A ROUTE TO THE EVOLUTION OF COOPERATION: INVESTIGATIONS OF MULTILEVEL SELECTION IN EVOLUTIONARY COMPUTATION

XIAONAN WU

001311

CENTRE FOR NFLD. STUDIES
MEMORIAL UNIVERSITY
OF NEWFOUNDLAND

A Route to the Evolution of Cooperation: Investigations of Multilevel Selection in Evolutionary Computation

by

© Xiaonan Wu

A thesis submitted to the
School of Graduate Studies
in partial fulfilment of the
requirements for the degree of
Doctor of Philosophy

Department of Computer Science
Memorial University of Newfoundland

September 2011

St. John's Newfoundland

Abstract

To debunk the myth of how cooperation can emerge through the competition induced by Evolutionary Computation, this dissertation, inspired by nature, presents a new route to reach the evolution of cooperation in computational settings. The inspiration is drawn from multilevel selection theory in biology. This theory is an extension of the well-known group selection theory, which explains the evolution of cooperation by considering selection taking place both within and between groups. Although within-group selection encourages individuals to compete, between-group selection posits competition between groups, which leads to cooperation within groups. The concept of individuals and group are relative: groups can be regarded as individuals on a higher level; therefore, multilevel selection claims that selection should take place on every level of this hierarchical structure.

Indeed, our biological world is hierarchically organized. However, most multilevel selection models in the literature take this hierarchical structure as given. The biological hierarchy, however, has developed gradually: simpler, smaller components appear before more complex, composite systems. Therefore, the new computational multilevel selection model we propose defines a bottom-up process, where entities on new levels are created with the help of a cooperation operator under the guidance of

predefined reaction rules. Hence, new entities are able to possess different genotypic or phenotypic traits than their constituents. Evolution is performed on each level to optimize the traits of the entities on that level. Selection pressure from higher levels forces entities on lower levels to cooperate. Between-level selection determines which level to select and controls the growth of the hierarchy. As a result of these features, the model shows an emergent property: the appropriate structure required reaching a predefined cooperative goal, i.e., the number of individuals and the role each individual playing in the cooperation, are automatically developed during evolution.

After introducing the model, we first experimentally evaluate the feasibility of our proposed multilevel selection model in achieving the evolution of cooperation on the N-player Prisoner's Dilemma (NPD) game. We further explore the transition ability of our model by using division of labor as an example. Our findings reveal that cooperation emerges and persists more easily in this model than in other models from the literature. In fact, the between-group selection is strong enough to ensure groups with all required skills emerging from a population of independent individuals, no matter whether the skills are equally rewarded or not. Next, we validate the cooperation and problem decomposition capability of this model in solving decomposable problems. Two case studies are performed on string covering problems and multi-class classification problems, respectively. The experiment results show that our model evolves faster and finds more accurate solutions than other cooperative evolutionary algorithms. More importantly, problem decomposition emerges through evolution without human intervention.

The thesis concludes with a discussion of achievements and further work building on our results.

To Zhen with love,

for making me a better me!

To my parents with gratitude,

for supporting me in fulfilling my dreams!

Acknowledgments

The long journey towards pursuing the Ph.D degree finally comes to a successful end. The journey started with an unknown course: working on a brand new research area and living in a brand new culture. Fortunately enough, many people contributed to smooth the transition from unknown to "known" in innumerable ways, and I am grateful to all of them.

First and foremost, I feel very privileged to have worked with my supervisor, Dr. Wolfgang Banzhaf. His knowledge, experience and great insights into many scientific areas have been invaluable to me. Wolfgang, thanks for contributing ideas, time, and funding to make my Ph.D study productive and stimulating; thanks for allowing me to explore and develop my own ideas while staying closely to steer me back to the right track when I was confused or lost; thanks for encouraging me to continue my research when I was about to give up; thanks for sharing with me your contagious research passion and the secrets to successful research. I also would like to thank other members of my Ph.D committee, Dr. Jian Tang and Dr. Todd Wareham for their time and effort in evaluating my thesis and giving great feedback from their fresh perspectives!

Special thanks to Dr. Tom Lenaerts and Dr. Arne Traulsen for taking their time

to provide detailed comments on group selection models and critical suggestions on my experimental design and analysis, which started my interest in group selection theory. Many thanks go out to Dr. Malcolm Heywood for his experience and insight on the field of cooperative evolutionary algorithms, and his constructive criticism and excellent advice on my current and future research work.

I would like to acknowledge the academic, financial and technical support from the Department of Computer Science. In particular, thanks to Dr. Garnett Wilson, Dr. Simon Harding, Dr. Taras Kowaliw, Dr. Tina Yu, Dr. Manrique Mata-Montero, Dr. Krishnamurthy Vidyasankar, and Dr. Siwei Lu for their help and many inspiring discussions on my study and research throughout my Ph.D. program; thanks to Dr. Gong for reading and commenting on my thesis; thanks to Mr. Nolan White for providing constant technical support and inviting me to St John's Orienteering Club (that was a lot of fun.); and thanks to Ms. Elaine Boone, Ms. Darlene Oliver, and Ms. Sharon Deir for their daily assistance.

I have been so lucky to come across many great people over my years living in St John's. A heartfelt thank you to Ms. Eileen Kelly-Freake for encouraging and appreciating me. A sincere thank you to my gang and friends in St john's, because their companionship made daily life much more enjoyable and unforgettable. I particularly want to thank my dearest friend Jingjing Cai and Lei Wang for sharing laughs and tears with me, for taking care of me, and for simply listening to me.

Lastly, and certainly not least, I extend my inexpressible gratitude to my parents and parents in-law. Without their unwavering love and support, this thesis would not have been possible. Thanks for believing in me. Most of all, I offer my deepest thanks to my husband Zhen. Your love, tolerance, encouragement, and understanding turned

this long bumpy journey into a pleasure. Only you know how to make me happier and how to bring out the best in me. I am truly blessed to have spent the past 14 years with you and eagerly looking forward to starting the next chapter in our lives.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

This dissertation is about the evolution of cooperation. Within this context, cooperation means that individuals have to give up some of their survival or reproductive potential to help others. How could cooperation evolve, if the evolutionary principle of "survival of the fittest" seems to predispose individuals to be selfish? Apparently, nature manages the conflict very well, as we observe cooperation everywhere, in cells, in insects, in animals, in human beings, or even in our political and economic world. For example, animals might evolve to reduce their fertility to avoid over-exploiting their resources [113]; animals and insects defend their territory or community fiercely which often causes their own death; individual animals give alarm calls to protect their group at the risk of being the most obvious target of a predator. Cooperation exists because it confers evolutionary advantages; it can dramatically increase the survival rate of a group or a species—hence members inside, can accomplish complicated tasks which are not or nearly not possible to be achieved by individuals, and can help a group of individuals to function more efficiently and effectively.

Evolutionary Computation (EC) is a burgeoning research field of computational

intelligence. EC makes use of the Darwinian evolutionary principle, applying mechanisms of variation and selection to perform practical tasks in a variety of domains. Compared to other problem-solving strategies, EC has a number of advantages, such as efficiency, adaptivity and robustness in dynamically changing environments, less susceptibility to trapping in local optima, and less requirement of knowledge about the problem being solved. Because of its evolutionary origin, EC employs the "differential reproduction success" feature of natural evolution. Therefore, EC is normally regarded as a competitive optimization process. This implies that EC may fail to deal with situations where cooperation is required; for example, when solving problems which need a set of individuals jointly to perform a computational task, those individuals are highly dependent on one another. From this perspective, EC should conduct not merely a multimodal search; the interactions between coadapted individuals should be taken into account. To complicate matters further, individuals may function differently in cooperation, and hence might carry unequal fitness. Weak individuals, however, are more likely to be eliminated from the population, despite their possible unique contributions in a collaboration. In order to provide reasonable opportunities for cooperation to emerge through evolution, it is necessary to consider extensions to basic evolutionary computation models. However, designing a cooperative approach is very challenging. Many critical issues have to be addressed, such as problem decomposition, coadaptation between individuals, completeness of cooperation, conflict mediation between individuals and their collaboration interests.

1.1 Motivation

The existence of cooperation poses a perplexing problem for the theory of evolution. Individuals who behave cooperatively or altruistically put themselves at an evolutionary disadvantage, because reaching out to help others diminishes their own chance for survival. How, then, does cooperation emerge through competition? The answer, according to Darwin, is selection on group levels. As he wrote in 1871: "There can be no doubt that a tribe including many members who ... were always ready to give aid to each other and sacrifice themselves for the common good, would be victorious over most other tribes; and this would be natural selection" [15]. This perspective to explain the evolution of cooperation has gradually developed into group selection theory [5].

Group selection theory suggests that natural selection mechanisms should operate at two levels: within groups and between groups. Within-group selection works on individuals within the same group. It encourages individuals to compete against each other in pursuit of their own interests (i.e. it selects for high individual fitness). In this respect, it equals natural selection in the common sense. Between-group selection, in contrast, considers the total productivity of groups, and favors groups with good performance or groups whose members cooperate well. To better understand this concept, imagine two groups of meerkats digging in the sand when searching for food. In one group, meerkats take turns to guard the surrounding and give warning signals to group members at the first sign of an approaching danger, while in the other group, all meerkats are busy searching for food for themselves without watching out for others. Within-group selection in this example will prefer meerkats busily feeding themselves. Between-group selection, on the other hand, will prefer meerkats looking

out for others, because such a cooperative behavior benefits the whole group and increases the overall survival rate.

In short, the between-group selection pressure forces individuals to coadapt and cooperate so that a cohesive group can be formed. It also resolves and reduces conflicts within groups, because conflicts would reduce group performance. Those are exactly the lingering issues that the evolution of cooperation in computational settings must address.

1.2 Objectives

Just like group selection that successfully promotes cooperation in nature, the evolution of cooperation in computational settings should consider selection on different levels in order to encourage cooperation. Therefore, the primary goal of this dissertation is to extend the classic artificial evolutionary computation model to multiple levels, allowing selection and variation to work on each level, so that cooperation becomes an emergent property. This new multilevel selection model is useful in two respects:

- It can be easily mapped to a new evolutionary algorithm useful for computer scientists and engineers to solve complex problems whose solution is in the form of multiple coadapted subcomponents. We expect that this new algorithm will improve accuracy and efficiency over other available cooperative evolutionary algorithms in the literature.

- It provides a computational model useful for those researchers who are interested in computational aspects of biology, and hope to better understand the nature of

multilevel selection and study biological changes caused by multilevel selection, such as the evolution of cooperation, evolutionary transitions and other related issues.

The scope of this research project includes the following:

- At the abstract level, we will design a new computational multilevel selection model to achieve cooperation. In addition, a hierarchical evolutionary algorithm which implements this model is presented, with the purpose of enhancing the limitations of existing cooperative evolutionary algorithms.

- At the analytical level, we will verify the ability of this new model to achieve cooperation. Experiments will be designed to understand how cooperation can evolve and persist stably, and why the model behaves differently when compared to other well-known group selection models. At the same time, investigation of a hierarchical evolutionary algorithm should also be conducted, focusing on its ability to address issues, such as cooperation and problem decomposition.

- At the practical level, we will use the new model to study the main factors leading a group of individuals to a new type of individual at a higher level with different heritable traits; this represents an evolutionary transition as a direct consequence of the evolution of cooperation. We will also assess the applicability of the new algorithm to solving real-world problems. We will show how to customize the algorithm to fit particular application domains. The performance of the new algorithm will be evaluated and compared to other similar evolutionary algorithms.

