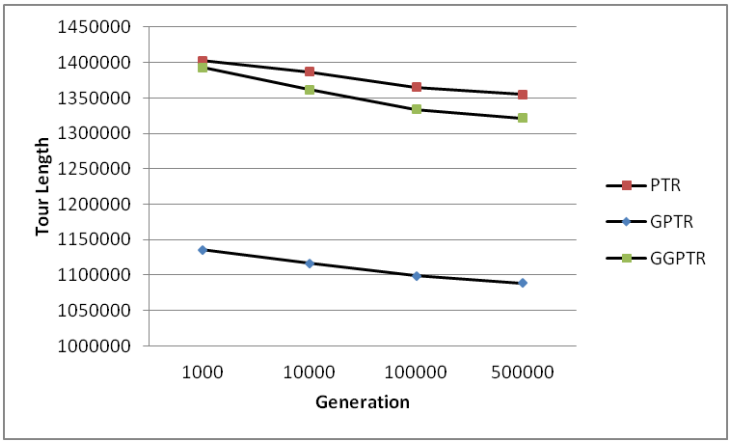

Figure 5.24: 532 City TSP Effect of Repair at Each Generation Milestone

This section illustrates that regardless of problem size the repair template chosen does not affect the choice of optimal mutation rate. The mutation rate is dependent on the problem size and not the repair template. This section went on to show that the most effective template is most efficient at every generational milestone using the 532 City TSP. The results already illustrated in this chapter were produced using the 101 City tour as the problem set. We can now see that the previously presented results were not dataset dependent. This section has illustrated that these results can be reproduced using different size TSPs.

5.5.11 Selection Methods

Returning briefly to the inter-domain analogy, it might be said that ancestral repair only works when there is little selection pressure - such as that found amongst the predominantly selfing *A.thaliana* (Hopkins *et al.* 2011). In this

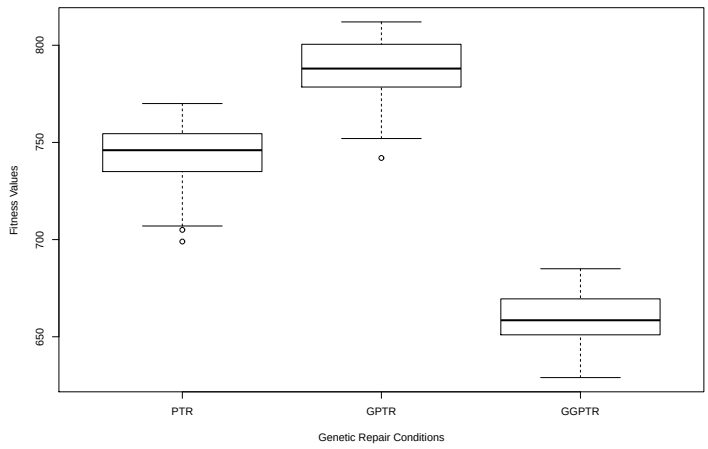

Figure 5.25: 51 City TSP with Tournament Selection Method

Section a comparison is made of the effect of selection method (See Section 2) on the relative performance of each of the repair templates. Each experiment in this section is carried out using the 51 City TSP with a mutation of 2% and a population of 100.

Hypothesis: *The choice of ancestral repair template is not changed by the selection method used*

Figure 5.25 shows the results produced when the tournament selection method is used while Figure 5.26 shows the results produced when the truncation selection method is used. In each of the graphs the best results are produced when the great-grandparent repair template is used. You can also see that the worst results are produced when the parent template is used in conjunction with the GeneRepair mechanism. The order of effectiveness of the repair templates is not sensitive to the selection strategy used. We can

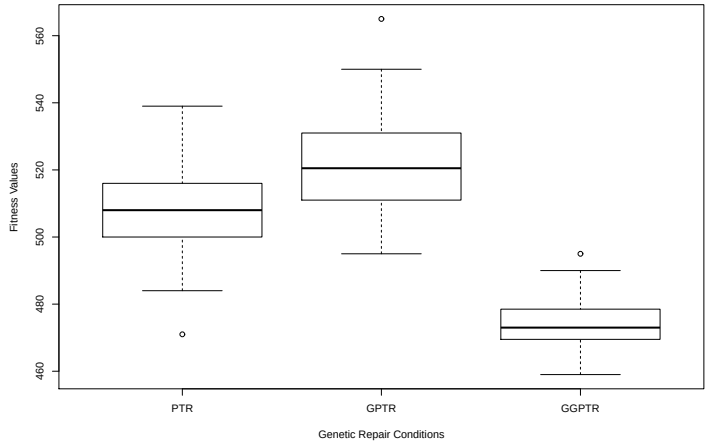

Figure 5.26: 51 City TSP with Truncation Selection Method

also see that the repair templates perform in the same order for both selection mechanisms. Interestingly, the results produced are better when the truncation method is used. Two separate figures (5.25 and 5.26 were used to clearly illustrate the results produced by each Selection method. The comparison of these results is then shown using Figure 5.27 so that the difference between the selection methods is clearly seen. In this Figure we can see that the order of efficiency of the templates is independent of the selection method used, while the superior results are produced when the truncation selection method is used.

To ensure this result the same experiment was carried out to compare the use of tournament and truncation selection at the optimal mutation rate (of those tested in Section 5.5.5 of 1.75%). The results of this experiment are illustrated in Figure 5.28. In this Figure we can see that even at the optimal mutation rate the truncation outperforms its tournament competitor. In

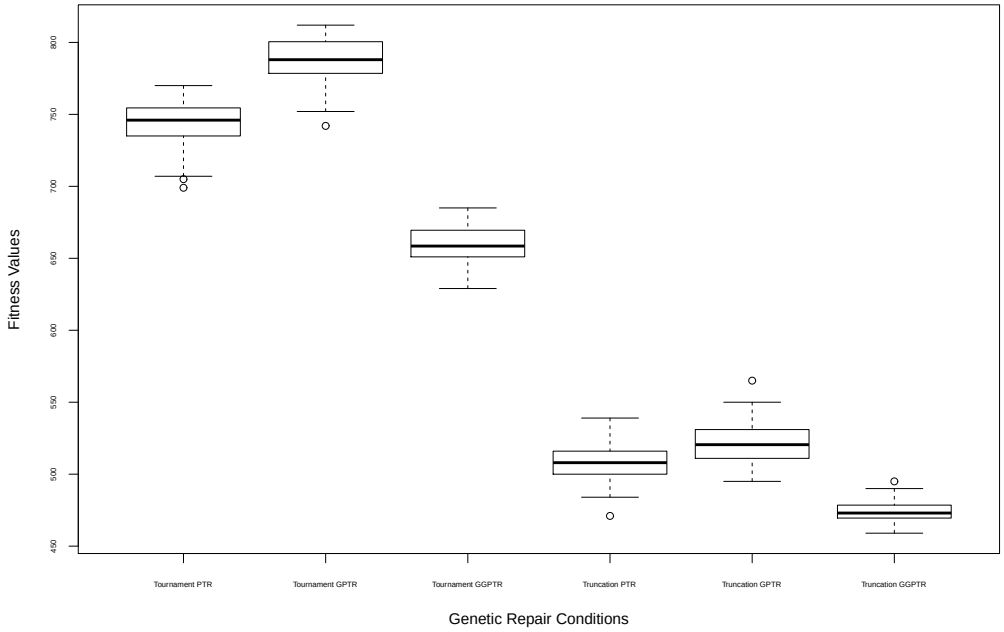

Figure 5.27: Comparing the use of both Truncation and Tournament Selection Methods on the 51 City TSP with 2% Mutation

Figure 5.28 the results produced using truncation selection are also tending towards the optimal which is 426 for this problem. Truncation selection is the method used in all experiments in this chapter. While there are a huge number of selection methods available the objective of this Section is not to find the best one but merely to show that they do not affect the choice of repair template chosen.

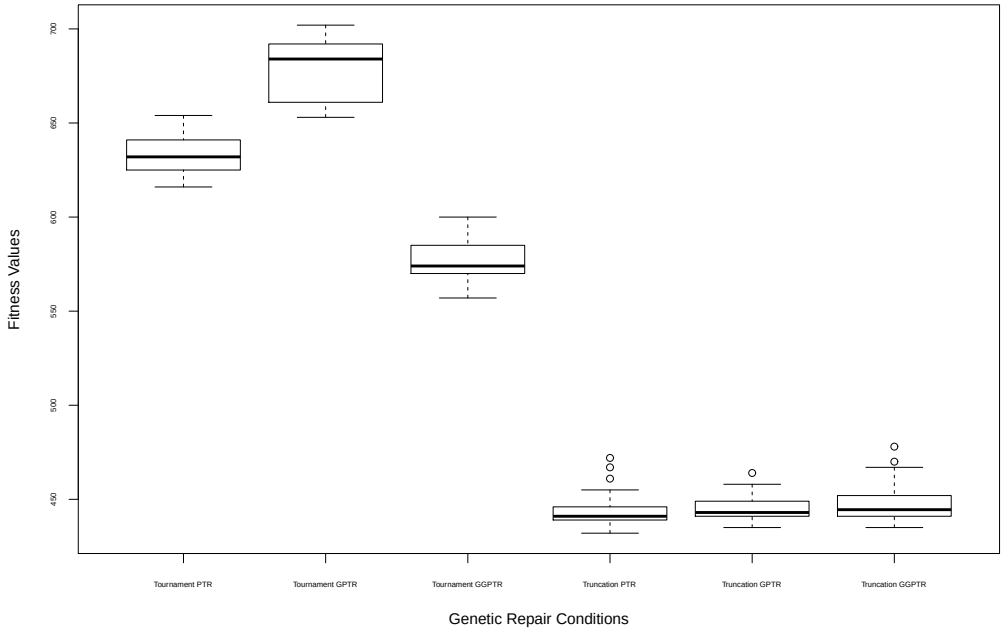

Figure 5.28: Comparing the use of both Truncation and Tournament Selection Methods on the 51 City TSP at the optimal mutation rate of 1.75%

5.5.12 Direction of Repair

Thus far we have seen how non-Mendelian repair templates can outperform their Mendelian counterparts. We have investigated the various parameter choices available within GeneRepair to examine the behaviour of the three ancestral repair templates in a wide variety of situations. We have found that using a randomly chosen template (as opposed to the fittest) produces superior results. We have also seen that for larger problems non-Mendelian repair templates outperform RTR, which is the method of using a randomly

created repair template as opposed to an ancestral repair template. In this Section we will examine the behaviour of GeneRepair further by looking at another parameter choice which is the direction of repair.

Hypothesis: *Performing repair in a consistent and uniform direction will produce weaker results than randomly varying the direction in which the repair process operates*

In Section 4.7 the effect of repair direction on the repaired individual was discussed. There are three choices of repair direction: Right to Left, Left to Right and a Random and Varying Direction. It is important to remember that the random and varying direction of repair is still linear and varies between the other two directions. The differing effect of these repair directions are clearly illustrated in Figure 4.9. This current section illustrates the result produced by a number of experiments to investigate the three directions of repair. Again, for this set of experiments a population of 50 individuals has been used and the experiment was run for 500,000 generations with a mutation of 2.0% using the 101 City TSP.

In Figures 5.29, 5.30 and 5.31 the effect of the repair template is compared when GeneRepair is carried out in a left to right, right to left and random and varying direction respectively. We can see from these three Figures that the order of effectiveness of the repair templates, when compared to each other, is identical for each of the three repair directions. In all three of the Figures (5.29, 5.30 and 5.31) the great-grandparent repair template is most effective. The parent repair template is more effective than the grandparent repair template which is least effective for this problem size. Regardless of

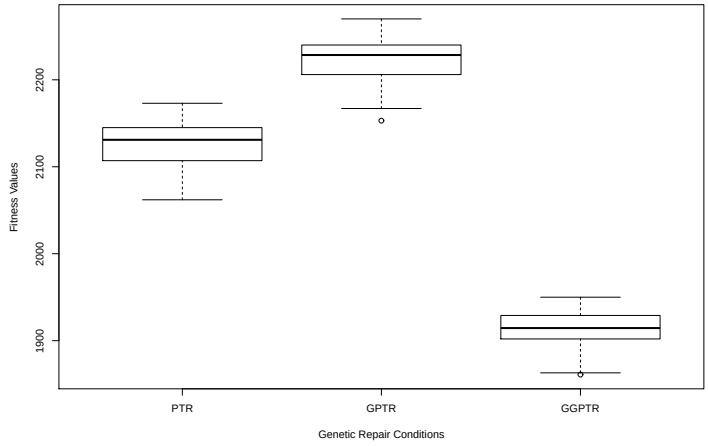

Figure 5.29: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair Performed in a **Left to Right** Direction

the direction of repair the non-Mendelian repair template (in the case of this problem size this is the grandparent) is superior to the Mendelian repair template (the parent).

In Table 5.7 the results are investigated further by looking at the actual tour lengths produced. The `Average´in Table 5.7 refers to the average tour length over the full set of experiments while the `Minimum´refers to the lowest tour length produced across the 26 repetitions of this experiment.

In Section 5.5.6 we saw that under certain circumstances a randomly chosen template, as opposed to the fittest template, produced the best results. Table 5.7 compares the three different repair directions using a randomly chosen template.