1.3 Contribution

The primary contribution of this dissertation is to introduce ideas from group selection theory into artificial evolutionary computation models; as a result, traditional natural selection is extended to selection acting on multiple levels. Specifically, the contributions of this work can be summarized along the following two axes:

Main contributions to evolutionary computation

- Problem decomposition, evolution on higher levels (implying multilevel selection), and diversity preservation are identified as three essential factors for integrating cooperation in computational evolution. They are believed to bridge the discrepancy between current models of cooperative evolutionary algorithms and what could be inferred from the mechanisms of cooperation in nature.
- A new multilevel selection model is proposed which incorporated the three factors mentioned above. A hierarchical evolutionary algorithm implementing this new multilevel selection model is introduced. This is a general problem solving algorithm acting as a guideline for the practice of evolving cooperation in a bottom-up fashion. Therefore, it can be applied to a variety of domains and is not limited to any particular evolutionary algorithms. Experiments on two practical problems demonstrate it evolves solution faster and more accurate than other evolutionary algorithms that achieve the similar goal.
- Evolutionary pressure on multiple levels has been shown by experiments to be a powerful force in terms of i) modeling the coadaptation of and the

interaction between individuals; ii) developing different roles for individuals who participate in cooperation as an emergent property; iii) mediating the conflict of interest between individuals and their collective they are part of; iv) discovering an appropriate number of individuals in the cooperation without *a priori* information.

The above contributions have been published in the Proceedings of GECCO 2010 [117] and the Proceedings of GECCO 2011 [122].

Main contributions to artificial life

- An empirical comparison is conducted on two well-known group selection models that could be used to evolve cooperative systems, focusing mainly on their sensitivity to key parameter changes. To the best of our knowledge, no similar study has been conducted before. The findings can help researchers to understand how conditions or mechanisms produce differences among various group selection models. This contribution has been published in the Proceedings of the ECAL 2009 [120].
- The new multilevel selection model is customized as an alternative to explain the evolution of cooperation. Cooperation is very important in many different aspects, as it is a necessary step towards other biological changes. As confirmed by experiments, cooperation is easier to emerge from this new model than other well-known group selection models.
- The new model can be used to study evolutionary transitions by multilevel selection theory. It attempts to simulate nature's way of building the hierarchy of life more closely, in which evolutionary transitions are important

processes for new levels to came into being. We show, through carefully designed reactions rules in the cooperation operator and group fitness definition, that independent individuals can transition to groups with each member playing different roles. This contribution has been published in the Proceedings of the ECAL 2011 [119].

1.4 Dissertation Structure

The dissertation is organized as follows:

- Chapter 2 discusses the evolution of cooperation in nature. Group selection theory is highlighted as a potent explanation to resolve the contradictions between cooperation and evolution. Various group selection models are reviewed, among which two well-known models are empirically compared. The relationship between the evolution of cooperation and evolutionary transitions is outlined.

- Chapter 3 discusses the evolution of cooperation in computation. This chapter starts with a brief introduction of evolutionary computation, including its working mechanisms and framework. After discussing the limitations of EC, we point out three desired features that any cooperative evolutionary algorithm should be expected to possess. Using these features as a guideline, a comprehensive survey of existing cooperative evolutionary algorithms is conducted to unveil their strengths and limitations at promoting cooperation.

- Chapter 4 proposes the new computational multilevel selection model and hierarchical evolutionary algorithm (HEA), inspired by group selection theory. We

show how the model addresses the limitations of other cooperative evolutionary algorithms, and what potential problem domains the model can be applied to.

- Chapter 5 investigates the feasibility of our multilevel selection model in producing the evolution of cooperation. A sensitivity analysis and a performance comparison with other group selection models are performed. This chapter also explores how multilevel selection can be used to explain evolutionary transitions in evolution. The concept of division of labor, a group trait resulting from evolutionary transitions, is studied as an example, where low-level independent entities with specialized skills cooperate to increase the reproductive success of high level complexes.

- Chapter 6 studies the cooperation and problem decomposition property of the hierarchical evolutionary algorithm on simple string covering problems. In particular, we designed experiments to investigate if the algorithm is able to preserve and optimize coadapted subcomponents with unequal fitness in solutions to the targeted problems.

- Chapter 7 challenges the hierarchical evolutionary algorithm with real-world classification problems. Seven multi-class classification problems with different features, such as non-linearity, skewed data distribution and large feature space, are benchmarked. These benchmarks better showcase the ability of the algorithm to model the interaction between coadapted subcomponents and to decompose problems without human interference.

- Chapter 8 summarizes the main message of this dissertation, recapitulates its main contributions and limitations, and suggests some directions for future re-

search.

Chapter 2

Cooperation in Evolutionary Biology

In nature, the success of cooperation is witnessed at all levels of biological organization, ranging from genes and cells to multicellular organisms, social insects, and human society. This chapter, therefore, is devoted to discussing the evolution of cooperation in nature with the intention of inspiring the evolution of cooperation in computational models. In Sect. 2.1, we will present an overview of the evolution of cooperation. Particularly, we will discuss briefly four possible mechanisms to explain the evolution of cooperation in nature. Sect. 2.2 focuses mainly on one of the mechanisms: the group selection theory, which is the inspiration of this research work. Group selection has been unpopular in biology for most of the past 40 years, but has re-emerged in recent years as an important ingredient of thought in evolutionary biology [5]. It explains the evolution of cooperation by introducing selection between groups, not just between individuals. The competition between groups results in cooperation within groups. Sect. 2.3 empirically investigates two well-known group

selection models that represent research strands in group selection theory. The observations from these experiments reveal what aspects of design benefit group selection models, which in turn will provide us with insights of the do's and don'ts of an implementation that follows when new group selection models are to be proposed and developed.

2.1 The Evolution of Cooperation

In biology, **evolution** is the change in the inherited traits of a population of organisms through successive generations [30]. Darwin, in his principal works [14, 15], presented a wealth of evidence for evolution, and proposed natural selection as the driving force behind it. According to Darwin, individuals with traits that best adapted to their environment will survive and produce more offspring, thereby increasing the proportion of individuals with such traits in each successive generation.

To survive and reproduce, individuals need resources, such as energy, space, food, and appropriate environmental conditions. Resources, however, are normally limited. If more than one individual wants to use the same resource, there will be a situation of competition [41]. From this perspective, natural selection seems to predispose individuals to selfishness, i.e., evolution implies competition.

Nevertheless, we observe cooperative behavior everywhere, in cells, in insects, in animals, in human beings, or even in our political and economic world. For example, animals can evolve lower fertility to avoid over-exploiting resources [113]; insects such as bees risk death to defend their hives; birds give alarm calls to warn others of danger. On evolutionary grounds, such behavior does not seem to be successful,

because individuals with helping traits would be expected to go extinct through the process of natural selection; for instance, by giving warning calls, a bird delays its opportunity to flee to safety and attracts the attention of predators, thus increasing the odds of being killed by predators. The question of how natural selection could favor individuals that carry helping traits over those that carry selfish traits has fascinated evolutionary and behavioral biologists for several decades.

The Evolution of Cooperation [2] is the study attempting to address this question. Cooperation is a rather general term; it can describe behaviors which benefit both the actor (focal individual who performs a behavior) and the recipients (individuals who are affected by the behavior of the actor), and it can also describe behaviors which are beneficial to the recipients but costly to the actor [105]. The latter is usually called **altruism**, and is also the cooperation that "the evolution of cooperation" often refers to.

Cooperation in altruism is quite sensitive to circumstances and hence is unstable. Actors, also known as cooperators, are very vulnerable to being exploited by recipients who refuse to fulfill their role as actors themselves; such recipients are often called defectors, because they gain benefits without giving anything back. Let us consider the following example (shown in Fig. 2.1) where a population starts with all cooperators. Suppose that during evolution a defector shows up due to mutation or migration. Compared to cooperators, a defector benefits from the cooperative behavior of the cooperators, without paying any cost itself. If the costs and benefits are measured in terms of fitness, the defector will have relatively higher fitness than cooperators. Therefore, it has more chances to becoming selected for reproduction and will spread quickly. In the end, cooperators will vanish from the population [70]. Obviously,

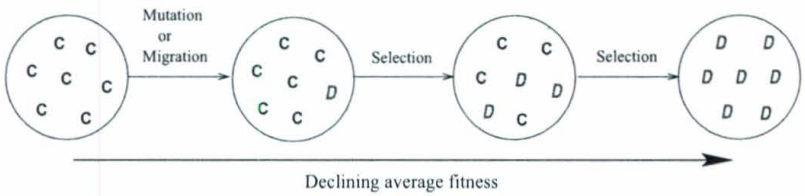


Figure 2.1: Without any special mechanism, the cooperation cannot be established during evolution. In a mixed population, cooperators (denoted as C) have relatively lower fitness than defectors (denoted as D), therefore natural selection continues to select against them until they are extinct. Adapted from "Five Rules for the Evolution of Cooperation" by Martin A. Nowak, 2006.

natural evolution needs additional concepts to allow the evolution of cooperation.

Mechanisms explaining under what conditions cooperation will emerge and persist during evolution include [68, 70]:

- **Kin selection**: Kin selection claims that natural selection favors cooperation when actors and recipients are genetically related. This theory expresses a gene-centered view of evolution [16, 40]: genes are the unit of evolution, while individuals are vehicles of selection. This differs from classic Darwinian theory where individuals are objects of evolution. Genes are "selfish" at promoting their own survival in order to spread in offspring. Cooperation indeed serves this need as kin share similar genes.

- **Reciprocation**: Kin selection sometimes fails to explain cooperation where relatedness is low or absent, for example the cooperation observed in symbiosis. Reciprocation has been proposed [55, 71, 99] to explain such cooperation in terms of deferring immediate personal gain toward potential benefits from

future mutual interactions. Mutual interactions can happen with repeatedly encountered individuals, or randomly encountered individuals, or individuals confined by a spatial structure. Evolutionary game theory is normally used to model and analyze reciprocation; that is to model the fitness consequences of social interactions between individuals [73].

- **Group selection**: Group selection is defined as the process of genetic change caused by the differential proliferation or extinction of groups of organisms [102]. Groups can be any unit of population structures, for example genes, cells, organisms, colonies, demes and possibly entire species. Selection conducted on groups would allow any traits that are costly to individuals but beneficial to groups, such as altruistic behaviors, to arise from evolution.

- **Social learning**: This mechanism refers to the preferential selection of the behaviors and skills individuals frequently encountered. In other words, individuals learn the most dominant behaviors and skills in their embedded social network. Simon [88] introduced the term "docile" to describe individuals who are adept to social learning, and who accept the instructions society provides to them. Cooperative individuals are docile, and accept the society's instruction to be altruistic as part of proper behavior. Therefore, they will gain extra benefits, despite the cost paid for being altruistic.

2.2 Group Selection and Multilevel Selection

Groups are common biological or social structures in nature. Colony, herd, pride, flock, and school all refer to groups of insects or animals. The cooperation within

a group, such as when hunting as a team or watching predators for others, offers its members a greater chance to survive severe competition. The emergence of co-operation, according to group selection theory, is due to the competition happening between groups. Although individual competition selects against altruistic behavior, group competition will favor altruistic behavior. This section first reviews the history of group selection theory and its relationship to other alternatives, like kin selection, the selfish gene theory and evolutionary game theory. Next, in order to show how the idea of group selection can be practically applied, three major group selection models based on biological observations are examined. Lately the discussion of group selection has been extended to a broader theme where selection can act simultaneously at multiple levels. This new perspective is called multilevel selection. The last part of this section, therefore, will explain multilevel selection and its implications for evolutionary transitions.