However when Mann Whitney statistical analysis was carried out on the results in this Table the confidence levels were not conclusive. Using a sample

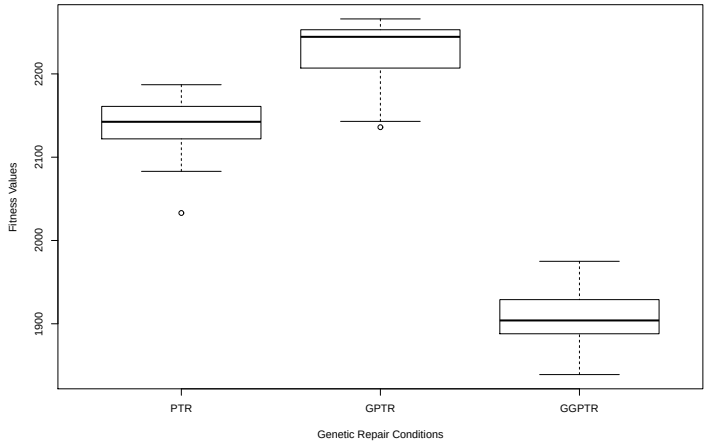

Figure 5.30: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair Performed in a **Right to Left** Direction

size of 26 the statistical analysis showed that the Random great-grandparent template did not outperform the left to right great grandparent template ($p < 0.4562$). This is quite a high p value which indicates there is not a strong enough difference between the result sets to draw a firm conclusion. The statistics also showed that the great-grandparent template acting in a random direction did not outperform the great-grandparent template acting in a right to left direction either with a p value of 0.4364 which is also too high to show a strong difference between the result sets. When each of the sets of results were compared with their alternative direction counterpart (grandparent template acting in a random and varying direction compared with grandparent template acting from right to left etc.) there were only two strong conclusive p values. The lowest p value was obtained when the grandparent repair template was used with repair acting in a random and varying

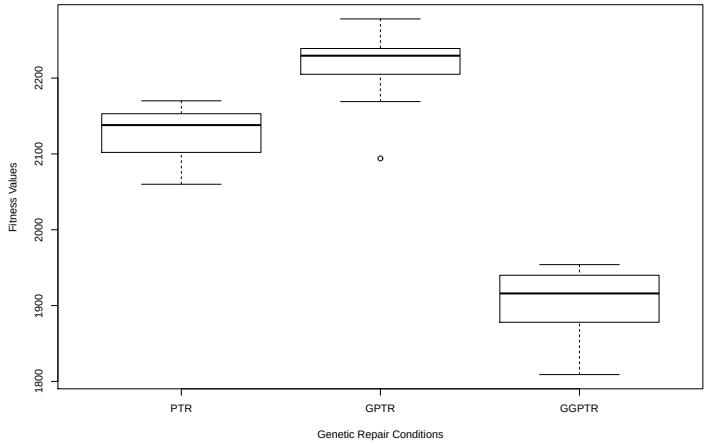

Figure 5.31: Comparison of Parent, Grandparent and Great-Grandparent Ancestral Repair Performed in a **Random and Varying** Direction

direction compared with using the grandparent repair template acting in a right to left direction. Mann Whitney statistical analysis showed that the random direction was more effective than the right to left direction with a p value of 0.0764. The other strong p value was obtained when parent template repair acting in a random direction was compared with repair acting from right to left. Repair acting in a random and varying direction was more effective than in a right to left direction with a p value of 0.1788.

From these two statistics and the fact that the lowest result overall results (shortest tour length) and the lowest average result were achieved by the great-grandparent repair template with repair carried out in a random and varying direction we can suggest that the repair direction to use should be the random and varying repair direction. The reason for this may be that the great-grandparent would give the highest amount of diversity across

Comparison of Repair Directions on choice of Ancestral Repair Template				
Direction	Statistic	PTR	GPTR	GGPTR
Right to Left	Average	2135	2227	1908
	Minimum	2033	2136	1839
Left to Right	Average	2127	2222	1912
	Minimum	2062	2153	1861
Random and Varying	Average	2129	2219	1909
	Minimum	2060	2094	1809

Table 5.7: Comparison of Three Different Repair Directions with Random Template Selection

the population and the random and varying direction would also maintain diversity. If we concentrate on the direction of repair - the results are so far away from their nearest competitor I suggest that carrying out repair in a random and varying direction produces the most suitable level of diversity which leads to strong results (FitzGerald & O'Donoghue 2008) especially for permutation problems. Additionally, in the absence of a clear reason for selecting a specific direction, we opted for the "random and varying"as the preferred repair direction

5.5.13 Storage of Ancestors

Hypothesis: *The method used to choose the specified ancestor stored does not affect the overall result*

In the experiments above there was a choice between two ancestors for each

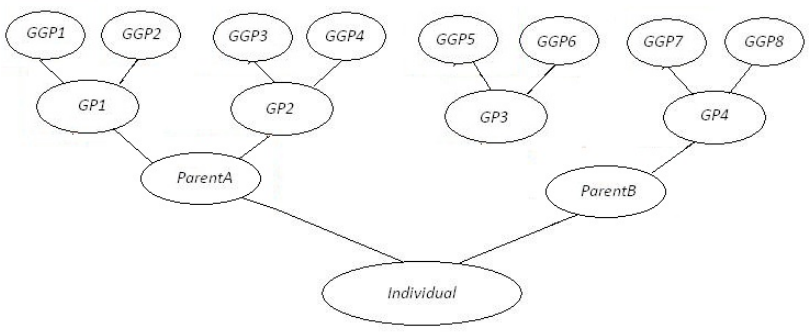

Figure 5.32: The Family Tree of each individual in the population

GeneRepair mechanism unless otherwise specified (See Section 5.5.6). The selection between available ancestors was carried out in an arbitrary way. In this section we address a potential criticism that our method for selecting the repair templates does not actually implement the stated strategy. For example in Section 5.5.6 one of our results attests to using the fittest great grandparent as a repair template. In the implementation of the EO a subset of the possible ancestors is stored for each individual. GeneRepair stores **two** of each ancestor (parent, grandparent and great-grandparent) for each individual. This improves computational time by decreasing memory costs and complexity. In this Section I will explain how the ancestors to be stored are chosen and compare this method to the alternative to show the effect on results produced.

In Figure 5.32 we can see the family tree for each individual in the population. For clarity each Grandparent is labelled GP and each Great-Grandparent is labelled GGP. We can see that the individual has two par-

ents, four grandparents and eight great-grandparents. The issue is that we present results at the great-grandparent level, however the templates are only stored anew when they are at the parent level. The system does not store all eight possible great-grandparents for each individual. That is, templates enter our ancestral "conveyor belt" that maintains templates of different ages. So, when we presented the fittest great-grandparent results this is the fittest of the stored pair for this individual.

The choice of which ancestor to store is arbitrarily made by choosing ParentA (See Section 3) of the parent and storing this at each generation. ParentA is the parent with the lowest array index of the two parents. For the first generation both parents are stored for the individual. At generation two ParentA of the first individual selected for the crossover operation is stored as the first grandparent and ParentA of the second individual selected for the crossover operation is stored is stored as the second grandparent. Thus begin an ancestral "conveyor belt". This choice was made to reduce the amount of information that needs to be stored and increase computation time. This arbitrary choice does not affect the results produced. In order to illustrate this I carried out an experiment where a random ancestor was chosen instead of always choosing ParentA of the given ancestor. I did this by randomly (with a 50% probability) swapping ParentA and ParentB of each individual so that a random side of the individual was stored.

In Figure 5.33 I have shown the results produced when three ancestral repair templates are compared through the running of an experiment of 101 Cities with a population of 50 and a mutation rate set to 2% for 500,000

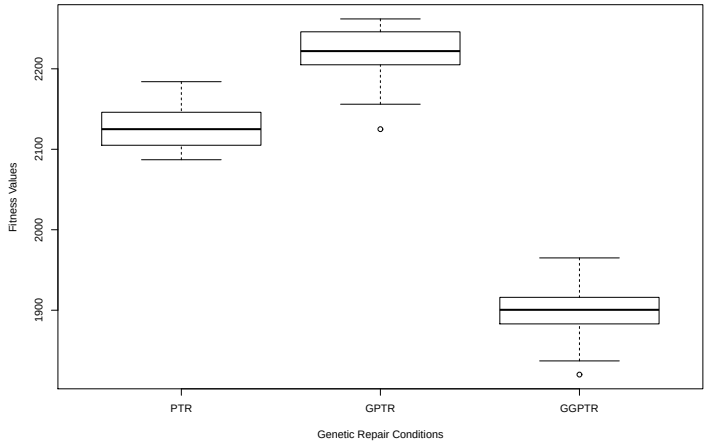

Figure 5.33: Random Parent Stored as opposed to arbitrarily choosing ParentA of the given ancestor

generations with repair acting in a random and varying direction. In this experiment ParentA and ParentB were randomly swapped (with a 50% probability) to ensure that ParentA of the given ancestor was not always the stored template. In Figure 5.34 the results produced by storing an arbitrary parent (as described previously in this Section) are illustrated. To compare these two sets of results Figure 5.35 compares parent, grandparent and great-grandparent repair template storage. In this graph we can see that the great-grandparent template stored in an arbitrary way sits on top of the results for the great-grandparent stored in a random way as do the parent templates and the great-grandparent templates. The order of effectiveness of the templates is the same. This supports the hypothesis that the method used to choose the ancestor stored does not affect the overall result.

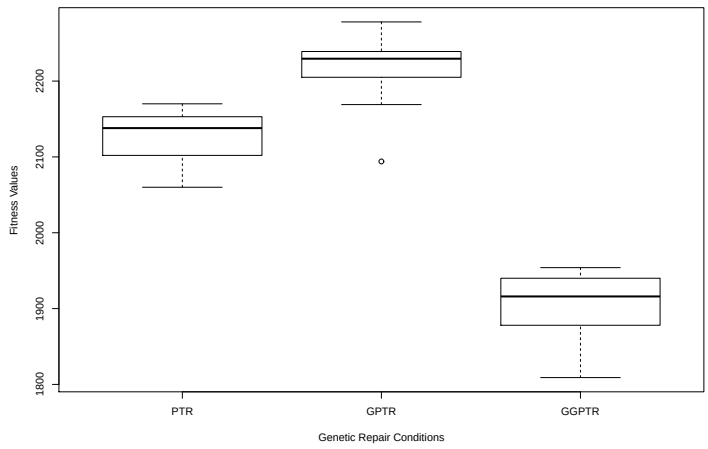

Figure 5.34: Storing ParentA of the given ancestor as in the rest of the experiments

5.5.14 Discussion of Computationally Focused Investigation

Given the results presented thus far in this chapter, there were two ways we could have proceeded. One was to apply ancestral repair to a variety of different problems which could possibly include other permutation problems, combinatorial problems, or constrained numeric optimisation etc. The current implementation is limited to certain permutation problems but could be adapted quite easily to produce valid solutions to a wider set of constrained problems.

The other option was to explore ancestral repair under more biologically inspired conditions. While the previous results were based on a loose analogy between the two domains, the rest of this chapter strengthens the analogy

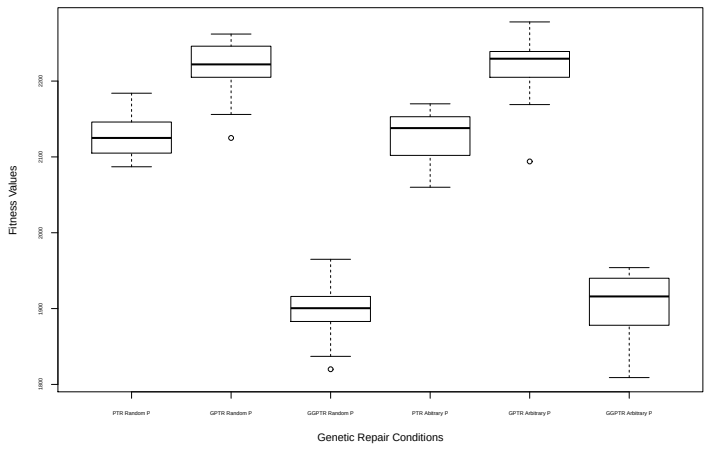

Figure 5.35: The Comparison of Storing ParentA of the given ancestor as in the rest of the experiments as opposed to a Random Parent

by importing more biologically founded parameters into our computational model. We chose this second route because our results were proving to be of huge interest to the original proposer of non-Mendelian repair in the *Arabidopsis thaliana* Dr Susan Lolle [1] and to a number of her collaborators. The results so far inspired a natural line of questioning as to whether the repair templates would remain as effective when more biologically inspired parameters were used. We decided to carry out a set of experiments with parameters that attempt to mimic those in the biological realm. This leads on to the second objective of this thesis.

[1]Personal communication with Dr Susan Lolle in University of Waterloo, Ontario, Canada

5.6 Objective 2 - Biologically Focused Investigation of GeneRepair

At this point, the reader is reminded that the proposed non Mendelian repair theory suggested to exist in the *Arabidopsis thaliana* plant (Lolle *et al.* 2005) has proven to be very controversial. In this section we attempt to use our model to see if we can shed any light on the process. In this section we perform stress or reliability tests to test the mechanism under, what could be seen as, more "biologically"inspired parameters. While inconclusive, these results have generated suggestions that are of great interest to the biological community [1].

While Section 5.5 compared Mendelian to non-Mendelian repair using a number of different parameters and problem sets, this Section will delve deeper into the parameters of interest in the field of biology. This Section also further investigates whether any other mechanism can be employed with GeneRepair to improve it even more. As previously stated in Section 1.5 what we term as "biological experiments"are still computational evaluations and should not be confused with systems biology or related disciplines.

5.6.1 Reduced Redundancy Representation and Ancestral Repair Templates

An optimal solution to a 3 city TSP could be represented in 6 different ways, as locii are interchangeable. Removing this locus interchangeability might

[1] Personal communication with Dr Susan Lolle in UW and Dr R. Palmer in ISU

focus evolution and thereby affect the relative fitness of the repair templates.

Hypothesis: *DNA is an order based representation with a definite start and end. Applying such a structure to our representation and thus reducing the number of representations for each optimal solution might affect the choice of ancestral repair template used*

DNA is an order based representation with definite start and end codons. Would such a structure applied to the representation of individuals in an EO alter the effect of the ancestral template? The TSP is a circular tour beginning and ending at the same point (See Section 5.2. For this reason the specific city that represents the first city of the tour (and also the end point) is not important as the tour can be seen as a closed loop. The important property of the tour is the order of the cities.

In this Section the use of one fixed allele is investigated(*Acyclic Representation*), where the first city is fixed to City 0 (an arbitrary choice), with Non-Fixed (*Cyclic Representation*), where the EO is run as normal and the first city is randomly decided by the EO, are compared. We acknowledge that the term "Acyclic"is an exaggeration and merely use this term to distinguish between the greater degree of redundancy in the "Cyclic"representation as compared to our "Acyclic"representation. By fixing the first city the computation time for the EO should be decreased. While one may argue that the acyclic representations reduce diversity at each locus and thus the diversity expected within the whole population, it may therefore slow down the EO the number of computations are greatly reduced by effectively removing one element from the individual. This could be seen as reducing the prob-

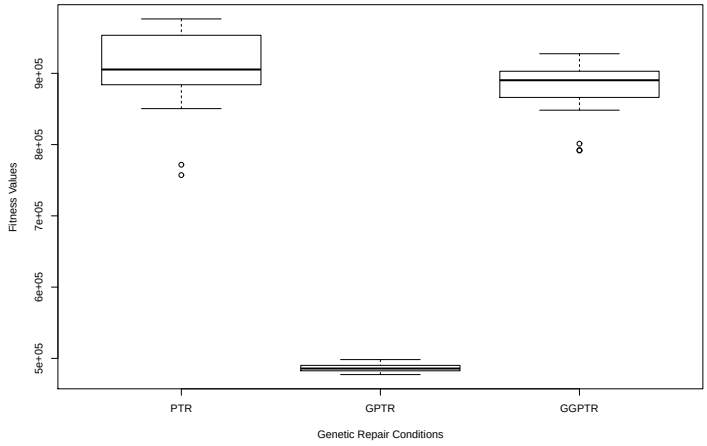

Figure 5.36: 1379 City TSP Without Fixing First City

lem complexity to that of *numberOfCities - 1* which will therefore reduce computation time for the EO.