2.2.1 An Overview of Group Selection Theory

Group selection is a longstanding controversial area in the evolution of cooperation. The idea can be traced back to Charles Darwin already. In his book *The Descent of Man and Selection in Relation to Sex* [15], he observed that what was good for the group might not be good for the individual. The solution, according to him, is that groups containing mostly altruists have a decisive advantage over groups containing mostly selfish individuals, even if selfish individuals have the advantage over altruists within each group [113]. This statement accurately presents Darwin's position of considering selection acting at a level above individuals.

Darwin's idea was further developed by other evolutionists during the first half

of the $20th$ century. Well-known population genetics, Ronald Fisher, J. B. S. Haldane, and Sewall Wright, gave the idea a mathematical foundation [114]. However, it was Wynne-Edwards who in 1962 introduced the idea of group selection in *Animal Dispersion in Relation to Social Behavior* [123]. He defined group selection as a process in which an individual acted for the good of the group, regardless of whether it should be beneficial or detrimental to itself. Unfortunately, he invoked group selection to explain phenomena which usually have obvious explanations by individual, kin, or sexual selection. No wonder that his theory led to strong responses and criticism from, among others, George Williams, William Hamilton, John Maynard Smith, and Richard Dawkins; for example, Williams' book *Adaption and Natural Selection* strongly asserts that group-related adaptations do not exist, because group selection cannot overcome individual selection. As a result, the concept of group selection was rejected by many biologists, and a gene-centered point of view was embraced within evolutionary biology instead.

Despite the apparent retreat of group selection ideas during the following 20 years, some biologists, such as D. S. Wilson, E. O. Wilson, M. J. Wade, and E. Sober, continued to explore the possibility of group selection against vigorous criticism. They demonstrated the validity of group selection from a theoretical perspective. Meanwhile, numerous pieces of empirical evidence, such as the experiments conducted on hens [69], on beetles [12], on crops [38], and even on multi-species communities [37], suggested that group selection might prevail over individual selection under certain circumstances. In fact, in recent work of D. S. Wilson an E. O. Wilson [114], they concluded that sometimes between-group selection is a weak evolutionary force and sometimes it is very strong; the balance between within-group and between-group

selection should be evaluated on a case by case basis.

The rejection of group selection was caused partly by the neglect of early models for genetically based interactions among individuals [38]. Indeed, many of the strongest critics of group selection theory, such as Hamilton and Williams, have acknowledged the existence of group selection and its role in the evolutionary process [5, 113]. According to Wilson, group selection is not another alternative to explain the evolution of cooperation. Instead, it unifies two alternatives: kin selection and evolutionary game theory (i.e. reciprocation); the two alternatives are actually "versions of group selection theory, but presented in a formal framework which tends to obscure the face" [73, 112].

Selfish gene theory is "the final nail in the coffin that had been built for group selection" [5]. It stemmed from the work of Hamilton and Williams, but was popularized by Dawkins. Dawkins argued that individuals only exist temporarily during evolution; genes, on the other hand, are the true unit of selection: genes struggle perpetually to bequeath as many copies of themselves as possible to future generations. They "program" individuals to express phenotypic traits which increase the likelihood of individuals to survive and reproduce. Through individuals genes would be able to increase in numbers in subsequent generations. Therefore, individuals are merely the vehicles of selection, even though they interact directly with their environment and are direct targets of selection.

However, Dawkins failed to realize that he in fact invoked the idea of group selection to explain cooperation. According to him, independent genes are "ganging up together" [17] into chromosomes because they might have gained benefits: their biochemical effects might have complemented each other [77]. This is analogous to

individuals forming groups. Dawkins also failed to distinguish between the unit of selection and of inheritance [78]. Selection is about which variants survive best and reproduce most, while inheritance concerns the transmission of genotypic and phenotypic characters across generations. From this perspective, genes are the unit of inheritance, and individuals are the unit of selection. That is the reason why Hull [47] introduced the terminology *replicators* and *interactors* to refer to genes and individuals, respectively. From our perspective, group selection theory does not argue against the viewpoint that genes are replicators. It simply argues that both individuals and groups can be viewed as the vehicles of selection or interactors, because an individual or a group is a conglomerate phenotype that results from a complex set of interactions of genes and the environment surrounding them. Well designed vehicles should out-compete less well designed ones, and hence will pass on the genes that reside in them to the next generation. In other words, the adaptations observed either on the individual or group level will benefit the underlying genes that produced them.

2.2.2 Group Selection Models

The idea of group selection is straightforward: "selfishness beats altruism within groups. Altruistic groups beat selfish groups. Everything else is commentary" [113]. However, how to apply this idea to explain the evolution of cooperation is still hotly debated.

The earliest group selection model, see Fig. 2.2, was proposed by Wynne-Edwards [123]. The population in this model is divided into several reproductively isolated groups containing different amounts of cooperators and defectors. When a group goes extinct (for example, group 3 in Fig. 2.2), group selection will choose the group

with most cooperators, hence the highest average fitness (group 2 in Fig. 2.2), to

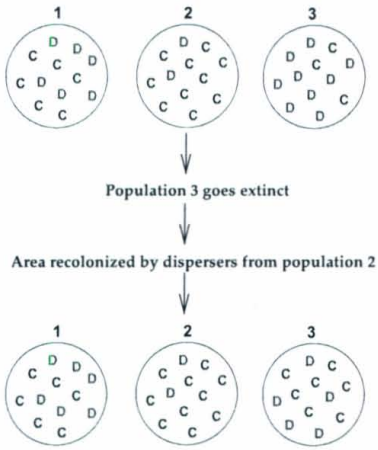


Figure 2.2: Naive group selection model. When a group goes extinct, group selection will select a group with many cooperators to recolonize this patch. The odds of a group going extinct is proportional to the frequency of its defectors. Adapted from Fig. 2.1 on page 20 in "Cooperation Among Animals" by L. A. Dugatkin, 1997.

recolonize this patch. The odds of a group going extinct are proportional to the frequency of defectors in the group. However, cooperators are unlikely to survive in this model [21, 111] because of two reasons. First, between-group selection depends on the extinction of other groups. Since the extinction of moderately sized groups is assumed to be rare, the possibility of propagating groups with many cooperators in a population is small [21]. Second, reproduction is restrained to the inside of a group; hence, within-group selection is against cooperators. As a result, a cooperative group will be quickly dominated by defectors before it gets the chance to propagate itself by recolonization. Although this model failed, it was the first attempt to understand

the evolution of cooperation by within- and between-group selection. This model is also known as "naive group selection".

Wilson and Sober [90, 111] developed a new trait group selection theory to replace the naive group selection model, see Fig. 2.3. This model is known as "modern group selection" theory, a generalized version of Maynard Smith's haystack model [59]. This model begins with a large global population of cooperators and defectors. When it comes to reproduction, they are randomly distributed into local groups. Natural selection first works on the group level. Groups with many cooperators will have the priority to be selected. Inside of a group, natural selection will select individuals with higher fitness, which are defectors, to produce offspring. However, the more cooperators there are in a group, the more chances exist to reproduce a cooperative offspring. After reproduction, groups dissolve, and individuals are mixed together, ready for another round of group formation and selection. Although within-group selection puts altruists in a disadvantaged position, cooperative groups will contribute more cooperators to the next generation. Furthermore, the mixing phase provides opportunities for cooperators to spread in the population, whereby the average fitness of the population is increased.

Recently Traulsen and Nowak proposed a minimalist stochastic group selection model [97, 98], see Fig. 2.4. The population in this model contains up to n groups, and there are no more than m individuals in a group. At each time step, an individual from the entire population is selected proportional to fitness for reproduction. An offspring is added to the same group. If the size of this group exceeds m at some point, the group has to split into two with a probability q. The individuals of the original group are randomly assigned to either of the two new groups. In order to

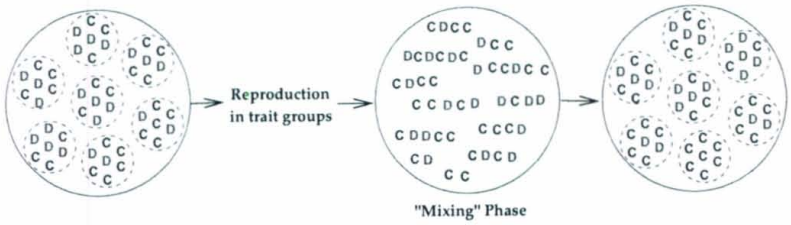


Figure 2.3: Trait group selection model. This model describes a group level process that is strikingly analogous to individual selection: Firstly, groups vary in genetic composition, productivity and/or persistence; secondly, the selection frequency is proportional to genetic variations among groups; finally, the selection on groups increases productivity and persistence of groups. Adapted from Fig. 2.1 on page 20 in "Cooperation Among Animals" by L. A. Dugatkin, 1997.

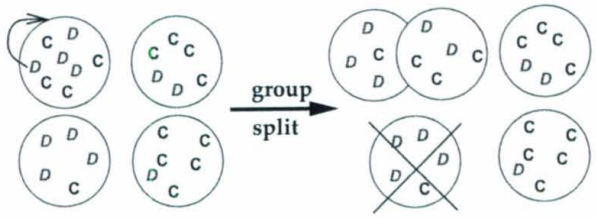


Figure 2.4: Traulsen's group selection model. The entire evolutionary dynamics are driven by individual reproduction. The evolution of individuals changes group size. When the groups reach a certain size, they will stay together or split. Groups with more cooperators reach the critical size faster and, therefore, split more often. Adapted from "Evolution of cooperation by multilevel selection" by A. Traulsen and M. A. Nowak, 2006.

maintain no more than n groups in the population, another randomly selected group has to be eliminated from the population. However, with probability $1-q$, the group does not divide. In such a case, an individual from this group is randomly selected and deleted to keep group size at m. The specialty of this model lies in the reproduction on the individual level, which triggers the splitting of a group, and further leads to selection on the group level. Cooperative groups reach critical size faster and split, therefore, more often. In a sense, the evolutionary dynamics is entirely driven by individual evolution, and group selection emerges from individual selection.

In summary, the three models demonstrated here have their own perspectives on how to apply within- and between-group selection to encourage cooperation. One key message obtained from these three group selection models is that it is possible to create various assortments of cooperators and defectors in groups, which directly will result different group fitness. Only when sufficient fitness variations between groups are maintained will the between-group selection be able to gain force. Wilson's and Traulsen's models regularly change the genetic composition of groups, i.e. the proportions of defectors and cooperation in groups, while the naive group selection model evens out the variations between groups by recolonization, which is another way to explain the failure of that model. The other key message is to distinguish group selection from between-group selection. Between-group selection simply applies selection pressure between groups, whereas group selection employs between-group selection at various stages, such as individual reproduction, group replacement, or individual replacement. In addition, it has to associate the reproduction probability of individuals to their group's through between-group selection. Therefore, individuals with cooperative traits but with low fitness values are able to survive the selection.

That is to say, a model which only adapts between-group selection cannot be called a group selection model.