In Figure 5.36 we can see the results of the experiment where a 1379 City TSP (Reinalt 1991) was used to compare the effectiveness of three ancestral repair templates. This experiment was run for 10,000 generations with a population of 100 and a mutation rate of 0.1%.

The results shown in Figure 5.37 were produced using the same parameters except that the first city was fixed to zero. We can see that the order of effectiveness of the three repair templates does not change when the first city is fixed. The grandparent repair template is shown to be the most effective template by a wide margin in both cases. Figure 5.37 and Figure 5.36 support the hypothesis that fixing the first city of the population does not change the order of effectiveness of the repair templates. Therefore, this result does not undermine the ancestral repair hypothesis.

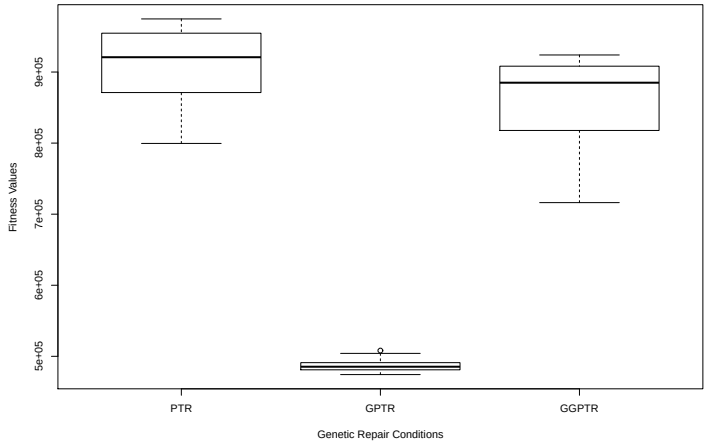

Figure 5.37: 1379 City TSP Effect of Fixing First City

Mann Whitney statistical analysis was carried out to compare these two sets of results using a sample size of 26. No difference was found between the cyclic and acyclic results for the great-grandparent template ($p < 0.4168$). When the grandparent template and parent template GeneRepair were compared when acyclic and cyclic representation was used no significant difference was found between both sets of results ($p < 0.496$). These p value show that the results produced by cyclic and acyclic representation are not different enough to produce a strong confidence level. This shows that the choice of acyclic and cyclic representation does not affect the efficiency of the repair templates and so also does not affect the order of efficiency of the ancestral repair templates.

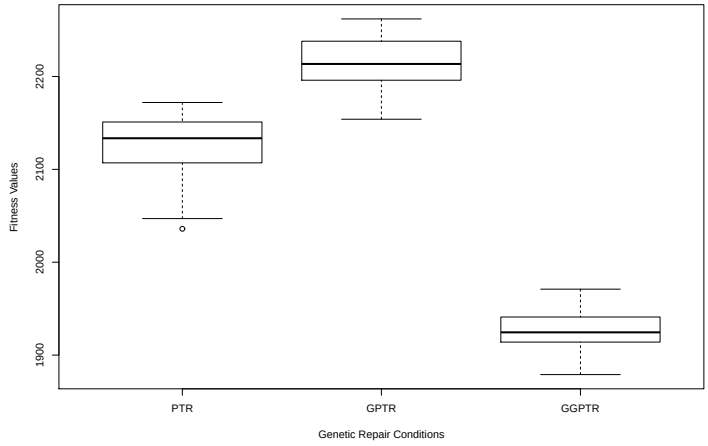

Figure 5.38: 101 City TSP with self-crossover Prohibited

5.6.2 Non-Self Crossover

In all of the previous results described in this chapter self crossover has been allowed. The reason that this crossover property is of interest and is examined in this thesis is that the *Arabidopsis thaliana* has the ability to self-crossover (Meinke, Cherry, Dean, Rounsley & Koornneef 1998) and in the biological world this is referred to as a *selfer*. Selfing is the main process for propagation - common to many crop plants(Hopkins *et al.* 2011).

Hypothesis: *Ancestral repair will perform differently on inbred populations where self-crossover is prevalent than in populations where self-crossover in prohibited*

In Figure 5.38 we can see the results produced by running the EO on a 101 city TSP with a population of 50 and a mutation rate set to 2.0% for 500,000 generations with repair acting in a random and varying direction.

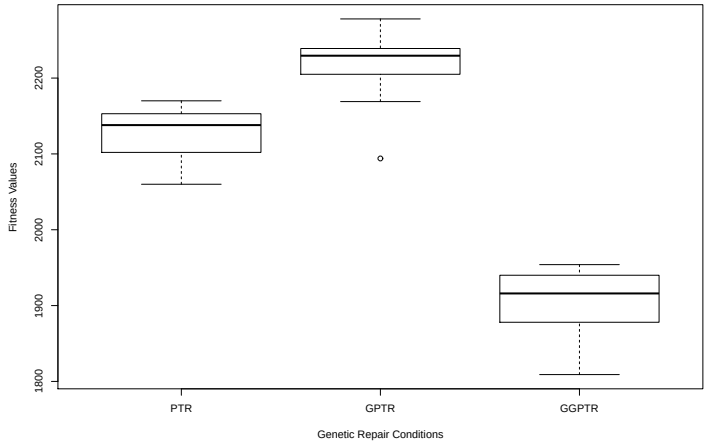

Figure 5.39: 101 City TSP with self-crossover Permitted

The difference with this experiment is that self-crossover was prohibited. If we compare this to Figure 5.39 we can see that the order of efficiency of the ancestral repair templates is not altered by the use or prohibition of self crossover. This allows us to reject the hypothesis that ancestral repair performs differently in non-inbred populations. Thus, ancestral repair templates are not greatly affected by the presence or absence of inbreeding in the populations. This was the expected result as the amount of inbreeding would generally be considered relatively small within the EO.

If we compare the use of great-grandparent repair templates using Mann Whitney statistical analysis with a sample of 26 allowing self crossover than when self crossover was not allowed, with a p value of 0.0668. This confidence level shows that there is a significant difference between the two sets of results and that allowing self crossover when using the great-grandparent ancestral repair template outperforms the use of this template when self crossover is

not allowed. This result indicates that allowing inbreeding can have positive consequences - we note also that this might be an indirect means of introducing "elitism"into our population. The reason that this investigation into the use of self crossover was carried out was in order to mirror the *Arabidopsis thaliana* plant which allows self-crossover. This finding that self crossover is beneficial to great-grandparent GeneRepair may link in some way to the fact that the plant allows self crossover and uses non-Mendelian inheritance. This may be an area that deserves further study and investigation in order to fully explain the meaning of this finding.

When the use of grandparent template repair was compared using self crossover and prohibiting self crossover, prohibiting self crossover was shown to produce stronger results but with a weaker p value of 0.1562 with a sample size of 26. This confidence value suggests that prohibiting self crossover produces stronger results when the grandparent repair template is used but not to the same extent as self crossover performs when the great-grandparent repair template is used. There was no significant difference between the two sets of parent repair templates with non self crossover producing stronger results with a p value of 0.409. Again this result suggests that population diversity is a significant factor in the use of older ancestral templates for genetic repair.

This section shows that the use or disuse of self-crossover does not affect the order of efficiency of the ancestral repair templates. We have also seen that for great-grandparent ancestral repair allowing self crossover produces better results than when it is not allowed.

5.6.3 Low Mutation Rates

Section 5.5.5 explained the effect of the mutation rate on diversity. When this mutation rate is decreased the amount of diversity it introduces into the population decreases also. Another step which introduces diversity into the EO is GeneRepair. As we decrease the mutation rate, we can see GeneRepair as one possible operator that can help re-introduce additional diversity into an overly homogenous population. In order to examine this property experiments were carried out to compare the effectiveness of GeneRepair at very low mutation rates. As previously stated, the biological inspiration for this thesis arises from a non-Mendelian repair mechanism in the *Arabidopsis thaliana* plant (Lolle *et al.* 2005). This theory might suggest that ancestral templates are more effective in the presence of low mutation rates. The *A.thaliana* plant experiences mutation at a rate of approximately only 1 per billion alleles per generation (Weigel & Jürgens 2005). This experiment may be seen as the closest model to the *Arabidopsis thaliana* plant of all of the experiments illustrated as it has a very large number of cities which corresponds to the long genome of the plant and it is being tested at very low mutations which would also correspond to the low mutation rate experienced by the plant.

Hypothesis: *At low mutation rates non-Mendelian repair templates can provide additional diversity to the EO*

Table 5.8 shows the results produced when an EO was run on the 18512 City TSP with a population of 10 for 10,000 generations. We can see that even at very low mutation rates the grandparent repair template is the most

Comparison of Repair Template Effectiveness using Low Mutation Rates				
Mutation Rate		PTR	GPTR	GGPTR
0.1%	Average	54927424	49961984	54860117
0.01%	Average	43811928	43054071	44319322
0.001%	Average	50137666	44726073	48071941

Table 5.8: Very Low Mutation comparison of Repair Template Effectiveness

effective with a p value of <0.0001 for mutations of 0.1% and 0.01%. The finding of this experiment is that at very low mutation rates the grandparent template should be used by GeneRepair to produce the strongest results. Non-Mendelian repair templates have shown themselves to be a strong contributor to the overall diversity of the EO. This introduction of diversity is of particular importance at low mutation rates when otherwise diversity may be greatly reduced leading to the EO finding plateauing at a local optimal.

5.6.4 Diversity Maintenance Illustrated by Investigating the Average Fitness of Individuals in each Generation

In order to illustrate that diversity is maintained across the experiment I have investigated the average fitness of the individuals in the population. If diversity decreased across the population this would mean that individuals would become similar and thus the average fitness would tend towards the lowest fitness as the range would decrease. If diversity was maintained across the population the average fitness would not tend towards the minimum

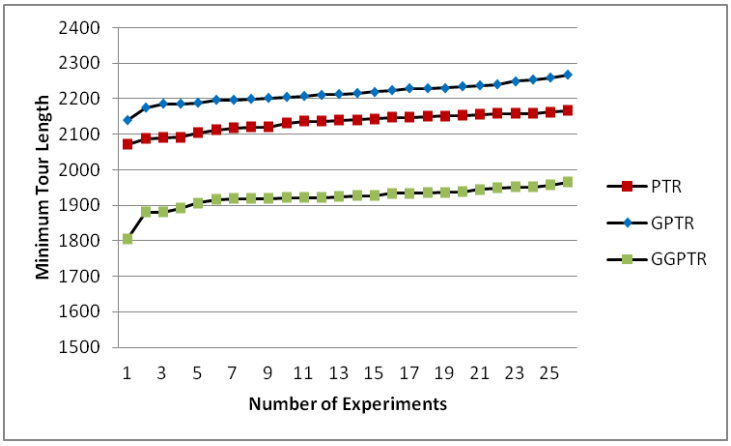

Figure 5.40: 101 City TSP with Minimum Tour Length Found for Each Experiment

fitness (fittest individual) as the large or diverse range of individuals would exist.

In the biological research community there is also an interest in average fitness of individuals as opposed to solely the fittest individual [1]. The last section of this investigation of results therefore examines the average fitness across the individuals at each generation.

In Figure 5.40 the results produced when a 101 City TSP was used with 2% mutation and a population of 50 and the experiment was run for 500,000 generations. This graph illustrates the minimum tour length, or best result, produced by each experiment. The best result produced by each repair template (the lowest tour lengths in Figure 5.40) was investigated further. Figure

[1]Personal communication with Dr Susan Lolle in University of Waterloo, Ontario, Canada.

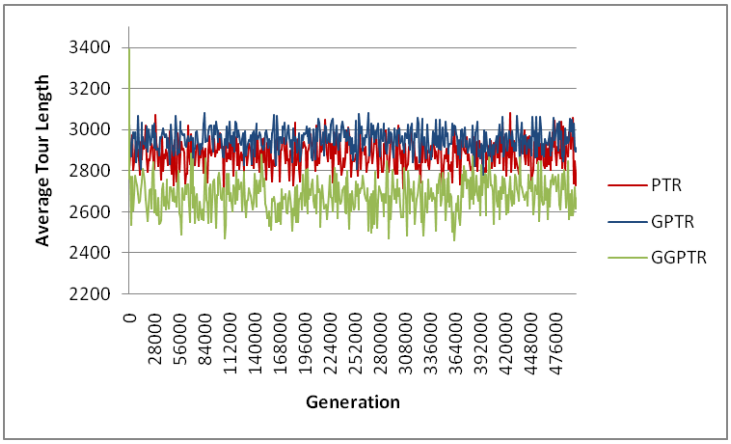

Figure 5.41: 101 City TSP Average Tour Length Found at Each Generation For the Best Tours of Each Ancestral Repair Template

5.41 shows the average fitness across the population at 1000 generation steps for each of these three results. Comparing the two graphs the first observation is that the order of efficiency of the ancestral repair templates is identical with great-grandparent performing best and grandparent performing worst (as expected for the 101 City TSP). When Mann Whitney statistical analysis was carried out on the results it was found that the average fitness across the great-grandparent template was less than that of the parent and grandparent template with p value of <0.0001 for both with a sample size of 500. The sample size is 500 as there were 500,000 generations and the average fitness across the population was calculated every thousand generations. The second observation is that the average fitness across the population is very different to the minimum tour length found at that generational milestone. When the experiments finish at 500,000 generations the average tour length

of the parent, grandparent and great-grandparent are 3382, 3384 and 3392 respectively while the minimum tour lengths found were 2070, 2138 and 1804 respectively. The statistical analysis in this section, however, shows that when taking minimum results or averages across the population the great-grandparent repair template outperforms the parent for this experimental set up. This suggests that regardless of whether the average or minimum tour length is being investigated non-Mendelian ancestral repair template outperforms the Mendelian ancestral repair template and also that diversity is maintained across the population as the average tour length is not close to the minimum tour length found.

Figure 5.42 shows the standard deviation of tour length over generations. While this graph does not give a precise picture statistical analysis spreads some light on the issue. Using the Mann-Whitney U test shows that the standard deviation of PTR <GGPTR with a confidence of $p<0.1271$. Further research could be carried out in this area to pinpoint spikes and ebbs in diversity by Mendelian and non-Mendelian templates.

Figure 5.43 shows the difference between the maximum and minimum fitness for one iteration at every 10,000 generations. When statistical analysis is carried out on these results the Mann-Whitney U test shows that PTR has a smaller absolute difference than GGPTR with a confidence of $p <0.0985$. This suggests that there is more diversity across the set of results produced by GGPTR than PTR.