2.2.3 Multilevel Selection and Transition

In group selection models, individuals and groups are relative: an entity can be regarded as a group for entities in the level below it and as an individual of a entity in the level above it. In fact, the biological world is hierarchically organized. The hierarchy of life, starting from the bottom level to the top level, includes atoms, molecules, organelles, cells, tissues, organs, organ systems, organisms, populations, communities, ecosystems and biospheres [107]. Entities at many levels of the biological hierarchy undergo reproduction or multiplication. Therefore, they exhibit "heritable variation in fitness". According to Lewontin [52], natural selection should operate on those entities, i.e., on different levels. This new perspective is now called multilevel selection (MLS) theory, an extension of group selection. D. S. Wilson and E. O. Wilson [114] interestingly suggested to apply the "Russian matryoshka dolls" metaphor to MLS theory: Levels are nested one within another. At each level evolution favors a specific set of traits to increase the relative fitness of entities on that level. However, the selection on two adjacent levels does not necessarily need to act in the same direction; a trait, such as selfishness, which is selectively advantageous at a level can be disadvantageous at a level above. The adaptation at the higher level determines whether or not such traits should be suppressed.

With respect to the hierarchy in MLS, where a number of individual entities are nested within each group entity, we need to clarify which entities should become the objects of evolution [76]. If we are interested in the changing frequencies of

different traits of individual entities, the individual entities will be the objects of evolution. Group entities are treated as a structure or an environment where the fitness-affecting interactions take place. Take Wilson's model as an example. This model concerns how to spread the altruistic trait among individuals in a population. To this end, groups are regularly formed. Groups with more altruists will have a higher fitness; hence cooperative individuals will have higher probabilities to produce offspring. Obviously, groups are temporary fitness-bearing entities. Even though they are selected, it is not them, but individuals that are reproduced, and also it is the frequency of individual traits that is changed. This type of MLS is called MLS type 1 (MLS1) [13, 76]. Alternatively, if we are interested in the changing frequencies of different group entities, group entities will be the objects of evolution. That is to say, group entities are not only an environment to individual entities or an object of selection; they actually have their own heritable traits. Group entities with higher fitness will reproduce more offspring group entities with similar traits. Individual entities may still undergo selection within each group entity, which leads to changes in the distribution of individual traits and potentially affects group entity traits. This type of MLS is called MLS type 2 (MLS2) [13, 76].

The key difference between MLS1 and MLS2 is the focal level [78], the level that goes through evolution. Because of the different focal units, evolution on each level is different and thus causes different evolutionary changes. The best group entities in MLS1 will also be the best ones in MLS2; however, the ones in MLS1 will contribute the most individual entities to the next generation, while the ones in MLS2 will contribute the most groups. Both MLS1 and MLS2 are distinct processes that can occur in nature [76]. A failure to distinguish clearly between MLS1 and MLS2 plagued

many traditional discussions of the levels of selection [13, 73]

Accepting multilevel selection, according to Wilson [113], has profound implications. It plays a very important role when thinking about the "major transitions" in evolution. The major evolutionary transitions [9, 63, 60] refer to the creation of new, higher-level complexes of simpler entities. Summarized by Michod [63] for example, they include the transitions "from individual genes to networks of genes, from gene networks to bacteria-like cells, from bacteria-like cells to eukaryotic cells with organelles, from cells to multicellular organisms, and from solitary organisms to societies". Of course, the existence of a biological hierarchy should not be taken for granted; multicellular organisms do not exist at the beginning of life. According to MLS theory, a major evolutionary transition occurs when higher level selection (i.e. between group selection) dominates lower level selection (i.e. within-group selection) [113].

Okasha [75] claims both MLS1 and MLS2 may be relevant to evolutionary transitions. An evolutionary transition is more complicated than the evolution of cooperation. However, before transitions take place and complexes emerge, simpler entities which constitute the complexes have to be able to work together. They need to sacrifice their individuality and exhibit cooperative traits. Therefore, in the early stage of evolutionary transitions, the evolution of cooperation has to emerge, so that cooperative traits can spread among simpler entities in the population. That is exactly what MLS1 does: using groups as an environment to help individual traits to propagate. Once individuals are willing to form cohesive complexes, evolution should work on complexes to gradually develop their own traits. In other words, complexes become the objects of evolution. Through selection and reproduction, complexes are better

adapted to their environment and eventually become discrete units, normally with traits different from their constituents' traits. It follows that MLS2 is applied at a later stage of evolutionary transitions.

The shift from MLS1 to MLS2 also indicates a change in group fitness definitions. In MLS1, group fitness is defined as the average fitness of the individuals within a group, while in MLS2, group fitness is defined independent of the average individual fitness. As the transition proceeds, group fitness gradually becomes "decoupled" from individual fitness [65], until it is no longer closely related to the average individual fitness. Once group fitness is decoupled, the transition has been achieved, and new complexes have been created that assume an existence of their own.

2.3 An Empirical Study of Wilson's and Traulsen's Group Selection Models

Wilson's [90, 111] and Traulsen's [97, 98] group selection models represent two research strands in organizing group structures: mixing/dispersing groups (i.e. Wilson's model) or not (i.e. Traulsen's model) during evolution. The within- and between-group selection, correspondingly, will work differently. Wilson's model has been well studied [48, 50, 82, 83] on the conditions that allow group selection to be effective, but not Trauslen's, as it is relatively new. Therefore, the purpose of this section is twofold; first, we investigate the two models for cooperation in the context of the *n-player* prisoner's dilemma game, in order to derive their differences in performance; second, we provide with this dissertation certain preparations for the application of group selection models in evolutionary computation. The empirical study conducted

here will reveal what aspects of design benefit group selection models most, which help us gain valuable insights into how encourage cooperation in evolutionary computation using the idea of group selection.

2.3.1 The N-player Prisoner's Dilemma Game

The $n-player$ prisoner's dilemma (NPD) game [2], an extension of the classic prisoner's dilemma game but involving any numbers of players, has been used extensively to study the evolution of cooperation. Each player or individual in this game faces two possible strategies, cooperate or defect, where the payoff to each player depends on his/her own strategy and the number of other players who play the cooperate strategy. Individuals get a higher payoff from playing defect than from playing cooperate. However, all n players are better off if all play cooperate than if all play defect. This game offers a straightforward way of thinking about the tension between the individual and group level selection [24], because the two selection forces produce starkly different outcomes. If individuals are selected to act in their own interest, all will defect. If they are selected to act in the group interest, all will cooperate.

In this study, we also use the NPD game as a research vehicle to explore the dynamics between the individual and group selection in Wilson's and Traulsen's models. In both models, N individuals are randomly divided into m groups. Individuals in a group independently choose to be a cooperator or a defector without knowing the choice of others. The fitness function of cooperators ($f_{C_i}(x)$) and defectors ($f_{D_i}(x)$)

in group i are specified by the following equations [90]:

$$f_{C_i}(x) = base + w(\frac{b(n_i q_i - 1)}{n_i - 1} - c), \quad (0 \leq i < m) \tag{2.3.1a}$$

$$f_{D_i}(x) = base + w\frac{bn_i q_i}{n_i - 1}, \quad (0 \leq i < m) \tag{2.3.1b}$$

where *base* is the base fitness of cooperators and defectors, b and c are the benefit and cost caused by the altruistic act, respectively, q_i the fraction of cooperators in group i, n_i the size of group i, w a coefficient. Evidently, cooperators have a lower fitness than defectors, as they not only pay a direct cost, but also receive benefits from fewer cooperators than defectors do. The fitness of group i is then defined as the average fitness of its individuals, shown in Eq. 2.3.2.

$$\begin{aligned} g_j(x) &= \frac{nq_i f_{C_i(x)} + n(1 - q_i) f_{D_i}(x)}{n} \\ &= base + wq_i(b - c) \end{aligned} \tag{2.3.2}$$

From Eq. 2.3.1 we can see that the relative fitness value of cooperators and defectors are directly affected by three parameters: q_i, n_i, and w. To understand the effects of those parameters, we plot, as an example, in Fig. 2.5 the fitness values of groups at size 5 and 10 with different percentages of cooperators (i.e., the changes of q_i), and, accordingly, the fitness values of cooperators and defectors. Parameters *base*, b, c and w are set to 10, 5, 1, and 1, respectively. The increase of q_i improves group fitness, but at a slower pace than the increase of individual fitness, especially when n_i is smaller. The increase of n_i has no effects on group fitness, but makes the relative fitness between cooperators and defectors more distinct. Obviously, the changes of different parameters affect individual and group fitness in various ways. It is not straightforward to conclude which conditions (i.e., the settings of parameters) allow between-group selection to dominate within-group selection. Therefore, experiments

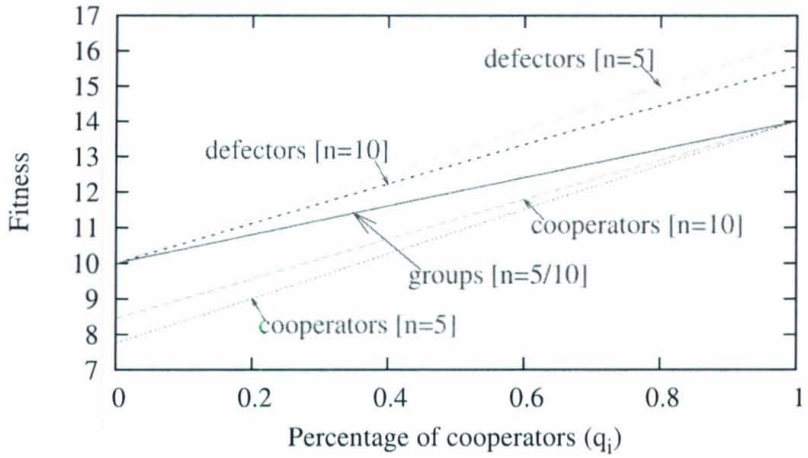


Figure 2.5: The fitness values of cooperators, defectors, and their groups with respect to the change of q_i when group size n are set to 5 and 10, respectively. *base* is set to 10, b to 5, c to 1 and w to 1.

have to be designed to serve this purpose.

2.3.2 Algorithm Design

A simple algorithm implementing Wilson's model (denoted as W) is described in Algorithm 1. This algorithm starts with a randomly initialized population P containing N individuals, r percent of which are cooperators. P is then divided into m groups. The individual and group fitness of the dispersed population P' are evaluated. Afterwards, reproduction begins; group gn is first selected, from which an individual idv is selected to produce offspring idv'. idv' is put back into the same group as its parent. Because the selection of groups is proportional to fitness, cooperative groups will contribute more offspring, thus resulting in various group sizes. In total, N' offspring

will be produced, which is calculated by Eq. 2.3.3 [90].

$$N' = \sum_{i=1}^{m} n_i \times (q_i \times f_{C_i}(x) + (1 - q_i) \times f_{D_i}(x)) \tag{2.3.3}$$

Normally N' is larger than N. This gives cooperators an opportunity to increase their frequency in the next generation. To maintain the original population size N, groups in P' are mixed and each contributes individuals proportional to its size to the new population P. The algorithm will repeat the above steps until the population converges or the maximum number of generations is reached.