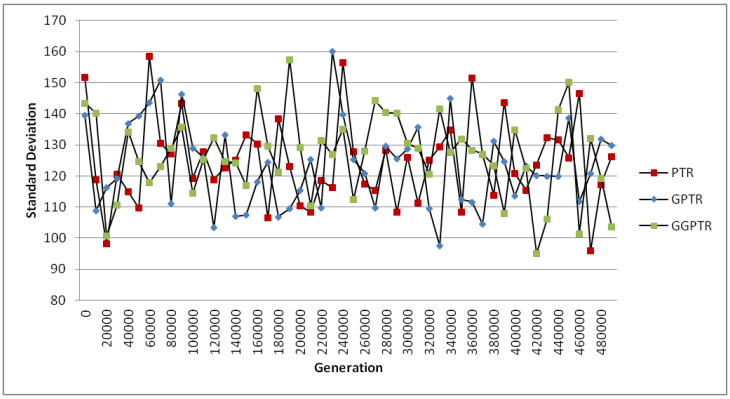

Figure 5.42: Standard Deviation of tour lengths produced using 101 City TSP

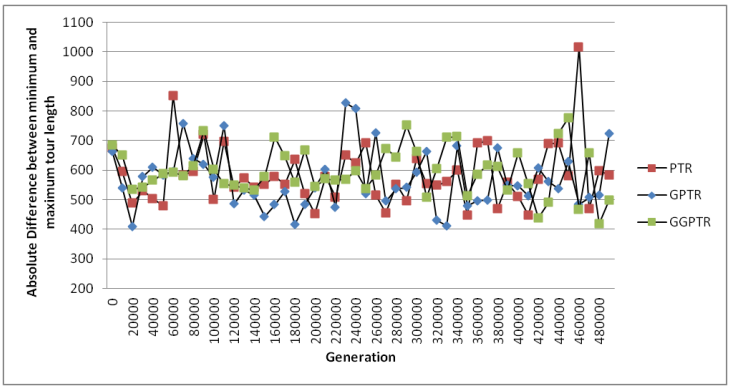

Figure 5.43: Absolute Difference between minimum and maximum tour length using 101 City TSP

5.7 Summary and Discussion

Throughout this thesis the effectiveness of Mendelian repair templates are compared to that of non-Mendelian ancestral repair templates under a variety of conditions. In this chapter the ancestral repair hypothesis presented in this thesis was tested within the context of EO to produce valid solutions to a standard constraint based problem. This ancestral repair mechanism was inspired by the controversial suggestion of a non-Mendelian repair mechanism in *Arabidopsis thaliana* (Lolle *et al.* 2005). It was controversially suggested that this plant uses non-Mendelian inheritance to enable it to use information that is not present in either parent but may be present in the grandparent, or a previous generation, to repair errors in the current individual. This controversial finding has led to the expansion of the analogy between EO and nature in order to include a repair mechanism in EO, as presented in this thesis. This repair mechanism, called GeneRepair has been implemented to enable EO to produce valid solutions to constraint based problems by giving it the tools to repair invalid individuals that break the problem constraints.

There are two main objectives underlying the results presented in this chapter: The first objective is to investigate whether non-Mendelian template repair can be used to enable EO to produce valid solutions to a standard constraint based problem. The second objective is to explore if the computer science side of the EO analogy can provide any evidence either supporting the non-Mendelian repair hypothesis (Lolle *et al.* 2005) or any evidence that appears to contradict this hypothesis. This results chapter is thus divided into two sections; Results that have implications for the Computer Science

Community (Section 5.5) and results that may have implications for the Biology Research Community (Section 5.6) even if these results are only analogically founded.

The chapter began by comparing the computational efficiency of Mendelian ancestral repair with the death penalty method and then went on to compare non-Mendelian ancestral GeneRepair to Mendelian ancestral GeneRepair. In order to measure these techniques to enable EO to produce valid solutions to a constraint based problem (TSP) was used. In Section 5.5.2 I showed that parent template GeneRepair is a far more efficient method of handling constraints than the death penalty. Parent based GeneRepair outperformed the death penalty for every mutation rate compared.

In Section 5.5.3 the use of the grandparent as a repair template was compared to the use of parent repair template. It was found that the grandparent produced stronger results than the parent, which led to the investigation of the use of the great-grandparent as a repair template. While Lolle's 2005 paper only focuses on grandparent, in this Section the great-grandparent showed itself to be the less efficient repair template to use with GeneRepair when compared to the grandparent but was also shown to be more efficient than the parent under the fixed direction fittest template condition.

At each ancestral level, (parent, grandparent or great-grandparent) there is a choice of two templates in the EO presented in this thesis. To investigate this choice using fittest of the two templates was compared randomly choosing between the two templates. While the results for this experiment were not as definite as other results produced, using the random great-grandparent

template (as opposed to the fittest) produced the strongest results, when parent, grandparent and great-grandparent were compared examining the use of the fittest and random templates.

For this reason there was further experiments carried out to examine the randomly selected ancestral template choice by comparing the three repair directions in conjunction with this parameter. It was found that repair carried out in a random and varying direction is more efficient than when carried out in a fixed right-to-left or left-to-right direction. It was also found that the great-grandparent produced stronger results when a random template was used as opposed to the fittest as did the grandparent when repair was carried out in a random direction. From these results it could be seen that non-Mendelian repair is most efficient when a randomly chosen template (as opposed to the fittest template) is used and repair is carried out in a random and varying direction. In the next set of results I examined the effect of population size on GeneRepair template choice and found that there is no effect of population size on the order of effectiveness of the repair templates. Throughout this thesis the effectiveness of Mendelian repair templates are compared to that of non-Mendelian ancestral repair templates under a variety of conditions. So far this thesis has shown that non-Mendelian templates outperform the parent template in a wide variety of situations.

This investigation continued to explore different mutation rates and it was found that the mutation rate that produced the best results for each template was (surprisingly) the same for each of the three different templates. This is important as it means that the choice of mutation rate is independent of the

choice of repair template.

Previous research (Mitchell 2007) has suggested that using a random repair template is the most effective method of template repair for EO. The use of random template repair (RTR) and this new GeneRepair technique was compared and found that for smaller problems the random repair template is more effective than ancestral repair templates. However for larger problems ancestral GeneRepair is far more effective than random template repair (As illustrated in Section 5.5.9). This is also a very significant finding because the focus of research tends towards a focus on larger problems, rather than smaller ones.

In order to make useful comparisons the majority of the results presented for Chapter 5 used the 101 city TSP. In order to show that the findings presented are independent of problem size I revisit some previously presented experiments using different problem sizes to show that the findings are not limited to the 101 city TSP. These results showed that GeneRepair is not sensitive to problem size. For 101 City TSP and smaller problems the great-grandparent repair outperformed the non-Mendelian parent repair while for 532 City TSP and larger problems the grandparent repair template outperformed the Mendelian parent repair template. Therefore, regardless of problem size, the non-Mendelian ancestral repair template outperformed the Mendelian ancestral repair template. Again this is a very significant finding and also echoes the controversial findings of Lolle *et al* (2005).

The comparison of two different selection methods to show that the results presented in this thesis are not dependent on selection method. Results show

that the truncation method was more effective with GeneRepair than the tournament method and also that the order of effectiveness of the repair templates is the same for both selection methods compared. This supports the hypothesis that GeneRepair is not sensitive to the selection method used. Therefore while using a different selection method may produce stronger results the "best "ancestral repair template remains the same.

While the results summarised so far fall into the Computer Science objective of my research (See Section 5.5), the next set of results are concerned with the Biological Field objective (See Section 5.6). I examined the advantages of fixing the first city of the population and found that fixing this city does not change the order of effectiveness of the repair templates. I also examined the effects of prohibiting self crossover in the EO. As the *Arabidopsis thaliana* allows self crossover I wanted to compare the effectiveness of the GeneRepair mechanism when this property is present and absent. I found that prohibiting self-crossover in the population produces weaker results than when self-crossover is allowed for great-grandparent repair. The mutation rate experienced by *Arabidopsis thaliana* in its natural environment is far lower than the rates used in evolutionary computation. For this reason the next experiment examined the performance of GeneRepair at very low mutation rates. I found that results produced by EO with very low mutation rates depend heavily upon the repair template used. I also found that the grandparent repair template is more effective than the parent and great-grandparent at very low mutation rates.

Overall, the results presented and discussed in this chapter, which are of

interest in both the academic field of Computer Science and Biology, support the use of GeneRepair with non-Mendelian repair templates to enable EO to produce valid solutions to constraint based problems.

Chapter 6

Repetition of Experiments Using CVRP Domain

6.1 Introduction

In Chapter 5 the results of a thorough investigation into the use of GeneRepair to enable EO to produce valid solutions to a constrained problem, were illustrated using the TSP as a problem domain. That chapter showed the results of many experiments into the parameters associated with GeneRepair as well as a wide variety of experiments to compare the efficiency of different ancestral templates. In this chapter it will be shown that these results are not specific to the TSP as many experiments described in Chapter 5 will be repeated with identical parameters and experimental setup using the Capacitated Vehicle Routing Problem (CVRP) instead of the TSP.

One of the advantages of GeneRepair is, unlike many other constraint

handling EO, it is not problem specific. In this chapter this property will be illustrated and supported by introducing the Capacitated Vehicle Routing Problem (CVRP). I will show the results of running similar experiments to those shown in Chapter 5 on this problem rather than the TSP. Importantly, the code for performing GeneRepair is unchanged between the TSP and CVRP problems and thus represents a good test of the potential generality of this permutation oriented version of ancestral template repair. This overcomes the disadvantage of problem specificity that is associated with many EO adaptations as shown in Chapter 2. The implementation of GeneRepair used in this thesis is specifically aimed at certain permutation problems (simple order based permutation problems) but with slight adaptation this implementation can be applied to a wide variety of problems. We again highlight that the ancestral repair algorithm itself originated in the domain of natural evolution, giving yet further weight to our claims for generality.

6.2 Structure of Chapter

This Chapter begins by introducing the Capacitated Vehicle Routing Problem. This problem will serve as an alternative to the TSP to investigate behaviour of GeneRepair on a different problem domain. Next the experimental setup for the results examined in this chapter is explained. This chapter then goes on to compare the use of Mendelian and non-Mendelian repair templates in a similar way to that of Chapter 5 except that the problem domain is different. The behaviour of GeneRepair when the population size is changed is then investigated. In a similar fashion to Section 5.5.5 the use of

GeneRepair at a variety of different mutation rates is examined. Finally the use of GeneRepair for different CVRP sizes is examined. Throughout this chapter we will be reminded of the results produced using the same parameters with the TSP and show how the results produced using the Capacitated Vehicle Routing Problem differ or match those shown in Chapter 5.

6.3 Capacitated Vehicle Routing Problem

The Capacitated Vehicle Routing Problem (CVRP) (Ralphs, Kopman, Pulleyblank, Trotter & Jr. 2001) is a combinatorial optimisation problem. It can be described as follows: n customers must be served from a unique depot. Each customer (or node) asks for a quantity qi of goods (i = 1,..., n) and a vehicle of capacity Q is available to deliver goods. Since the vehicle capacity is limited, the vehicle has to periodically return to the depot for reloading. The problem data provides *n-1* nodes, one depot and distances from the nodes to the depot as well as between the nodes. All nodes have demands which the depot can satisfy and the optimal result is the tour with minimal total length that satisfies the node demands without breaking the trucks capacity constraint.

6.4 Experimental Set-Up

As in Chapter 5 a standard population of 100 was used with 500,000 generations and a mutation rate of 2%. This experimental set up was used for all the experiments described in this chapter unless otherwise stated. The CVRP

problem data was obtained from the library of CVRP data presented in the TSPLIB collection by Heidelberg University in Germany (Reinalt 1991).

6.5 Ancestral Templates

In Sections 5.5.2 and 5.5.3 the use of ancestors as repair templates for the GeneRepair mechanism was examined. This technique repairs invalid individuals in the current population using the chosen ancestor of this individual to serve as the repair template (FitzGerald & O'Donoghue 2008). In this chapter we investigate whether the results in Chapter 5 relate to aspects that are specific to the TSP problem alone or are the results representative of the behaviour of ancestral repair. This technique mirrors the repair mechanism found in the *Arabidopsis thaliana* plant where, it has been suggested, the mutant plant repairs itself using information found in a generation previous to the parent (Lolle *et al.* 2005).

In this section the results produced from carrying out similar experiments to those of Chapter 5 are presented except that a different problem has been used. As previously stated, GeneRepair is not problem specific and has been implemented to produce valid solutions to any order based permutation problem that uses a simple linear representation. The results in this section were produced using the same technique on the CVRP problem. The same code was used with the same parameters and problem sizes. The reason for this mirror technique is to prove the problem independence property of GeneRepair (FitzGerald & O'Donoghue *in preparation*).

In Figure 6.1 the effectiveness of the three ancestral repair templates are

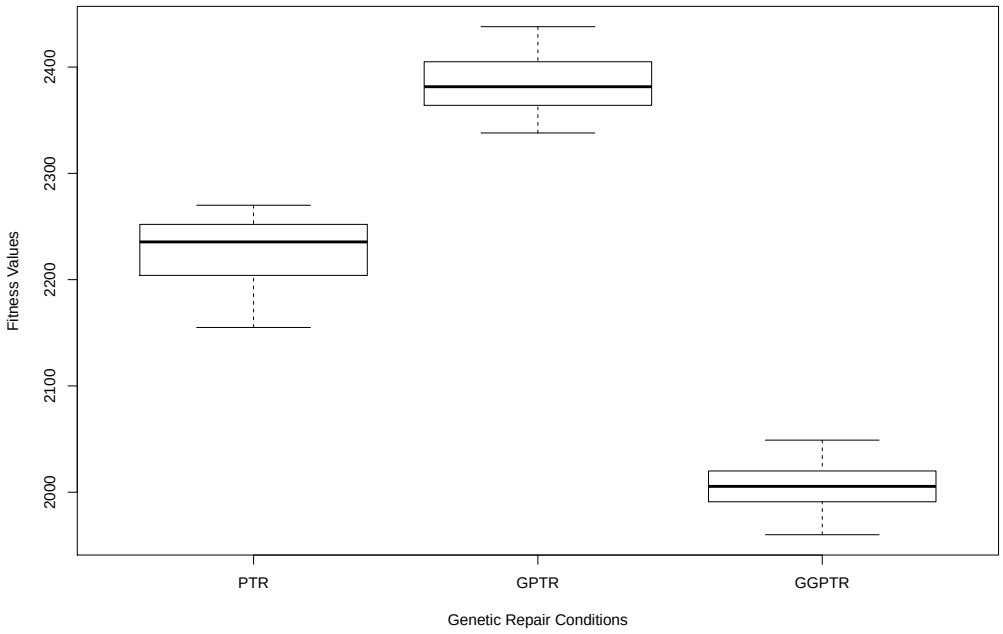

Figure 6.1: CVRP Ancestral GeneRepair Template Comparison

compared using the 101 City CVRP. Kruskal-Wallis statistical analysis indicated that there was a significant different among the three groups (PTR, GPTR and GGPTR) with H = 68.46 with df = 2 and p <0.0001. We can see from this graph that the Great-Grandparent repair template is most effective, with GGPTR <PTR and GPTR (p <0.0001). This supports the hypothesis from Section 5.5.3 that non-Mendelian ancestral templates can be more effective as GeneRepair templates than their Mendelian counterparts. The results mimic those of Section 5.5.12 (Figure 5.31)that non-Mendelian repair templates outperform their Mendelian counterparts for experimental conditions shown. This is not only applicable to the Computer Science community but also to the Biology Research community as it shows that when the con-

troversial suggestion of non-Mendelian repair mechanisms is implemented in an EO the non-Mendelian template outperforms the Mendelian template.