Algorithm 1: An algorithm implementation of Wilson's model

```
1  P ← Initialize_Population(N, r);
2  while population does not converge or max generation is not reached do
3      P' ← Disperse_Population(P, m);
4      Evaluate_Fitness(P');
5      for i ← 0 to N' do
6          gn ← Select_Group(P');
7          idv ← Select_Individual(P', gn);
8          idv' ← Reproduce_Offspring(idv);
9          Add_Individual(idv', gn)
10     end
11     P ← Mixing_Proportionally(P');
12 end
```

Similarly, the algorithm implementing Traulsen's model is shown in Algorithm 2. This algorithm initializes, disperses and evaluates the population the same way algo-

Algorithm 2: An algorithm implementation of Traulsen's model

```
1  P ← Initialize_Population(N, r);
2  P' ← Disperse_Population(P, m);
3  while population does not converge or max generation is not reached do
4      Evaluate_Fitness(P');
5      for i ← 0 to N'' do
6          idv ← Select_Individual_from_Population(P');
7          idv' ← Reproduce_Offspring(idv);
8          Put_Back_to_Group(idv', gn);
9          if Group_Size(gn) > n then
10             rnum ← Generate_Random_Number(0, 1);
11             if rnum < q then
12                 Split_Group(gn);
13                 Remove_a_Group();
14             else
15                 Remove_an_Individual_in_Group(gn);
16             end
17         end
18     end
19 end
```

rithm W does. However, there are two major differences. First, the population only divides once at the beginning of the process; the groups are kept isolated afterwards. Second, the reproduction step is different. An individual idv is selected from the entire population for reproduction, rather than from a particular group. Offspring idv' is put back into its parent's group, group gn. If the size of group gn exceeds the pre-defined group size n, a random number $rnum$ is generated. If $rnum$ is less than a group splitting probability q, group gn splits and its individuals are randomly distributed into offspring groups. An existing group has to be removed to maintain a constant number of groups; otherwise, an individual from group gn is removed. In Traulsen's model, a group or an individual to be eliminated is randomly selected. As an extension, we also investigate selecting such a group or individual inversely proportional to its fitness. Therefore, two variations of Algorithm 2 are implemented and we refer to the former as $T1$ and to the latter as $T2$.

2.3.3 Experimental Setup

The investigations focus on the effects caused by different group size n, initial fraction of cooperators r, and coefficient w. Parameters n and r affect the assortment between cooperators and defectors in groups, and coefficient w affects the individual and group fitness; both cause changes in selection dynamics.

To focus on the selection dynamics, we assume asexual reproduction without the interference of mutation. A roulette wheel selection is adopted in the reproduction step for all 3 algorithms. Parameters that are common to all experiments are set as follows: runs $R = 20$, generation $gen = 5,000$, population size $N = 200$, base fitness $base = 10$, benefit $b = 5$, cost $c = 1$, group splitting probability $q = 0.05$, N''=10, and

N' is decided by Eq. 2.3.3 [90].

For each algorithm, we measure the success ratio by the number of runs whose population converges to cooperators to the total number of runs 20. The larger the ratio, the more likely an algorithm favors cooperation. We also collect the average variance ratio [25], as defined in Eq. 2.3.4.

$$v = \frac{var_B(q_i)}{var_T(Q)} = \frac{\frac{\sum n_i(q_i-Q)^2}{N}}{\frac{A(1-Q)^2+S(0-Q)^2}{N}} \tag{2.3.4}$$

where N is the population size, A the total number of cooperators in the population, S the total number of defectors in the population, $Q = A/N$, q_i the fraction of cooperators in group i, and n_i the size of group i. v indicates composition difference between groups. The higher this ratio, the more prominent the effect of group selection.

2.3.4 Parameter Sensitivity Analysis

Sensitivity to group size and initial fraction of cooperators

First we investigate how the three algorithms behave under different group sizes. We set $r = 0.5$ and $w = 1$. Group size n is varied in steps from $\{5, 10, 20, 50, 100\}$. The success ratio and average variance ratio (in brackets) for each setting are listed in Table 2.1. The average variance ratio is not shown for $T1$, because $T1$ is used as a reference of $T2$.

As can be seen, the performance of $T1$ degrades as n grows. The population in W converges to cooperators when small groups are employed ($n = 5$ or 10). As n increases, evolving cooperation becomes difficult. In contrast, $T2$ converges to cooperators except for $n = 100$.

These observations can be explained by the following figures collected from a par-

Table 2.1: The effects of group size n on W, $T1$ and $T2$ when r=0.5

n	W	T2	T1
5	1(0.196)	1(0.820)	1
10	1(0.092)	1(0.655)	0.85
20	0.8(0.045)	1(0.291)	0.65
50	0(0.015)	1(0.112)	0.15
100	0(0.004)	0(0.011)	0

ticular run. Figure 2.6 shows that the variance ratio v of W decreases as n increases, which reduces the effect of group selection. As a result, selection on the individual level is becoming the dominant force, so the population converges quicker to defectors, see Fig. 2.7. The same trend between v and n is also observed in $T2$. However, given that n ranges from 5 to 100, its v value is much higher than or at least equal to the highest v value of W (see Fig. 2.8). This implies that $T2$ preserves variance between groups better than W, and explains why $T2$ is more effective than W in evolving cooperation. Unlike W, the speed of $T2$ converging to cooperators does not accelerate as n gets smaller; for example, runs with $n = 10$ and 20 converge faster than runs with $n = 5$ (see Fig. 2.9). When groups are too small or too large, much averaging (i.e., repeated group splitting and replacing) is required to remove defectors from the population.

We further adjusted the value of r from 0.5 to 0.3 and 0.1. We were curious about the response of the three algorithms to this change, because when r drops, the number of cooperators assigned to groups is smaller, which increases the influence

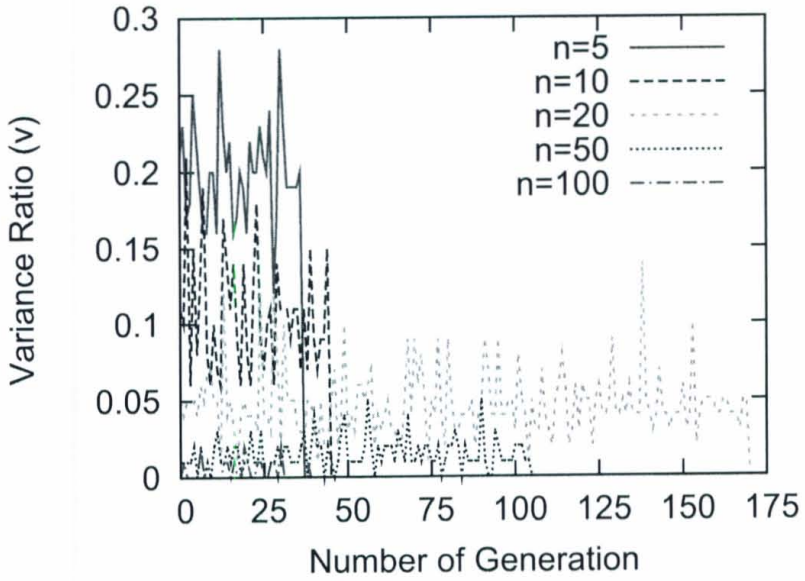


Figure 2.6: Variance ratio v of W as n increases.

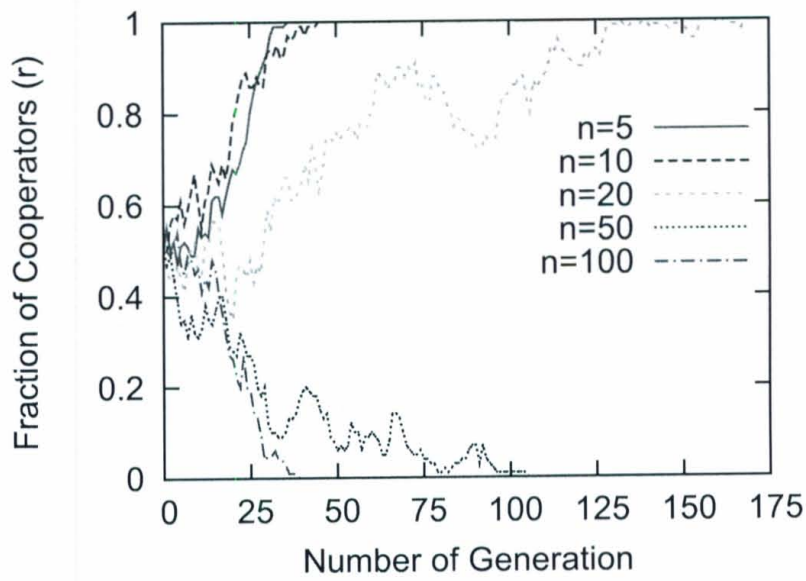


Figure 2.7: Fraction of cooperators in W as n increases.

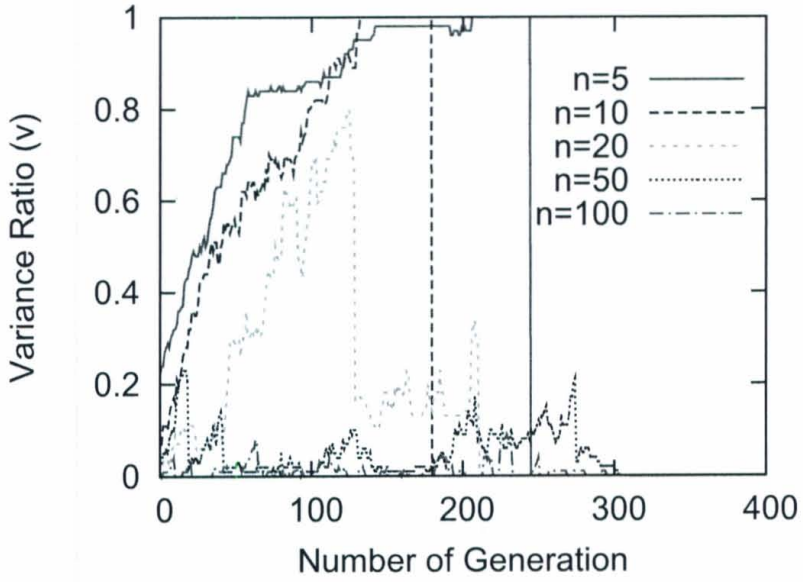


Figure 2.8: Variance ratio v of $T2$ as n increases.

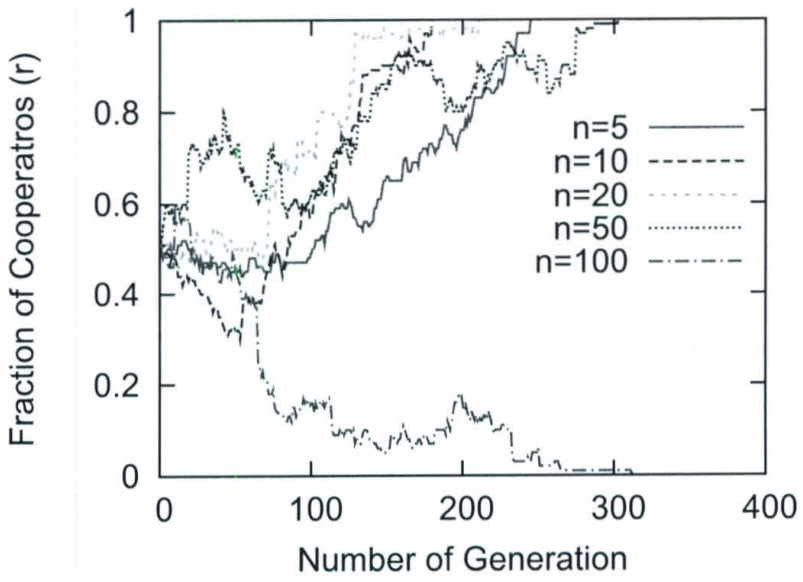


Figure 2.9: Fraction of cooperators in $T2$ as n increases.

of individual selection in a group. As shown in Table 2.2, the performance of $T1$

Table 2.2: The effects of group size n when r=0.3 and r=0.1

n	r=0.3			r=0.1		
	W	T2	T1	W	T2	T1
5	1(0.201)	1(0.853)	0.95	1(0.196)	1(0.893)	0.7
10	1(0.098)	1(0.665)	0.55	1(0.095)	1(0.767)	0.2
20	0.55(0.045)	1(0.398)	0.25	0.25(0.042)	0.65(0.465)	0.1
50	0(0.016)	0.8(0.105)	0.1	0(0.015)	0.55(0.049)	0.05
100	0(0.005)	0(0.014)	0	0(0.005)	0(0.015)	0

decreases as r drops. For W and $T2$, when n is small (5 or 10), due to the strong group selection effects, the decrease of r does not affect the success ratio, but only slows convergence speed towards cooperation; for larger groups, as n increases (group selection is weaker) and r decreases (individual selection is stronger), group selection can hardly dominate individual selection, so it becomes difficult for both algorithms to preserve cooperation. However, $T2$ is less affected, because for a given group size, similar v values in W are observed despite the changes of r, while relatively high v values are produced by $T2$ even if r drops.

Sensitivity to selection pressure

The composition of groups is not the only factor that drives selection dynamics; a difference in fitness values of cooperators and defectors is another one, as it affects the pressure put on groups and individuals. We change coefficient w to adjust the

selection pressure, namely to weak and strong selection. Strong selection means that the payoff is large compared to baseline fitness, i.e. w is large; weak selection means the payoff is small compared to baseline fitness, i.e. w is small [72].

We tested the three algorithms with r=0.5 and set w to {0.1, 0.5, 1, 2, 5, 10}, respectively, on all group sizes. Results are shown in Table 2.3. One first notices that the performance of the three algorithms increases and then decreases as selection pressure goes from weak to strong. If selection is too weak, the fitnesses between the two roles and between groups are very close. Hence, group and individual selection become neutral, especially if large groups are adopted, so defectors can more easily take over the population. If the selection is too strong, cooperators are more difficult to be selected because of the larger relative fitness between the two roles, even though group selection still favors cooperative groups. To be more specific, for small groups (n = 5 or 10) only $T2$ can successfully preserve cooperation under both weak and strong selection. The increase of selection pressure raises the influence of individual selection. In response to this increase, the variance ratio in W for a given group size does not change at all, while $T2$ still keeps noticeably high variance ratios. This also explains why $T2$ outperforms W with larger groups.

2.3.5 Discussion

The above experiments demonstrate that maintaining variance between groups has great impact on group selection models. For W, if groups are randomly formed, small group sizes are desired because small groups increase group variance. This confirms previous investigations (see [25, 50, 102] for examples). We further show that such a requirement only works if selection is weak. $T2$, because it is able to introduce high

Table 2.3: The performance of W, $T1$ and $T2$ under weak and strong selection.

	w=0.1			w=0.5			w=1		
n	W	T2	T1	W	T2	T1	W	T2	T1
5	1(0.197)	1(0.949)	0.6	1(0.201)	1(0.884)	0.9	1(0.196)	1(0.820)	1
10	1(0.095)	1(0.766)	0.6	1(0.096)	1(0.515)	0.8	1(0.092)	1(0.655)	0.85
20	0.85(0.044)	0.95(0.601)	0.55	1(0.046)	1(0.370)	0.65	0.8(0.045)	1(0.291)	0.65
50	0.4(0.015)	0.45(0.174)	0	0(0.015)	1(0.115)	0.3	0(0.015)	1(0.112)	0.15
	w=2			w=5			w=10		
5	1(0.196)	1(0.806)	1	0.9(0.196)	1(0.820)	1	0(0.190)	1(0.875)	1
10	1(0.096)	1(0.543)	0.9	0.1(0.096)	1(0.596)	0.8	0(0.096)	1(0.638)	0.85
20	0.1(0.042)	1(0.309)	0.8	0(0.050)	0.8(0.334)	0.5	0(0.046)	0.8(0.347)	0.15
50	0(0.014)	0.8(0.079)	0.15	0(0.014)	0.45(0.021)	0	0(0.016)	0.1(0.053)	0

group variance, is more robust towards parameter changes. One reason lies in how the two models manage groups. Mixing and re-forming groups like in Wilson's model constantly averages the variance between groups, so in Fig. 2.6 we observe the variance between groups fluctuating. In contrast, because groups in Traulsen's model are kept isolated, and the selection step in reproduction is proportional to individual fitness, the fraction of cooperators in a cooperative group grows faster than in a less cooperative group, hence gradually increasing the variance between groups. The other reason is because of group splitting. Group splitting changes group size and group composition, which in a way increases the dynamic between groups. In contrast, W which always maintains fixed group sizes needs external help to increase between-group variance, such as migration [48, 82, 83] or special group structures [82]. $T2$ performs better than $T1$ under all settings, because removing an individual or a group according to the value of its fitness inverse at survival selection is very likely wiping out defectors, thus it certainly helps cooperators.

2.4 Chapter Summary

The evolution of cooperation is a fundamental problem in evolutionary biology. The mechanisms by which cooperative or even altruistic behaviors could evolve have been vigorously debated over the last several decades. The most prominent theories include kin selection, reciprocation, group selection and social learning. Group selection theory, which used to be considered as a typical example of flawed evolutionary thinking, has now re-emerged as an important component of evolutionary biology [5]. It explains the emergence of cooperation by selection acted on group levels: between-group

selection favors traits that are detrimental to individuals but are beneficial to groups, such as cooperation. The concept of individuals and groups are relative; a group can be regarded as an individual of entities at one level higher. That is to say, levels are nested within each other, and natural selection may operate simultaneously at more than one level. This new perspective is known as multilevel selection theory, which can be used to explain the "major transitions in evolution". Three well-known group selection models proposed respectively by Wynne-Edwards, Wilson and Traulsen were described to give readers a flavor of how the idea of group selection can be practically applied. Among the three models, Wilson's and Traulsen's represent two archetypal ways to change the selection dynamics between individuals and groups. In order to derive their differences in performance, and most importantly in order to identify the aspects that benefit group selection models most, these two models were investigated on the *n-player* prisoner's dilemma game under different parameter settings.

We conclude that maintaining variance between groups has great impact on group selection models. Specifically, avoiding a regularly mixing of groups or promoting changes in group size and composition helps to increase variance. These observations give us valuable insights into encouraging cooperation, and we shall use the same idea of group selection as in nature in evolutionary computation.

Chapter 3

Cooperation in Evolutionary Computation

Having discussed cooperation in evolutionary biology, this chapter focuses on the cooperation in evolutionary computation. To understand the necessity of introducing cooperation in evolutionary computation, in Sect. 3.1 we first briefly sketch two fundamental concepts: natural evolution and evolutionary computation (EC). Then we explain the evolutionary process in both nature and computational settings, through which the evolutionary difficulties of EC can easily be identified. To address these difficulties, Sect. 3.2 introduces Cooperative Evolutionary Algorithms (CEAs), a relatively young and growing branch in EC. An up-to-date review of CEAs is conducted with the purpose of providing an accurate picture of research trends in CEAs and pointing out limitations of existing CEAs.

3.1 Evolutionary Computation

Evolutionary computation (EC) [23] is a subfield of computational intelligence that abstracts key principles of natural evolution into algorithms for searching solutions normally requiring the traversal of a huge space of possibilities. The advantage of EC, when compared to traditional computational systems, is that it works well for problems which are usually highly nonlinear and contain inaccurate and noisy data [124]. EC has been successfully applied to numerous problems across a wide range of domains, such as bioinformatics, aerospace engineering, financial industry, robotics, machine learning and so on.

The field of EC encompasses a number of different classes of algorithms: Genetic Algorithms (GAs) [33, 43], Genetic Programming (GP) [4, 49], Evolution Strategies [84, 86] and Evolutionary Programming [26]. Although these different types of evolutionary methods were developed independently, their underlying ideas are similar and all inspired by evolution in nature.

Therefore, in this section we first introduce the working principle of natural evolution and its metaphor as an optimization process in Sect. 3.1.1. Next, we briefly describe a general framework of evolutionary computation, and a classical evolutionary algorithm in Sect. 3.1.2 and Sect. 3.1.3, respectively. Finally, in Sect. 3.1.4 we discuss evolutionary difficulties of classical evolutionary algorithms.

3.1.1 Natural Evolution As a Metaphor for Optimization

Darwin's finches are probably the best known, or most often cited proof of "evolution in action". Finches, depending on the different ecological niches they nest in, show

a great variety of beak sizes and shapes, each adapted to a specific food source; for example, some finches have developed large, sturdy beaks for cracking big seeds; some have tiny, pointy beaks for cracking small seeds or probing flowers and cacti; and some have thin, long beaks for poking into holes to extract grubs.

The different beaks observed in finches, or any physical characteristics, such as height, eye color, and hair texture, in all living organisms, are determined partly by the environment and partly by genes. Genes are made up of short segments of DNA, which are sequences of nucleotides lined up in a long linear string. The order of nucleotides in a gene carries genetic information, similar to how the order of letters carries information for words. This information is the instruction for building and maintaining a living organism. Genes are strung together and tightly packed into structures called chromosomes. At reproduction, offspring inherits chromosomes from their parents; to be more specific, for sexual reproduction, offspring receive half the chromosomes of the mother and half of the father (i.e., the recombination of parent chromosomes); for asexual reproduction, offspring receive the identical chromosomes as the parent. In this way physical characteristics will pass on from parent to offspring. The chromosomes of offspring also face the risk of being altered by the external environment or by errors during meiosis or DNA replication. Such changes are termed mutations. Taken together, heredity, reproduction and mutation explain why offspring often look like one or both parents, but still vary to some degree. These are mechanisms essential to ensure the variation of inherited traits within a population, and that, therefore, evolution will occur [66].

However, the adaptation of beaks according to available food sources needs an explanation from another powerful mechanism, called natural selection. Offspring

possesses similar, but not identical, genetic information or genotypes to that of their parents, and hence may express different physical characteristics, known as phenotypes. Although genotypes are a major influencing factor in the development of phenotypes, environmental conditions should not be ignored. The reproductive success (the fitness) of an organism is determined by interactions between its heritable phenotypic traits and the environment. For example, the finches with short, heavy beaks are unlikely to survive in an environment where grubs hiding in holes are the only food source. That is to say, traits, through interactions with the environment, will affect the chances of their bearers to survive and reproduce. Because traits are inheritable, beneficial traits also increases their own replication opportunities through the reproduction of their bearers, hence will become common within a population. On the other hand, detrimental traits will tend to decrease in frequency. Gradually organisms adapt to their environment; or we say natural selection produces adaptation in evolution.

Evolution by natural selection demonstrates an optimization characteristic. Evolution is responsible for the changes in the heritable traits of a population. However, which chromosomes parents will contribute at reproduction or which genes will undergo mutation is totally random. This randomness is good for exploring "gene space" and enables the identification of novel genes with new functions. It might lead to harmful new traits, but at the same time it also increases the possibility of discovering new traits which might help organisms to cope with new environments or conditions. Natural selection then judges whether the changes should be maintained or wiped out from a population. Consequently, it turns the random novelty into genuine, adaptive creativity. Once the improvements of heritable traits are integrated

into a population, they become the new starting point of a next generation: many random variations occur, most are discarded, and occasionally one is retained and propagated [66]. Ultimately, traits are optimized via evolution by natural selection to best suit the environment. The reproductive success (the fitness) of organisms in a population, therefore, is observed to increase over generations.