6.6 Variation of the Population Size

In this Chapter the same experiment were run on the CVRP problem with the same conclusion about ancestral templates found as those illustrated in Section 5.5.7. This was carried out to show that the GeneRepair technique can be applied to another (simple) order based permutation problem. In Figure 6.2 the results produced when the 101 node CVRP is used to compare three ancestral repair templates. The minimum result produced for each of the 52 repetitions of the experiment are shown. Kruskal-Wallis statistical analysis indicates a significant difference among the three groups with H =132.31, df = 2 with p <0.0001. The Mann-Whitney U test on these results further indicates that GGPTR <GPTR and PTR (p <0.0001). If we compare Figure 6.2 which used a population of 50 to Figure 6.1 with a population of 100 we can see that the order of effectiveness of the repair templates is identical. This supports the hypothesis in Section 5.5.7 which states that the choice of GeneRepair template is not impacted by population size. While in Section 5.5.7 this hypothesis was supported using the TSP, the results depicted in Figure 6.2 show that this hypothesis is not TSP specific as it has also been supported using the CVRP.

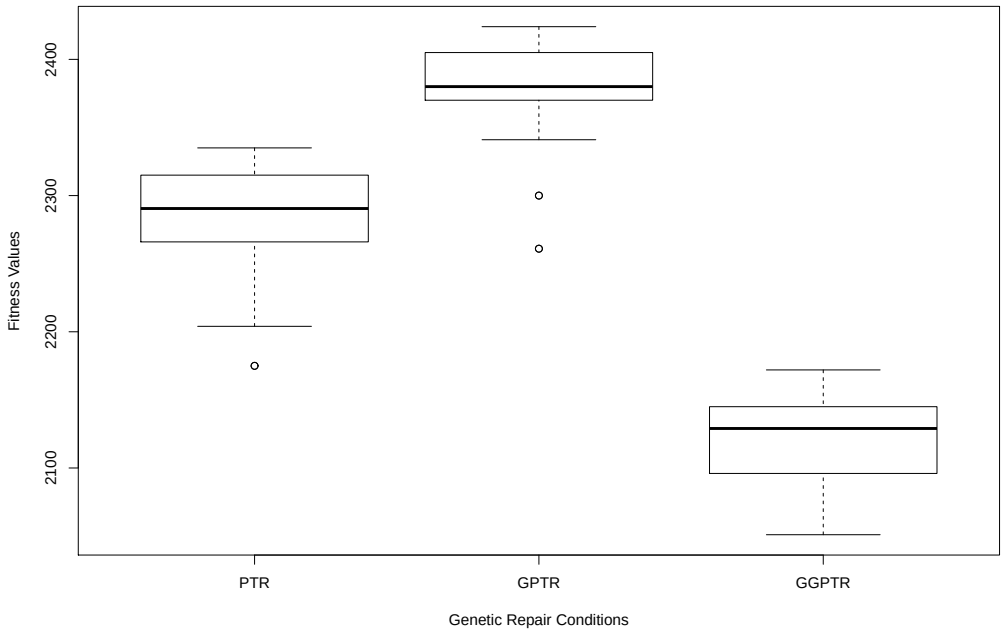

Figure 6.2: A comparison of the three ancestral repair templates for 101 city CVRP using a population of 50

6.7 Comparison Across Mutation Rates

In Section 5.5.5 the effect of the repair template on the optimal mutation rate for the specific problems was investigated using the TSP. It was found that the choice of repair template does not affect the optimal mutation as each of the repair templates converged to the same optimal mutation rate for the TSP. In this Section this finding is investigated using the CVRP. The results presented in this section were produced by exploring the (average) fitness produced across a variety of mutation rates between 2% and 0.1%. This experiment used for the 101 node CVRP with a population of 100 run

for 500,000 generations. This experiment mirrors the experiment illustrated in Figure 5.13 however the results shown in Figure 6.3 were produced using the CVRP as opposed to the TSP.

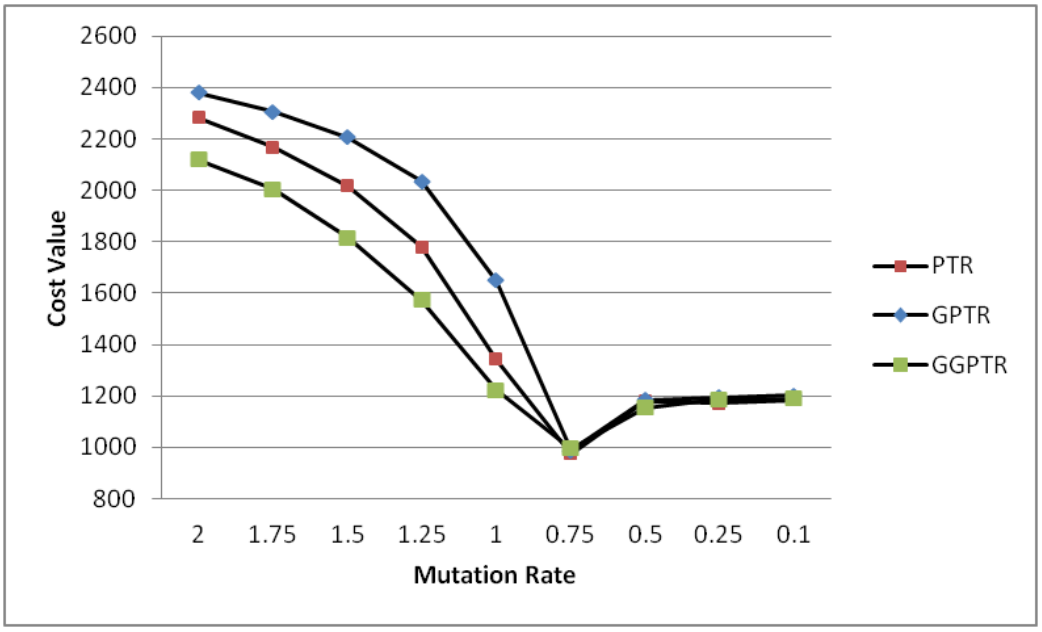

Figure 6.3: A comparison of the three ancestral repair templates across a range of Mutation Rates for 101 node CVRP

Figure 6.3 shows that the optimal mutation rate occurs at 0.75% mutation - for each of the three ancestral repair strategies. This supports the hypothesis from Section 5.5.5 which states the choice of repair template does not affect the optimal mutation rate for the specific problem. In fact, for a similarly sized TSP problem (eil101) the optimal mutation rate was also found to be 0.75% for each of the repair templates. This was to be expected as the TSP is a 101 City problem and the CVRP is a 101 node problem with the added complexity of the depot property. We can see from the results

presented that this property of GeneRepair is not problem specific. This continues to support the thesis that GeneRepair is not problem specific.

As in the results presented in Section 5.5.5, Figure 6.3 shows that the difference between the repair template effect is stronger for mutation rates above the optimal (0.75%) and there is little difference at the optimal and below it. This finding echoes that of Section 5.5.5 which again shows that the results produced by GeneRepair are not problem specific (to the TSP).

6.8 Different Problem Size

In Section 5.5.10 we saw that the ancestral effect of GeneRepair for the TSP is not specific to problem size. In Figure 6.4 this experiment was repeated using a 51 node CVRP as opposed to the 101 City CVRP used in previous experiments in this Chapter. Kruskal-Wallis analysis indicates a significant difference among the three sets of results with H = 47.64, df = 2 with p <0.0001. The Figure shows that for a problem of this size the great-grandparent repair template produces the strongest results, with GGPTR <GPTR and PTR (p <0.0001). This graph echoes the results discussed in Section 5.5.10, the finding that the order of efficiency of ancestral repair templates is independent of problem type but specific to problem size with great-grandparent repair producing the strongest results for small problems (101 City TSP and less). This section shows that this property of GeneRepair is not specific to the TSP. While the properties of GeneRepair are not specific to problem size they are also not specific to problem domain of the TSP as the results illustrated in Figure 6.4 were produced using the 51 node

CVRP.

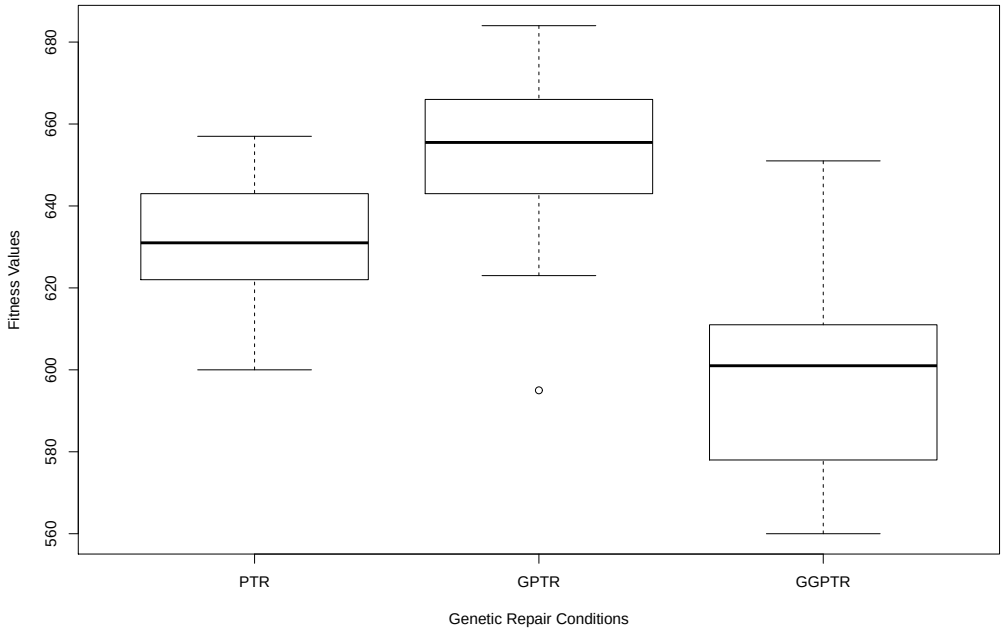

Figure 6.4: A comparison of the three ancestral repair templates used with GeneRepair on a 51 node CVRP with Mutation Rate of 2%

6.9 Conclusion

In Chapter 2 it was shown that a number of EO implementations to handle constraints are problem specific. This chapter addresses this issue by extending some previous research on GeneRepair. We investigate whether GeneRepair applicable to other constraint problems (in addition to the TSP investigated in a previous chapter) and whether our implementation of GeneRepair is applicable to other order based permutation problems. In order to investi-

gate whether GeneRepair is TSP specific a number of the experiments from Chapter 5 were repeated in this Chapter using a different problem domain. These investigations also investigate the ancestral template effect found in Chapter 5, that generally the non-Mendelian repair templates outperformed the Mendelian alternative. The problem domain used in this Chapter is the Capacitated Vehicle Routing Problem (CVRP) [1] which is the problem of routing multiple vehicles around n nodes. Each node has a demand and this demand is met by the vehicle returning to the depot in order to fill the node. Each vehicle also has an associated capacity limit. The optimal results is the minimum tour length that satisfies all node demands and does not violate the vehicles finite capacity.

The results indicate the same order of effectiveness of the ancestral repair templates as when compared with the TSP results from the previous chapter. This means that when the great-grandparent template was most effective for a certain set of parameters using the TSP then it was also most effective for the CVRP. This indicates that the same choice of repair template should be made for both the TSP and the CVRP permutation problems. The results also showed that the optimal mutation rate (of those tested) was the same for the same size TSP and CVRP. This indicates that the optimal mutation rate for the 51 City TSP is equal to the optimal mutation rate for the 51 node CVRP. The order of efficiency of the ancestral repair templates was also the same for the both problem types. While this illustrates that the same mutation rate can be used in conjunction with the repair template for

[1] All CVRP Data available online at http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/vrp/

both the TSP and CVRP the conclusion from these sets of results is that the order of efficiency of the repair templates is the same at each mutation rate. Specifically, the best results are reliably produced by using the great-grandparent as the repair template - this template producing better results than either the parent or grandparent templates. This Chapter has shown that the application of ancestral repair templates and GeneRepair in not limited to the TSP. The results presented in Chapter 5 are therefore not isolated instances where the GeneRepair technique is successful but rather a strong indication of how the technique will work with other constraint based problems.

Chapter 7

Conclusion

7.1 Introduction

Evolutionary Algorithms (EA) are an excellent technique for solving difficult problems. Where EA are challenged however is when they are used to solve constraint based problems. In addition to presenting large and complex problem spaces, these problems call for a number of constraints to be satisfied in a given solution and this can cause difficulty for EA. The reason that standard EA are not suited to solving this type of problem is that they create a wide and varying range of solutions and lack a method to ensure that these solutions obey the problem constraints. EA create diverse solutions and avoid reaching a local maxima by keeping the population spread across the feasible search space. Often the fitness function is inapplicable outside of the feasible search space and the feasible search space may be small in comparison with the overall search space. In order to produce so-

lutions that obey specific problem constraints the generic EA (as shown in Section 2.2.2) must be adapted. This adaptation can be done in a variety of ways. At present there are 4 main schools of thought on this subject. The EO can be tweaked to use modified operators, it can be adapted to use a pareto-optimal strategy, it can incorporate a penalty system or it can use a form of repair on invalid solutions. Each of these methods are explained and reviewed in Chapter 2. While each of these fields enable EO to handle constraints their disadvantages include non-biologically inspired implementation, problem specificity, complex implementations and difficulty in result reproduction based on information provided.