3.1.2 A General Framework of Evolutionary Computation

As can be seen from the discussion of Sect. 3.1.1, evolution by natural selection can be simply summarized as an iterated optimization process through heredity, reproduction, mutation and natural selection. This key principle can be easily extended to fields beyond biology. To computer scientists, it can be used as an "algorithm" for solving optimization problems, where a population of individuals, which are analogous to candidate solutions to a particular problem, undergo reproduction and random variation (recombination/mutation) under the selection pressure proportional to their appropriateness for the task at hand. The study of computational techniques based on, or inspired by, natural evolution is then called Evolutionary Computation (EC) [23]. Evolutionary Algorithms (EAs) covered by EC generally share the framework shown in Fig. 3.1.

Just as natural evolution uses chromosomes to carry genetic information, EAs use a special data structure also referred metaphorically as chromosomes to represent the proposed solution for a target problem. An EA starts with randomly instantiating the chromosome to obtain a set of candidate solutions. In the terminology of EC, we call this set "population" and candidate solutions "individuals". The performance of every individual is evaluated according to an explicitly defined fitness metric, called

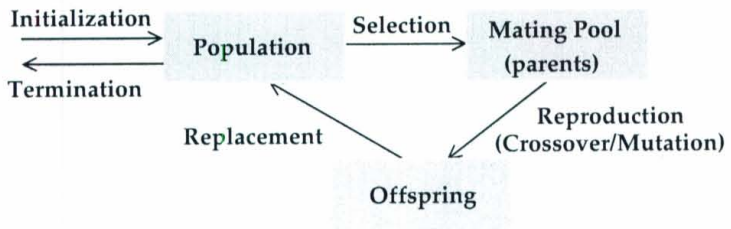


Figure 3.1: A general framework of evolutionary computation.

a fitness function. The fitness function assigns a numeric value to each individual measuring how good an individual is at the task. Given this fitness metric to be maximized, individuals are selected proportional to their fitness to enter the mating pool for reproduction (parent selection). Crossover (that is recombination) and mutation operators are applied on these parent individuals to generate offspring individuals. Based on their fitness, offspring compete with their parents for a spot in the next generation (survivor selection). This process is iterated until a solution is found or a limit on the number of iterations is reached.

Evidently, selection and variation are two fundamental forces that push evolution forward. Fitter individuals have greater chances to survive due to the selective pressure, and will reproduce more varied offspring. Offspring generated by crossover and mutation are biased towards regions of the search space where good solutions have already been discovered. As a result, the fitness of a population has a great chance to be improved over generations.

A primary advantage of EC is that it is simple in concept. It captures the gist of natural evolution but leaves many details out; after all, the primary goal is not to build a biologically plausible evolutionary model. In certain situations, EC is more efficient, compared with traditional search techniques, in that it involves search

with a "population" of possible solutions, not a single solution which might have to backtrack. It is also more robust and adaptive to dynamically changing environment. Other advantages of EC, such as easy parallelizability, independent representation, and smooth integration with other traditional optimization techniques, exist and are discussed in [26].

3.1.3 A Classical Evolutionary Algorithm

Having said that, evolutionary algorithms come in many flavors, including Genetic Algorithms (GA) [33, 43], Genetic Programming (GP) [4, 49], Evolutionary Strategies [84, 86], and Evolutionary Programming [26]. For most of the work that follows in this dissertation, GAs and GP are of particular interest. Both algorithms implement the general framework of EC, and at least follow the steps outlined in Algorithm 3. The algorithm completes in many iterations (also known as generations) during which selection and variation are applied repeatedly. The collection of all generations is termed a run. At the end of the run the algorithm will return the most highly fit individuals in the population as solutions. Since the general notion of an EA is clear from the discussion of the framework, we will restrict ourselves to exact definitions of representations, selection methods and variation operators used in a simple GA/GP system.

Representation The chromosomes of GAs are normally in the format of a finite length string over the binary alphabet {0,1}, as shown in Fig. 3.2. The chromosomes contains n genes, where n is the number of parameters to be optimized. Each gene contains several nucleotides which carry the binary encoding of the specific value of

Algorithm 3: A classical evolutionary algorithm

```
1  t ← 0;
2  P ← Initialize_Population(N);
3  Evaluate_Fitness(P);
4  while population does not converge or max generation is not reached do
5      for i ← 0 to m do
6          idv_1 ← Select_Parent (P);
7          if p < p_c then
8              idv_2 ← Select_Parent (P);
9              (idv'_1, idv'_2) ← CrossOver (idv_1, idv_2);
10             if p < p_m then
11                 Mutation(idv'_1, idv'_2)
12             end
13         end
14         Add_Individual(idv'_1, idv'_2, P');
15     end
16     Evaluate_Fitness(P');
17     P'' ← Survival_Selection(P, P');
18     P ← P'';
19     t ← t + 1;
20 end
```

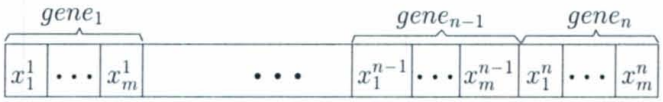
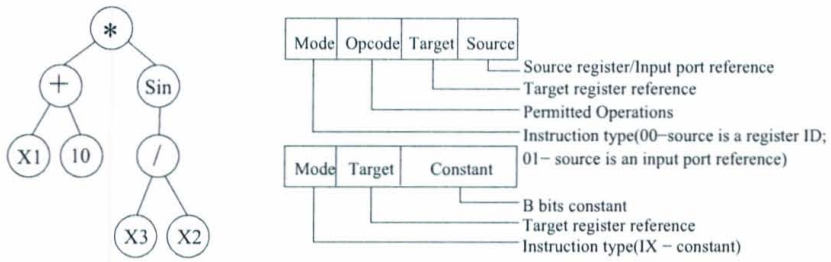


Figure 3.2: A chromosome in a GA contains n genes, where n is the number of parameters to be optimized. Each gene contains several nucleotides, which encode specific value of a parameter.

a parameter. Over the years, other types of encodings have been suggested, such as real values, categorical values, or the combinations of them [118].

Genetic programming automatically evolves computer programs, which originally were confined to expression tree structures, as illustrated in Fig. 3.3a. Functions, either arithmetic or logic, are located at the inner nodes, while variables and constants are at leaf nodes. The main limitations of tree-based GP are bloat and translation. The former refers to excessive tree growth [4], and the later refers to the translation at the fitness evaluation step from tree structures to symbolic expressions (S-expressions) in LISP [1], and then to instructions understood by computers. In order to boost performance, Linear Genetic Programming (LGP), another major branch of GP, evolves sequences of instructions written by an imperative programming language or a machine language [8]. As shown in Fig. 3.3b, instructions operate on one or two indexed variables (registers), or on constants from predefined sets. Therefore, individuals are manipulated and executed directly without requiring processing by an interpreter during fitness calculation.

[1]LISP is a programming language designed primarily for symbolic data processing used for symbolic calculations in differential and integral calculus, electrical circuit theory, mathematical logic, game playing, and other fields of artificial intelligence [62].



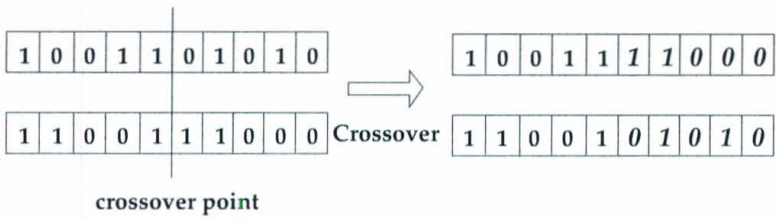
(a) Tree GP (b) Linear GP [91]

Figure 3.3: Chromosome structures of Tree GP and Linear GP.

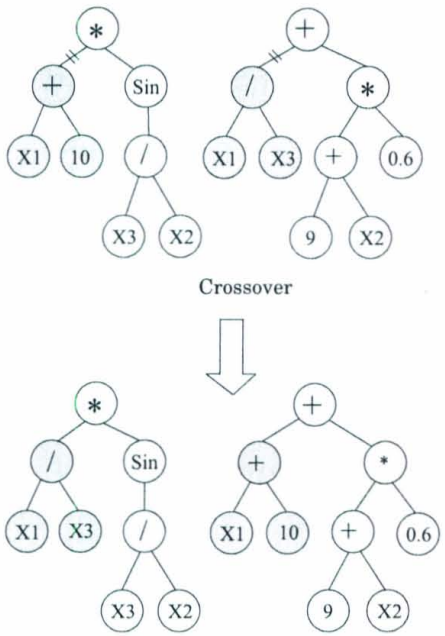
Parent Selection Selection is an important part of a GA/GP; it determines which individuals are eligible to produce offspring. Without selection directing the algorithm towards fitter solutions there would be no progress. Mimicking natural selection, the selection strategies are also based on the principle of survival of the fittest: fitter solutions are more likely to reproduce and pass on their genetic material to the next generation via their offspring. A number of popular selection techniques exist, including roulette wheel, tournament and ranking [4, 33]. Roulette wheel selection depends on a roulette wheel analogous to those found in casinos. Each individual is mapped to a slice on a wheel such that the size of the slice is proportional to its fitness value. A random number is generated and the individual whose slice corresponds to the random number is selected. Tournament selection chooses a number of individuals randomly from the population and selects the best individual from this group as parent. Ranking selection sorts the population according to the fitness values. A rank value is then assigned to each individual depending on its position in the sorted sequence. The selection probability is proportional to rank values. Ranking introduces a uniform scaling of fitness across the population.

Reproduction Both GA and GP systems apply crossover and mutation operators to introduce new individuals into the population. The use of crossover distinguishes GA/GP from Evolutionary Programming and early versions of Evolution Strategies, in which random mutation is the only source of variation. Crossover is performed with some probability defined by a crossover rate p_c. To be specific, if a randomly generated number p is smaller than p_c, two selected parent individuals will exchange parts of their chromosomes to create two new offspring individuals. The simplest and also the most commonly used form of crossover is one-point crossover. It randomly selects a crossover point and exchanges the substrings (GA), subtrees (Tree GP), or instructions (LGP) after the crossover point (as illustrated in Fig. 3.4). The examples shown here for Tree GP and LGP are also called homologous crossover, because the two parents share a common crossover point. Nevertheless, each parent individual has the freedom to select its own crossover point.