This thesis explored a biologically inspired method to enforcing constraints within an EO. This method falls into the repair category of techniques for handling constraints within EA. This method called GeneRepair is a repair mechanism to enable EO to produce valid solutions to constraint based problems. GeneRepair was formed by strengthening the existing analogy between EO and a recently postulated theory (Lolle *et al.* 2005) for natural evolution. In 2005 a non-Mendelian repair mechanism was suggested in the *Arabidopsis thaliana* plant (Lolle *et al.* 2005). It is suggested that this controversial repair mechanism uses information that is not present in the parent to repair its own genetic defects. This non-Mendelian inheritance incurred both support (Chaudhury 2005), (Ray 2005), (Weigel & Jürgens 2005) and doubt (Mercier *et al.* 2008), (Peng *et al.* 2006) in the field of biology. Analogy is the cognitive process of transferring information or meaning from a particular subject (the source) to another subject (the target). In this the-

sis the analogy between nature and computer science was used, in particular the analogy that can be drawn between the proposed repair mechanism of the *Arabidopsis thaliana* plant and the derived repair mechanism in Evolutionary Optimisation. In this analogy the source is nature and more specifically the *Arabidopsis thaliana* plant and the target is EO. It should be noted that although work is ongoing, at the time of writing there has still been no independently published work confirming this finding. This possible biological breakthrough was used in this thesis to extend the analogy between biology and EO. This thesis compared the effectiveness of a number of different ancestral repair templates to not only each other but also to a penalty approach. This thesis also went on to compare the effectiveness of GeneRepair on a second problem domain.

The findings in this thesis can provide support to the suggestions made by Lolle *et al* (Lolle *et al.* 2005) as non-Mendelian ancestral repair templates were shown to be more effective than their Mendelian counterparts for a wide variety of experiments carried out (See Chapter 5 and Chapter 6).

7.2 Summary of Main Results

In this thesis the effectiveness of Mendelian repair templates are compared to that of non-Mendelian ancestral repair templates under a variety of conditions. In Chapter 5 it was shown that parent template GeneRepair is a far more efficient method of handling constraints than the death penalty. Parent based GeneRepair outperformed the death penalty for every mutation rate compared. It was then shown that the grandparent template produced

stronger results than the parent template, which led to the investigation of the use of the great-grandparent as a repair template. The great-grandparent was shown to be the less efficient repair template to use with GeneRepair on smaller problems when compared to the grandparent but more efficient than the parent under the fixed direction fittest template condition. When the comparison of conducting repair in the three directions was shown it was found that repair carried out in a random and varying direction is more efficient than when carried out in a fixed right-to-left or left-to-right direction. It was also found that the great-grandparent produced stronger results when a random template was used as opposed to the fittest as did the grandparent when repair was carried out in a random direction. From these results it could be seen that non-Mendelian repair is most efficient when a randomly chosen template (as opposed to the fittest template) is used and repair is carried out in a random and varying direction. The effect of population size on the GeneRepair template choice was examined and it was found that population size does not affect of the order of effectiveness of the repair templates. When mutation rates were explored it was found that the mutation rate that produced the best results for each template was (surprisingly) the same for each of the three different templates which means that the choice of mutation rate is independent of the choice of repair template. The use of random template repair (RTR) and this new GeneRepair technique was compared and found that for smaller problems the random repair template is more effective than ancestral repair templates. However for larger problems ancestral GeneRepair is far more effective than random template repair (As

illustrated in Section 5.5.9). This is also a very significant finding because the focus of research tends towards a focus on larger problems, rather than smaller ones. The investigation went on to show that GeneRepair is not sensitive to problem size. For 101 City TSP and smaller problems the great-grandparent repair outperformed the non-Mendelian parent repair while for 532 City TSP and larger problems the grandparent repair template outperformed the Mendelian parent repair template. Therefore, regardless of problem size, the non-Mendelian ancestral repair template outperformed the Mendelian ancestral repair template. When selection methods were compared results showed that the truncation method was more effective with GeneRepair than the tournament method and also that the order of effectiveness of the repair templates is the same for both selection methods compared. I also examined the advantages of fixing the first city of the population and found that fixing this city does not change the order of effectiveness of the repair templates. When self crossover was prohibited it was found that the population produces weaker results than when self-crossover is allowed for great-grandparent repair. When we examined the performance of GeneRepair at very low mutation rates it was found that results produced by EO with very low mutation rates depend heavily upon the repair template used. It was also found that the grandparent repair template is more effective than the parent and great-grandparent at very low mutation rates.

Overall, the results presented and discussed in this thesis, which are of interest in both the academic field of Computer Science and Biology, support the use of GeneRepair with non-Mendelian repair templates to enable EO to

produce valid solutions to constraint based problems. We went on to examine similar experiments carried out using a different problem domain, the CVRP as opposed to the TSP, and found that the results produced were similar and that the order of effectiveness (and thus the choice) of repair template was the same.

7.3 Future Work

GeneRepair using non-Mendelian inheritance is an exciting new technique to enable EO to produce valid solutions to constraint based problems. While previous findings have been published on GeneRepair (FitzGerald & O'Donoghue *in preparation*), (FitzGerald & O'Donoghue 2010), (FitzGerald & O'Donoghue 2008), (FitzGerald *et al.* 2009), (FitzGerald & O'Donoghue 2009), (Hatton, O'Donoghue & FitzGerald 2010), (Hatton & O'Donoghue 2011), (Mitchell 2007) this is the first PhD thesis which concentrates on comparing Mendelian and non-Mendelian repair templates under a wide variety of conditions. This thesis investigates the use of GeneRepair and suggests findings that relate to the fields of both Computer Science and Biology. This thesis has opened up new possibilities for future researchers to explore. The analogy that inspired my research is potentially a developing one, with ongoing work by Dr Susan Lolle and a number of other active research groups involved in plant genetics. This ongoing research into the natural inspiration behind this thesis may enable future research which will build upon the foundation created by this thesis.

7.3.1 Other Problem Sets

In this thesis I investigated the different properties of GeneRepair as applied to solving the TSP. In order to illustrate the generality of GeneRepair I went on to show similar properties and findings when the CVRP was used as the problem domain. While this shows that the properties of GeneRepair are not specific to the TSP it also suggests that GeneRepair could be used with other combinatorial optimisation problems. Future work might even include investigating the use of GeneRepair on non-combinatorial optimisation problems. This would widen the possible uses of GeneRepair and would be of interest to a wide community of researchers in the field of EO.

7.3.2 Multiple Constraint Problems

The experiments presented in Chapter 5 and Chapter 6 used a single constraint problem (TSP and CVRP). The method of GeneRepair could also be used to find valid solutions to multiple constraint problems. One suggestion of how to conduct GeneRepair as part of an EO on a multiple constraint problem would be to implement a small change to the algorithm. The GeneRepair step of the EO would be carried out on each constraint until all constraints are satisfied. The next individual would then undergo GeneRepair. A mandatory repair of hard constraints may be imposed with a percentage repair of soft constraints but further research into multiple constraint problems is necessary.

7.3.3 Beyond Permutation Problems

Thus far I have investigated and hypothesised how GeneRepair in conjunction with EO could produce valid solutions to constrained permutation problems. Further to this, GeneRepair could also be used to find solutions to the "Graph Colouring Problem"where the objective is to find the minimum number of colours necessary to fill a map or graph with the constraint that no two touching boundaries may be of the same colour. This problem could be represented as a matrix of integers where each integer represents a colour. The integers would start at 1 and continue increasing as each colour is added. When a colour is found to be redundant the highest integer would be removed and the matrix corrected accordingly. GeneRepair could be carried out to ensure that no touching elements in the matrix (diagonal, vertical or horizontal) are the same integer. The fittest solution would be the one with the lowest maximum integer used. Further research into this algorithm could be done to enable GeneRepair with EO to produce valid solutions to this problem.

7.3.4 Further *Arabidopsis thaliana* Study

The non-Mendelian repair mechanism suggested (Lolle *et al.* 2005) is highly controversial and the subject to ongoing discussion in the academic field of biology (Mercier *et al.* 2008), (Peng *et al.* 2006) as existing results have yet to be independently confirmed. As further research is carried out and more light is shed on this repair mechanism an adaptation to the GeneRepair mechanism to strengthen the analogy with biology (See Chapter 3) could be made. This

adapted implementation may provide a stronger repair mechanism for the field of Computer Science and at least some evidence of support for the findings made in the field of biology.

7.3.5 Further Ancestral Repair Research

There is currently research being carried out into the use of ancestral repair templates that are between one and many thousands of generations old (Hatton & O'Donoghue 2011). This research may lead to further collaboration with Dr Susan Lolle if the natural inspiration for this repair mechanism (Lolle *et al.* 2005) is independently supported.

7.3.6 Further PTR Research

In the findings presented in this thesis non-Mendelian repair templates (grandparent or great-grandparent depending on problem size) were shown to outperform their Mendelian (parent) counterparts for the the conditions investigated. While non-Mendelian templates outperformed the parent template, the parent template usually ranked second out of the three templates investigated. This raises questions and opens the door to further research into whether the parent template is the "safer"option to use as, while it is not usually the best, it is also not usually the worst option, out of the three options investigated.

7.4 Summary of Findings on GeneRepair

This body of research establishes GeneRepair as a technique in the repair category of mechanisms to enable EO to produce valid solutions to constraint based problems. This thesis indicates that extending the analogy between EO and nature can enable the EO to produce valid solutions to constraint based problems in a biologically inspired manner. This method called GeneRepair is easily implemented and is meta heuristic. Inspired by the repair mechanism suggested in the *Arabidopsis thaliana* plant GeneRepair uses an ancestral repair template to repair errors or constraint violations in the current population of the EO. Investigating this method led to a number of conclusions.

1. **GeneRepair enables EO to produce valid solutions to a constraint based problem**

   The first conclusion of this thesis is that ancestral driven repair improved the functionality of EO on the TSP. This ancestral GeneRepair improved EO by enabling it produce valid solutions to constraint based problems.

2. **Non-Mendelian repair templates,in conjunction with GeneRepair, produce superior results to the Mendelian template**

   The second conclusion of this research is that non-Mendelian repair templates are frequently more effective than their Mendelian counterparts. We saw that for smaller problems (101 City TSP and smaller) the great-grandparent outperformed the parent and grandparent template

while for problems larger than this the grandparent template outperformed the parent and great-grandparent. This finding indicates than non-Mendelian repair templates can outperform their Mendelian counterparts.

3. **GeneRepair is not problem specific**

   The third conclusion of this thesis is that ancestral driven repair is improves EO on the two chosen problem domains. This finding illustrates the non-problem specificity of GeneRepair as it is not limited to the TSP domain. Ancestral GeneRepair has been evaluated on two permutation problem. It can in principle be applied to other permutation problems with relatively minor modification to the current implementation.

4. **GeneRepair outperforms RTR for large problems**

   The next conclusion of this thesis is that ancestral GeneRepair works well across varied mutation rates, with typically a better performance than the Random Template Repair (RTR) (which has been favoured in the past (Lichtblau 2002) (Mitchell 2007). In the experiments carried out for this body of research GeneRepair was only outperformed by RTR under a limited number of conditions. It was beaten by RTR on small problem sizes but at larger problem sizes it outperformed RTR.

5. **Favouring the Fitter of the Ancestral Repair Templates does not produce superior results**

It was found that using a randomly chosen template (of the two available) produces superior results to choosing the fittest of the two templates available. ‘

An aim of this research was to compare Mendelian and non-Mendelian repair templates to produce valid solutions to constraint based problems. While the results do not show a clear winner between the grandparent and great-grandparent templates the parent template is generally beaten by one of the two. In Chapter 5 there are only a small number of conditions where the parent template produces the best results but this is only for small problems at the optimal mutation rate. For the large portion of the results the non-Mendelian template outperformed its Mendelian counterpart. In Chapter 5 we saw that the Ancestral repair strategy continues to perform well, even under more biologically inspired conditions such as reduced mutation rates and much larger problem sizes that correspond to the larger genome found in natural organisms. The success of ancestral GeneRepair may be attributed to the fact that it uses an evolved solution, that has survived the selection process, as a basis to repair a more evolved but invalid solutions which may otherwise be eliminated from the population.

In Chapter 6 it was shown that the effectiveness of this non-Mendelian GeneRepair mechanism is not limited to the TSP. In Chapter 6 the problem domain was changed to the CVRP and the non-Mendelian repair templates continued to outperform their Mendelian counterparts. This thesis has illustrated how GeneRepair could be applied to permutation problems using content-based repair. It has also been shown how GeneRepair could be ap-

plied to combinatoric optimisation problems. GeneRepair could be implemented to solve combinatoric optimisation problems by making use of different error signatures. Looking at GeneRepair with a broader view it could be used to solve general constraint problems. We again point out that the inspiration for ancestral repair lies in the *Arabidopsis thaliana* plant. In this thesis it was used to produce valid solutions to the TSP and the CVRP but with slight adaptation of the implementation it could in theory be used to solve a wide range of numeric or Genetic Programming Problems. In Genetic Programming a similar technique proposed by Conor Ryan *et al* (Murphy, Ryan & Howard 2007) using run-time transferable libraries. Run-time transferable libraries allow the transfer of useful libraries of evolved solutions - between "independent"runs of their Genetic Programming system. This technique could be adapted to use non-Mendelian ancestral information instead.

The second objective of this thesis was to conduct stress or reliability testing of the mechanism by comparing the use of Mendelian and non-Mendelian repair mechanisms in an EO. It should be again pointed out that while the objective was essentially biological, the analysis was conducted exclusively at the algorithmic level. Thus the aim was to assess the effectiveness of the ancestral repair hypothesis. Our analogy extension and evaluation approach has shed light on non-Mendelian crossover for researchers in the field of biology(Hopkins *et al.* 2011). This has also been discussed during ongoing collaboration (which included a lab internship) with Dr Susan Lolle, Biology Department of University of Waterloo. By examining the repair process from a different perspective this research has provided tentative support for

the non-Mendelian inheritance repair mechanism. Our results suggest that for larger genomes and when populations lack diversity, then non-Mendelian repair templates greatly outperform their Mendelian (and death penalty) alternatives. While this support is cross-disciplinary, and so only suggestive, it may suggest new avenues of research for those working in the biological field.

The principal finding of this thesis is that non-Mendelian ancestral GeneRepair inspired by the *Arabidopsis thaliana* plant enables EO to produce valid solutions to constraint based problems. This non-Mendelian ancestral GeneRepair process outperforms its Mendelian counterpart across a wide number of different parameters. This finding lends support to the suggestion that non-Mendelian repair occurs in nature as it may be more effective than the Mendelian alternative.

This is the beginning of the road for ancestor driven GeneRepair. This thesis has illustrated a thorough investigation into the technique. A wide number of experiments have been discussed which test and examine the different properties including the ancestry of the repair template, population size, problem size, mutation rate, number of generations and fitness of template utilising two different problem domains. Conclusions have been drawn on parameter effectiveness and these suggest which ancestral repair templates are most effective. This thesis opens up new vistas on evolutionary computation and makes novel contributions to the study of evolution.

Bibliography

Ahn, C. W. & Ramakrishna, R. S. (2002), 'A genetic algorithm for shortest path routing problem and the sizing of populations', *IEEE Trans. Evolutionary Computation* **6**, 566–579.

Arroyo, C. (2002), 'A parallel repair genetic algorithm to solve the unit commitment problem', *IEEE Transactions on Power Systems* **17**(4).

Atmar, W. (1994), 'Notes on the simulation of evolution', *IEEE Transactions on Neural Networks* **5**(1), 130–147.

Aubusson, P. J., Harrison, A. G. & Ritchie, S. M., eds (1996), *Metaphor and Analogy in Science Education*, Springer.

Bäck, T. (1992), 'Genesys 1.0. software distribution and installation notes', *Systems Analysis Research Group* .

Bäck, T. (1993), Optimal mutation rates in genetic search, *in* 'Proceedings of the fifth International Conference on Genetic Algorithms', Morgan Kaufmann, pp. 2–8.

Bäck, T., Schütz, M. & Khuri, S. (1995), A comparative study of a penalty function, a repair heuristic and stochastic operators with the set-covering problem, *in* J.-M. Alliot, E. Lutton, E. M. A. Ronald, M. Schoenauer & D. Snyers, eds, 'Artificial Evolution', Vol. 1063, Springer, pp. 320–332.

Bean, J. C. (1992), Genetics and random keys for sequencing and optimization, Technical Report TR 92-43, Dept. of Industrial and Operations Engineering, University of Michigan.

Bean, J. C. & Hadj-Alouane, A. B. (1992), A dual genetic algorithm for bounded integer programs, Technical Report TR 92-53, Dept. of Industrial and Operations Engineering, University of Michigan.

Blickle, T. & Thiele, L. (1996), 'A comparison of selection schemes used in evolutionary algorithms', *Evol. Comput.* **4**, 361–394.

Campbell, K. H. S., McWhir, J., Ritchie, W. A. & Wilmut, I. (1996), 'Sheep cloned by nuclear transfer from a cultured cell line', *Nature* **380**, 64–66.

Camponogara, E. & Talukdar, S. N. (1997), 'A genetic algorithm for constrained and multiobjective optimization', *3rd Nordic Workshop on Genetic Algorithms and Their Applications (3NWGA)* pp. 46–92.

Chaudhury (2005), 'Hothead healer and extragenomic information', *Nature* **437**.

Coello Coello, C. A. (1999), 'Use of a self-adaptive penalty approach for engineering optimization problems', *Computers in Industry* .

Coello Coello, C. A. (2002), 'Theoretical and numerical constraint handling techniques in evolutionary algorithms: A survey of the art', *Computer Methods in Applied Mechanics and Engineering* **191**, 1245–1287.

Coello Coello, C. A. & Nacional, L. (1999), 'Treating constraints as objectives for single-objective evolutionary optimization', *Engineering Optimization* **32**, 275–308.

Craenen, B. G. W., Eiben, A. E. & Marchiori, E. (2001), How to handle constraints with evolutionary algorithms, *in* 'Chapmann & Hall/CRC Press, Ch 10', Chapman & Hall/CRC, pp. 341–361.

Darwin, C. (1872), *Origin of the Species*, 6 edn, Murray.

Dasgupta, D. & Michalewicz, Z. (1997), *Evolutionary Algorithms in Engineering Applications*, Springer.

Davidor, Y. (1989), Analogous crossover, *in* 'International Conference on Genetic Algorithms', pp. 98–103.

Davis, L. (1991), *Handbook of Genetic Algorithms*, Van Nostrand Reinhold.

Deb, K. (1998), An efficient constraint handling method for genetic algorithms, *in* 'Computer Methods in Applied Mechanics and Engineering', pp. 311–338.

DeGaris, H. (1990), Genetic programming - building artificial nervous systems with genetically programmed neural network modules, *in* R. Porter & B. Mooney, eds, '7th International Conference on Machine Learning', Morgan Kaufmann Publishers Inc., pp. 132–139.

Eiben, A. E. (2001), Evolutionary algorithms and constraint satisfaction: Definitions, survey, methodology, and research directions, *in* 'Theoretical Aspects of Evolutionary Computing', Springer-Verlag, pp. 13–58.

FitzGerald, A. & O'Donoghue, D. (2009), Investigating the influence of population and generation size on generepair templates, *in* 'proceedings of the China-Ireland information and communications technologies conference', Dept. of Computer Science, National University of Ireland, Maynooth, Co. Kildare, Ireland, pp. 252–255.

FitzGerald, A. & O'Donoghue, D. P. (2008), 'Genetic repair for optimization under constraints inspired by *arabidopsis thaliana*', *Parallel Problem Solving From Nature* pp. 399–408.

FitzGerald, A. & O'Donoghue, D. P. (2010), Biologically inspired non-mendelian repair for constraint handling in evolutionary algorithms., *in* 'GECCO (Companion)'10', pp. 1817–1824.

FitzGerald, A. & O'Donoghue, D. P. (*in preparation*), 'Naturally inspired repair: Ancestral repair templates and *arabidopsis thaliana*'.

FitzGerald, A., O'Donoghue, D. P. & Liu, X. (2009), Genetic repair strategies inspired by *arabidopsis thaliana*, *in* 'AICS', pp. 61–71.

Fogel, D. (1994), 'An introduction to simulated evolutionary optimization', *IEEE Transactions on Neural Networks* **5**(1), 3–14.

Fogel, D. B. (1995), *Evolutionary Computation - Towards a New Philosophy of Machine Intelligence*, IEEE.

Gawrylewski, A. (2008), 'Mendel upended?', *The Scientist* **22**(2), 30.

Gentner, D. (1983), 'Structure-mapping: A theoretical framework for analogy', *Cognitive Science* **7**(2), 155–170.

Glover, F. & Kochenberger, G. A. (1995), Critical event tabu search for multidimensional knapsack problems, *in* 'Proceedings of the International Conference on Metaheuristics for Optimization', Kluwer Academic Publishers, Dordrecht, Netherlands, pp. 113–133.

Goldberg, D. E. & Lingle, Jr., R. (1985), Alleleslociand the traveling salesman problem, *in* 'Proceedings of the 1st International Conference on Genetic Algorithms', L. Erlbaum Associates Inc., Hillsdale, NJ, USA, pp. 154–159.

Handa, H., Watanabe, K., Katai, O., Konishi, T. & Baba, M. (1999), 'Coevolutionary genetic algorithm for constraint satisfaction with a genetic repair operator for effective schemata formation'.

Hatton, D., O'Donoghue, D. & FitzGerald, A. (2010), Investigating the effect of stochastic replacement rates on generepair templates, *in* 'Proceedings of China-Ireland International Conference on Information and Communications Technologies', pp. 81 – 85.

Hatton, D. & O'Donoghue, D. P. (2011), Explorations on template-directed genetic repair using ancient ancestors and other templates, *in* 'Proceedings of the 13th annual conference companion on Genetic and evolution-

ary computation', GECCO '11, ACM, New York, NY, USA, pp. 325–332.

Hinterding, R. & Michalewicz, Z. (1998), Your brains and my beauty: parent matching for constrained optimisation, *in* 'Proc. of the 5th Int. Conf. on Evolutionary Computation', pp. 810 –815.

Holland, J. H. (1975), *Adaptation in Natural and Artifical Systems*, University of Michigan Press.

Holyoak, K. J., Novick, L. R. & Melz, E. R. (1994), 'Component processes in analogical transfer: Mapping, pattern completion, and adaptation. analogical connections.', *Analogical connections, Advances in connectionist and neural computation theory* **2**, 113–180.

Homaifar, A., Qi, C. X. & Lai, S. H. (1994), 'Constrained optimization via genetic algorithms', *Transactions of The Society for Modeling and Simulation International* **62**, 242–253.

Hopkins, M., Chang, P., Lai, D., Doerr, M. & Lolle, S. (2011), 'De novo genetic variation revealed in somatic sectors of single *Arabidopsis* plants'. available at `http://www.plant-a-seed.com/#research-paper/`.

Houck, C. R., Joines, J. A., Kay, M. G. & Wilson, J. R. (1997), 'Empirical investigation of the benefits of partial lamarckianism'.

hyun Han, K. & Kim, J.-H. (2000), Genetic quantum algorithm and its application to combinatorial optimization problem, *in* 'in Proc. 2000

Congress on Evolutionary Computation. Piscataway, NJ: IEEE', Press, pp. 1354–1360.

Ishibuchi, H. & Narukawa, K. (2005), Spatial implementation of evolutionary multiobjective algorithms with partial lamarckian repair for multiobjective knapsack problems, *in* 'Hybrid Intelligent Systems, 2005. HIS '05. Fifth International Conference on', p. 6 pp.

Jimnéz, F. & Verdegay, J. L. (1999), Evolutionary techniques for constrained optimization problems, *in* '7th European Congress on Intelligent Techniques and Soft Computing (EUFIT'99)', Springer-Verlag, Aachen, Germany.

Khuvung, L. P. (2009), 'Learn from nature', *Nagaland Post* .

Kimbrough, S. O. & Wood, D. H. (2007), 'On gray-coded binary representation for supporting a (repair-by-interpolation) genetic operator for constrained optimization problems', *CEC* pp. 1141–1148.

Kowalczyk, R. (1997), Constraint consistent genetic algorithms, *in* 'Proceedings of the 1997 IEEE Conference on Evolutionary Computation', IEEE Press, Indianapolis, USA, pp. 343–348.

Lichtblau, D. (2002), Discrete optimization using mathematica, *in* N. Callaos, T. Ebisuzaki, B. Starr, J. Abe & D. Lichtblau, eds, 'World Multi Conference on Systemics, Cybernetics and Informatics', Vol. 16, International Institute of Informatics and Systemics, pp. 169–174.

Liepins, G. & Potter, W. (1991), 'A genetic algorithm approach to multiple-fault diagnosis', *Handbook of Genetic Algorithms* pp. 237–250.

Lin, S. & Kernighan, B. W. (1973), 'An Effective Heuristic Algorithm for the Traveling-Salesman Problem', *Operations Research* **21**(2), 498–516.

Lolle, S. J., Victor, J., Young, J. & Pruitt, R. (2005), 'Genome-wide non-mendelian inheritance of extra-genomic information in arabidopsis', *Nature* **434**(1), 505–509.

Luger, G. F. (2002), *Artificial Intelligence Structures and Strategies for Complex Problem Solving*, fourth edn, Addison-Wesley.

Luke, S. (2009), *Essentials of Metaheuristics*, LuLu.

Mani, A. & Patvardhan, C. (2009), A novel hybrid constraint handling technique for evolutionary optimization, *in* 'Proceedings of the Eleventh conference on Congress on Evolutionary Computation', CEC'09, IEEE Press, Piscataway, NJ, USA, pp. 2577–2583.

Matsumoto, M. & Nishimura, T. (1998), 'Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator', *ACM Transactions on Modeling and Computer Simulation* **8 No. 1**, 3–30.

Meinke, D. W., Cherry, J. M., Dean, C., Rounsley, S. D. & Koornneef, M. (1998), 'Arabidopsis thaliana: a model plant for genome analysis.', *Science* **282**(5389), 662, 679–682.

Mendel, G. (1865), 'Experiments in plant hybridization'.

Mercier, R., Pelletier, G., Jolivet, S., Drouaud, J., Durand, S., Vignard, J. & Nogu, F. (2008), 'Outcrossing as an explanation of the apparent unconventional genetic behavior of arabidopsis thaliana hth mutants', *Genetics* **180**(4), 2295–2297.

Michalewicz, Z. (1995), A survey of constraint handling techniques in evolutionary computation methods, *in* 'Proceedings of the 4th Annual Conference on Evolutionary Programming', MIT Press, pp. 135–155.

Michalewicz, Z. (1996), *Genetic algorithms + data structures = evolution programs (3rd ed.)*, Springer-Verlag, London, UK.

Michalewicz, Z. & Fogel, D. B. (2000), *How to Solve it: Modern Heuristics*, Springer Verlag, Berlin, Heidelberg, New York.

Michalewicz, Z. & Nazhiyath, G. (1995), Genocop iii: A co-evolutionary algorithm for numerical optimization problems with nonlinear constraints, *in* D. B. Fogel, ed., 'Proceedings of the Second IEEE International Conference on Evolutionary Computation', IEEE Press, Piscataway, New Jersey, pp. 647–651.

Mitchell, G. (2007), 'Evolutionary computation applied to combinatorial optimisation problems - phd thesis', *Dublin City University* .

Mitchell, G. G. (2005*a*), Quality-time tradeoff in a distributed parameter-less genetic algorithm, *in* 'Artificial Intelligence and Applications', pp. 738–742.

Mitchell, G. G. (2005*b*), Validity constraints and the tsp - generepair of genetic algorithms, *in* 'Artificial Intelligence and Applications', pp. 306–311.

Mitchell, G. G., O'Donoghue, D. & Trenaman, A. (2000), 'A new operator for efficient evolutionary solutions to the travelling salesman problem', ,Applied Informatics, Innsbruck, Austria, pp 771-774.

Mitchell, G., O'Donoghue, D., Barnes, D. & McCarville, M. (2003), 'Generepair-repair operator for genetic algorithms', *GECCO Conference Illinois* .

Murphy, G., Ryan, C. & Howard, D. (2007), [seeding methods for run transferable libraries] capturing domain relevant functionality through schematic manipulation for genetic programming, *in* 'Frontiers in the Convergence of Bioscience and Information Technologies, 2007. FBIT 2007', pp. 769 –772.

Nakano, R. & Yamada, T. (1991), 'Conventional genetic algorithm for job shop scheduling problems', *Proc. 4th International Conference on Genetic Algorithms* pp. 474–479.

Negnevitsky, M. (2005), *Artificial Intelligence A Guide to Intelligent Systems*, second edn, Addison-Wesley.

Neumann, F. & Witt, C. (2010), *Bioinspired Computation in Combinatorial Optimization Algorithms and Their Computational Complexity*, Springer.

Normal, B. A. & Bean, J. C. (1995), Random keys genetic algorithm for scheduling: Unabridged version, Technical Report 95-10, University of Michigan.

O'Donoghue, D. P. (2007), Statistical evaluation of process-centric computational creativity: Evaluating computer-generated analogies, *in* '4th International Joint Workshop on Computational Creativity (IJWCC)', Goldsmiths, University of London.

Orvosh, D. & Davis, L. (1993), Shall we repair? genetic algorithms, combinatorial optimization and feasibility constraints, *in* 'Proceedings of the 5th International Conference on Genetic Algorithms', Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 650.

Paredis, J. (1994), Co-evolutionary constraint satisfaction, *in* 'Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature: Parallel Problem Solving from Nature', PPSN III, Springer-Verlag, London, UK, pp. 46–55.

Parmee, I. C. & Purchase, G. (1994), 'The development of a di-rected genetic search technique for heavily constrained de-sign spaces'.

Peng, Chan, Shah & Jacobsen (2006), 'Increased outcrossing in hothead mutants', *Nature* **443**(E8), 28.

Poole, D., Mackworth, A. & Goebel, R. (1998), *Computational Intelligence A Logical Approach*, Oxford University Press.

Powell, D. & Skolnick, M. M. (1993), Using genetic algorithms in engineering design optimization with non-linear constraints, *in* 'Proceedings of the 5th International Conference on Genetic Algorithms', Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 424–431.

Ralphs, T. K., Kopman, L., Pulleyblank, W. R., Trotter, L. & Jr. (2001), 'On the capacitated vehicle routing problem', *Mathematical Programming Series* **94**, 343.

Ray, A. (2005), 'Plant genetics: Rna cache or genome trash?', *Nature* **437**(E1-E2).

Ray, T., Kang, T. & Chye, S. K. (2000), An evolutionary algorithm for constrained optimization, *in* 'Genetic and Evolutionary Computation Conference 2000', Morgan Kaufmann Publishers Inc., San Francisco California, pp. 771–777.

Rechenberg, I. (1971), Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Dissertation, *in* 'Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Dissertation'.

Reinalt, G. (1991), 'Tsplib - a travelling salesman problem library', *OSRA Journal of Computing* **3**, 376 – 384.

Reinelt, G. (1991), 'Tsplib - a traveling salesman problem library', *ORSA Journal on Computing* **3**(4), 376–384.

Riche, R. L., Knopf-lenoir, C. & Haftka, R. T. (1995), A segregated genetic algorithm for constrained structural optimization, *in* 'International Conference on Genetic Algorithms', pp. 558–565.

Ronald, E. M. A. (1995), When selection meets seduction, *in* 'International Conference on Genetic Algorithms', pp. 167–173.

Runarsson, T. P. & Yao, X. (2000), 'Stochastic ranking for constrained evolutionary optimization', *IEEE Transactions on Evolutionary Computation* **4**, 284–294.

Salcedo-Sanz, S. (2009), 'A survey of repair methods used as constraint handling techniques in evolutionary algorithms', *Computer Science Review* **3**(3), 175–192.

Schoenauer, M. & Xanthakis, S. (1993), Constrained ga optimization, *in* 'In Proc. of 5th Int'l Conf. on Genetic Algorithms', Morgan Kaufmann, pp. 573–580.

Smith, J. & Fogarty, T. (1996), Self adaptation of mutation rates in a steady state genetic algorithm, *in* 'Proceedings of IEEE International Conference on Evolutionary Computation', pp. 318 –323.

Surry, P. D. & Radcliffe, N. J. (1997), 'The comoga method: Constrained optimisation by multi-objective genetic algorithms'.

Tate, D. M. & Smith, A. E. (1993), Expected allele coverage and the role of mutation in genetic algorithms, *in* 'Proceedings of the 5th International

Conference on Genetic Algorithms', Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 31–37.

Terrall, M. (2002), *The Man Who Flattened the Earth: Maupertuis and the Sciences in the Enlightenment*, University Of Chicago Press.

Veale, T., O'Donoghue, D. & Keane, M. (1999), Cultural, psychological and typological issues in cognitive linguistics, *in* M. K. Hiraga, C. Sinha & S. Wilcox, eds, 'Cultural, Psychological and Typological Issues in Cognitive Linguistics', John Benjamins Publishing Company, pp. 129–154.

Walters, T. (1998), 'Repair and brood selection in the traveling salesman problem', **1498**, 813.

Weigel, D. & Jürgens, G. (2005), 'Hothead healer', *Nature* **434**, 443.

Weimer, W., Forrest, S., Le Goues, C. & Nguyen, T. (2010), 'Automatic program repair with evolutionary computation', *Commun. ACM* **53**, 109–116.

Yeniay, Ö. (2005), 'Penalty function methods for constrained optimization with genetic algorithms', *Mathematical and Computational Applications* **10**, 45–56.

Appendix A

Appendices

A.1 Implementation of GeneRepair

A.1.1 `EvolutionaryOptimisation` File

The main file of the GeneRepair package is the `EvolutionaryOptimisation` Class. The pseudo code for this class is shown in Algorithm A.1. This class executes the EO by following the steps of the basic Evolutionary Algorithm as described in Section 2.2.2. The population of individuals is a 2 dimensional array of tours or individuals where cities are represented by integers and each tour is a simple ordered sequence of cities - thus making as few assumptions about the underlying problem domain as possible. The last value in the tour stores the fitness of this tour. The sequence diagram (Figure A.2) illustrates how this class interacts during the running of the EO.

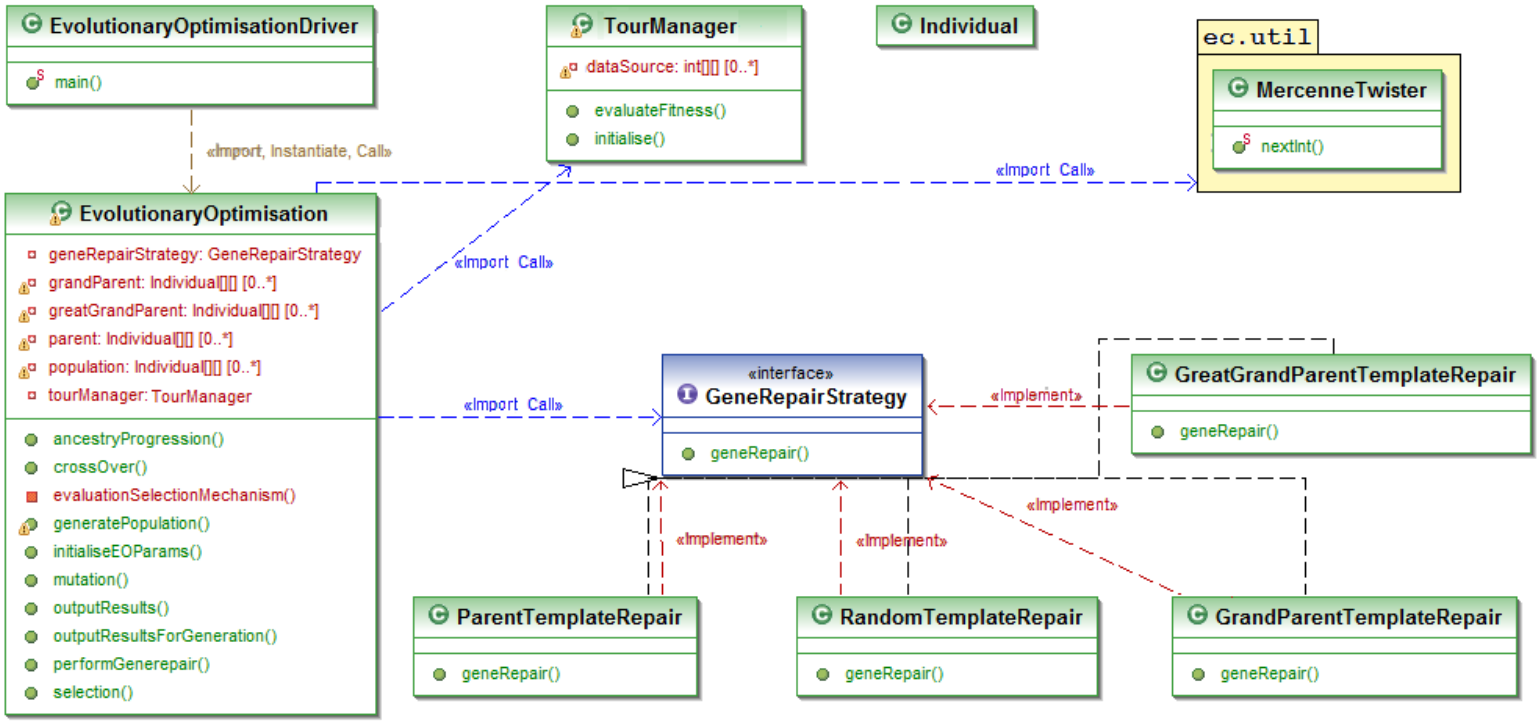

Figure A.1: GeneRepair Class Diagram

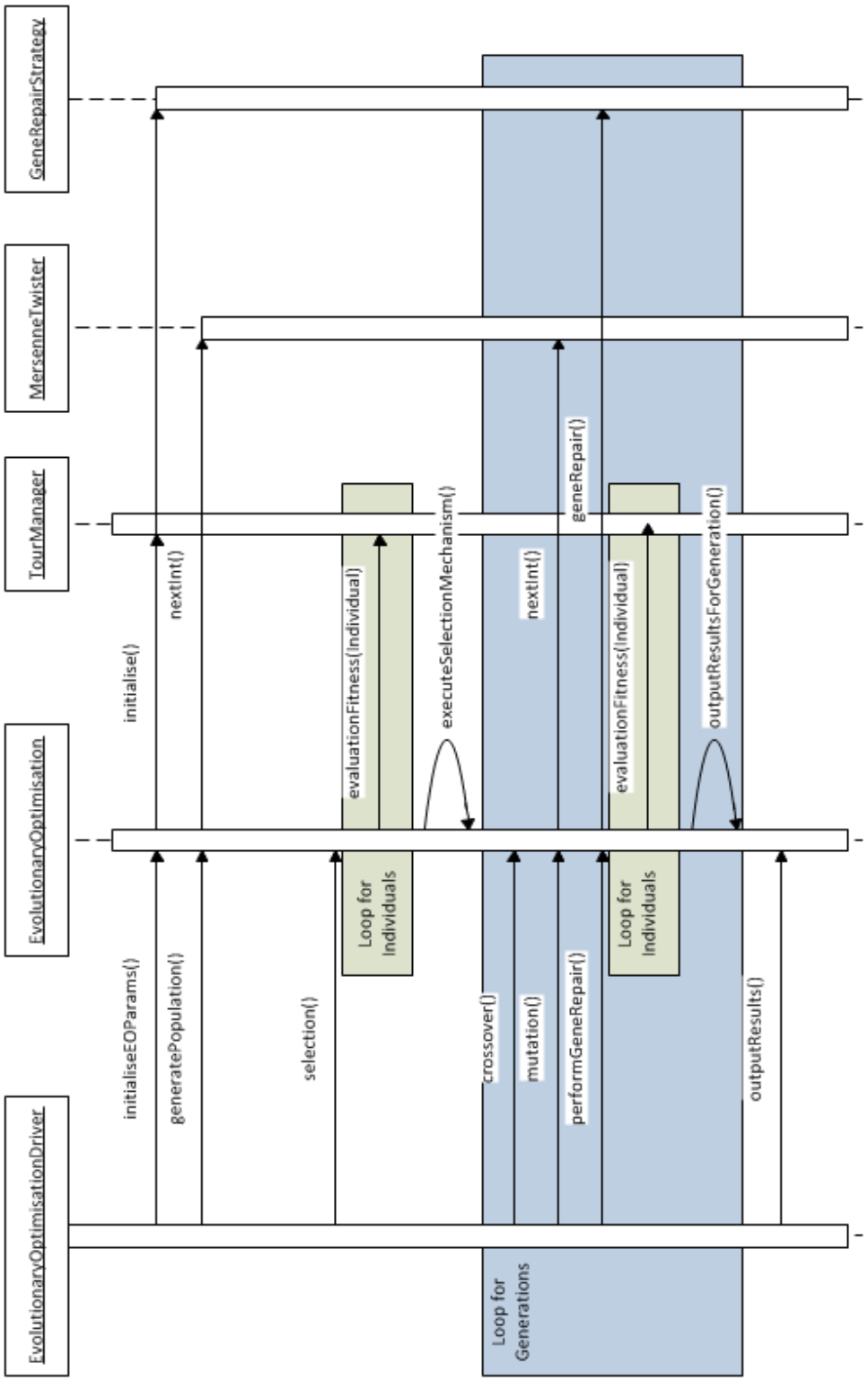

Figure A.2: GeneRepair Sequence Diagram

```
1 Generate an initial population reading information from the Map File
  for generation = 0; generation <numberOfGenerations; generation++
  do
2     Perform Chosen Selection Method on population;
3     Copy each ancestral repair template into its the subsequent
      template. (parent into grandparent etc);
4     Crossover parents using single point crossover;
5     Perform mutation according to rate provided by EODriver;
6     Call repair method in PTR, GPTR or GGPTR file according to
      parameter passed in by EO driver;
7 end
8 Print all findings to file and flush file;
```

Algorithm A.1: EO Pseudo Code

A.1.2 `EODriver` File

The EO Driver creates an instance of the EO passing in all of the necessary parameters. These parameters are:

- Population Size
- Number of Generations EO will Run
- Problem Data File Name
- Selection Method Identifier
- Mutation Rate
- GeneRepair Type Identifier (PTR, GPTR or GGPTR)
- GeneRepair Ancestor Fitness Identifier
- GeneRepair Direction

A.1.3 `Map` File

The Map file reads in the problem data and stores it in a two dimensional array. This reduces computation time as the information is read in once at the beginning of the EO and can be easily accessed from then on.

A.1.4 `TourManager` File

The Tour Manager file is used to access the data in the Map file and carry out calculations and manipulations using this data. This file is responsible for calculation of fitness.

A.1.5 Repair Files

The `PTR`, `GPTR`, `GGPTR` and `RTR` files can all be seen as repair files. The only difference between each of these files is the template they use to repair invaliditys in the individual. Each of these repair files uses the same algorithm which is illustrated in Algorithm A.2. The *error detection* phase is identical for all repair strategies - only the *error correction* differs between them. *Error correction* then uses different templates to support the repair operation. Erroneous alleles are replaced with corrective information that is selected from the relevant repair template.

A.1.6 Mersenne Twister Package

In order to ensure reliability when random numbers were used by the GeneRepair package a Mersenne Twister package was used. This `MersenneTwister` package version is Version 16 and is based on version MT199937(99/10/29) of the Mersenne Twister algorithm[1] and was written by Sean Luke in October 2004. This is a Java version of the C-program for MT19937: Integer version which was created by Makoto Matsumoto and Takuji Nishimura (Matsumoto & Nishimura 1998). This was used to avoid the pitfalls known to be associated with the java.util.Random method.

[1]This can be found at http://www.math.keio.ac.jp/matumoto/emt.html

```
1  for Every Individual in the Population do
2      Identify Repair template according to parameters;
3      Identify Direction of Repair;
4      Scan the individual and store a list of duplicates and a list of
       missing elements (Error Detection phase);
5      for i = first Missing City on List, i <total number of missing
       cities; i++ do
6          Scan invalid individual and identify first instance of erroneous
           information;
7          Scan repair template and identify first instance of repair
           information;
8          Replace identified extra city with missing city identified in
           repair template;
9      end
10 end
```

Algorithm A.2: GeneRepair Pseudo Code

A.1.7 Problem Data Files

The problem data files used for the experiments illustrated in Chapter 5 are the TSP and CVRP text files[2]. The files that were used to produce the results illustrated in this thesis are included in the Appendix. All datasets are benchmark problems from the TSPLIB.

[2]These text files are freely available to download at http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/