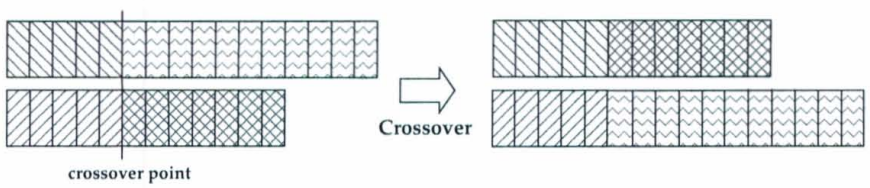
Mutation, when used in conjunction with crossover, ensures the population against permanent fixation at any particular locus and thus plays more of a background role [67]. It takes place after crossover, and randomly changes the new offspring with probability p_m, where $p_m << p_c$. Like crossover, mutation depends on chromosomal structure. Mutation in GA either changes every gene with a mutation probability p_m (as shown in Fig. 3.5a) or swaps the genes at two selected positions. In Tree GP, mutation replaces multiple selected nodes with new ones (see Fig. 3.5b for an example), or substitutes the subtree rooted at a selected node with a randomly generated subtree. Mutation in LGP has many variants [8], such as inserting a randomly created sequence of instructions to a selected position, deleting a selected subsequence of instructions, copying an effective instruction to a selected position, modifying the op-



(a) GA



(b) Tree GP

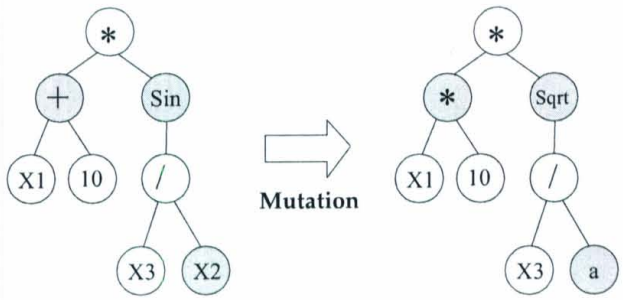


(c) Linear GP [91]

Figure 3.4: Crossover in GA, Tree GP and Linear GP.

erator or operands of a selected instruction, or exchanging two selected instructions. The selection of mutation points in all three chromosomal structures are random.

(a) GA



(b) Tree GP

Figure 3.5: Mutation in GA and Tree GP.

Survival Selection After reproduction, the algorithm holds two populations: the current population from which parents were selected (P) and the population of offspring (P'). Survival selection is required to determine which individuals of these populations can enter the next generation. The simplest method is to completely replace P with P', provided the two populations have the same size. Since reproduction is random, the GA/GP algorithm is at risk of losing the best individual in this step; that is, the fitness of the best individual in P' might be worse than the fitness of the best individual in P. To overcome this limitation, one option is to select the best n (population size) individuals from P and P'. Another option replaces parent indi-

viduals by their offspring only when offspring have better fitness than their parents. Either way guarantees the improvement in fitness.

For more complicated parent selection methods, crossover and mutation operators, and survival selection mechanisms, please refer to [4, 8, 67].

3.1.4 Evolutionary Difficulty for Classical EAs

Evolutionary computation is an exciting development in the field of computer science. Since its invention, EC has earned wide popularity for solving real-world problems across a spectrum of disciplines.

However, classical EAs are not a panacea; they are reported to be not entirely adequate for solving complex problems whose solution contains multiple subcomponents. One such problem, for instance, may require multiple individuals possessing different resources or functionalities to work collectively. For example, three robots, two equipped with infra-red sensors and one with light sensors, are required to approach a light source while avoiding collisions. Infra-red sensors would take care of obstacle avoidance, and light sensors would take care of phototaxis. The solution, of course, is to find at least two different movement strategies that take advantage of the specific equipment and at the same time to cooperate with others to move forward [100]. Another exemplary problem is one which is too complex and too large to expect a single solution to solve it effectively. For instance, in the concept learning task illustrated in Fig. 3.6, given a set of data examples (denoted by "+" signs), it is impossible for a single concept rule (represented as a circle) to cover all examples of the unknown concept (represented by shaded regions) at the desired generality and accuracy; normally a collection of rules is required.

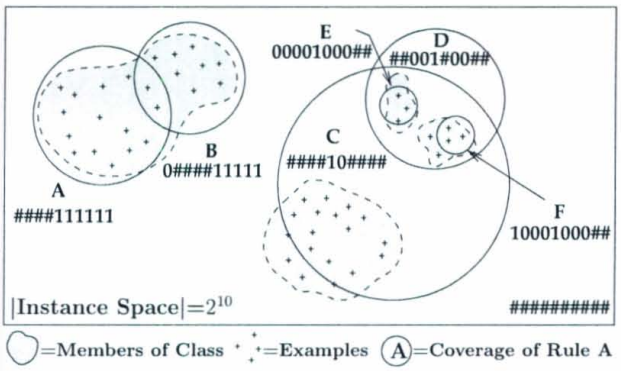


Figure 3.6: Concept learning as a set covering problem. The task is to find a small set of accurate rules represented as circles to cover the examples of the unknown concept represented as shaded regions [44].

One important reason that contributes to this failure is an implication of natural selection. In the same way that evolution in nature results from survival of the fittest, EC favors the fittest solution among a set of randomly varied ones. As a result, selection drives the evolving population toward a uniform or nearly uniform distribution of the fittest individual. That is to say, classical EAs have a strong tendency to converge to a single solution, which in respect to above examples could be only one movement strategy or one concept rule.

In addition, individuals, such as movement strategies or rules, are not independent. They **cooperate** with each other to provide the final solution. Each individual participating in the cooperation adapts and optimizes in the context of others. However, classical EAs evaluate individuals in isolation. Since the interactions between coadapted members of a population are not modeled, there is no evolutionary pressure for coadaptation to occur [81].

Therefore, it is necessary to consider extensions of classical EAs in order to solve

problems which require a set of cooperative individuals jointly to perform a computational task.

3.2 Cooperative Evolutionary Algorithm

Introducing the idea of cooperation into EC extends its ability to solve increasingly complex problems. It simplifies problems by dividing them into a set of solvable subproblems, and then concentrates on designing solutions for each subproblem. In a sense, it reduces the size of the search space and the search effort. This type of new evolutionary algorithm is called Cooperative Evolutionary Algorithm (CEA). In this section, we first discuss three critical issues that should be addressed by CEAs, which would also be useful features to be incorporated in CEAs. Keeping the three key issues in mind, we then review previous work relevant to the contributions of this dissertation. Based on the literature review, we will summarize the limitations of existing CEAs.

3.2.1 Cooperation in Evolutionary Computation

Prior to a discussion of cooperation in evolutionary computation, we would like to clarify that the complex problems discussed in this thesis are restricted to decomposable problems. A decomposable problem [103] is a problem that can be divided into subproblems, but the effect of changing a subproblem is a deformation of the fitness landscapes of other subproblems; as a result, the optimal solution to one subproblem may be different depending on the solution of other subproblems. Decomposable problems are different from separable problems, which can be divided into indepen-

dent subproblems; in other words, there is no interaction between subsolutions in the latter case.

Since the subproblems in a decomposable problem are highly interdependent, subsolutions need to work cooperatively. However, to allow coadapted subsolutions to emerge, CEAs need to address the issue of problem decomposition, evolution of collaboration, and diversity preservation.

Problem Decomposition Problem decomposition determines how to divide a complex problem into a set of simpler subproblems. As demonstrated in the empirical analysis in [22], decomposition can speed up an evolutionary process by 10 times.

The first thing to decide, of course, is the number of subproblems the problem should be decomposed into. After decomposition, individuals for solving each subproblem are assigned to the different search spaces. Since they are confined in their surrounding environment, they may exhibit different behavior and functionalities. Hence, problem decomposition should also decide the roles individuals play in a cooperation. For some problems, this information can be easily identified *a priori*. Consider the task of requiring three robots moving together: two subsolutions are enough, one for sensing light and one for sensing obstacles. For some other problems, we may have little or no information available for deciding either the number or the roles of subsolutions in the decomposition. Take concept learning as an example, it is impossible to tell beforehand how many rules are needed to cover an unknown concept, or which region each rule will cover. Therefore, it is the responsibility of the CEA to address problem decomposition as an emergent property [81].

Problem decomposition also affects population structure. Because subsolutions may possess unique roles or experience different external environments, they may demand different chromosomal structures in terms of size, data types or input constraints. Therefore, they can no longer be evolved in one population, multiple subpopulations should be considered. This also affects how crossover and mutation are conducted.

Evolution of Collaborations Cooperative EAs are similar to classical EAs in the sense that they evolve individuals in one population. Individuals which represent partial solutions compete with others for their own survival chances. Cooperative EAs are also different from classical EAs, as they return a set of individuals as solution, not a single individual. This set is known as a collaboration.

The returned collaboration is selected from the population, and hence is a subset of the population. Obviously, the decision of which individuals should be selected into the collaboration directly affects the quality of a solution. The simplest method is to consider individuals who perform their roles best. This strategy, however, can sometimes be too greedy and potentially results in poor performance. A good analogy to explain the problem is the all-star team phenomena. A sports team composed of the best individual players from a whole league may not necessarily beat the best team in the league. This is because the best team is good by virtue of its member's abilities to cooperate.

Once collaborations are formed, their evolution should be considered. Just like individuals are optimized through evolution, the performance of collaborations should be optimized through evolution, too; after all, the aim of CEAs is to search for

the best performing collaboration, not the best performing individual. Because of this change of the evolutionary objects from individuals to collaborations, genetic operators, mainly crossover and mutation, have to be re-defined, accordingly. In addition, we need a metric to measure the performance of whole collaborations, or how well a set of coadapted individuals can cooperate as a single solution. One advantage of considering such a metric is that it indicates evolutionary progress of collaborations, and guides the search towards optimal collaboration performance. Another advantage is that it puts extra constraints on members in a collaboration. In order to achieve high collaborative performance, members have to give up some of their own interests especially when individual interests are in conflict with the collaboration interest, as such conflict will compromise collaboration performance. This extra pressure will also force individuals to search different areas and to develop different and unique roles.

Diversity Preservation Diversity is critical to the success of the evolution of cooperation. On the one hand, diversity needs to be preserved in the population long enough so that algorithms are able to explore the search space exhaustively. On the other hand, diversity promotes the formation and maintenance of stable niches occupied by different subsolutions [57]. The existence of different niches provides the overall evolutionary process with basic building material, from which the most suitable pieces are selected to compose the final solution.

Individuals, depending on their roles, will usually contribute differently in a collaboration. Unfortunately, such contributions are not reflected in their fitness values, as individual fitness only indicates how well individuals perform their own tasks. This is an evident gap between individual fitness and individual contribution to the overall

goal. It is highly probable that a subsolution with a comparatively low fitness, but unique contribution to the collaboration, is at risk of being eliminated by selection. A feasible way to protect such individuals is to offer rewards or penalties on fitness according to the contribution of each individual in cooperation. This is called credit assignment. Credit assignment is another way to protect diversity, as it encourages individuals to develop various roles with unique contributions.

Please note niching and credit assignment are two different ways to preserve diversity. Credit assignment requires external feedback from the environment, while niching depends on internal competition.

3.2.2 Related Work

Keeping these three key issues in mind, we will now review related EC approaches and algorithms that have been proposed to evolve coadapted subsolutions. Through this discussion, we hope to answer the following questions: what are the main features of approaches that lead to the emergence of cooperation in EC? Are there any limitations in these approaches? If so, what are they?

Based on how final solutions are presented, CEAs can be categorized into population-based approaches and team-based approaches.

Population-based Approaches In population-based approaches, the entire population becomes the solution of targeted problems. Examples of this type of approach include Learning Classifier Systems (LCS) [42] and niching-based methods [28, 46, 89].

LCS is a rule-based system for concept learning, which employs reinforcement

learning and a classic GA to evolve a set of binary encoded rules[2]. Reinforcement learning is similar to credit assignment, which adjusts individuals' fitness according to the feedback from input data. This feedback is in the form of numerical values that reflect the errors between the output predicted by rules and the expected output. The greater the error, the less the reward. Rewards are distributed among rules which are involved in prediction and are accumulated into a fitness score which later affects a GA at discovering new rules. In the end, individuals in the population will be specialized in response to different aspects of the input data.

Niching methods are normally embedded in classical EAs as an operator, whose original purpose is to control and prevent unbalanced proliferation of genotypes. Their inspiration stems from niches found in nature. Individuals competing for the same set of limited resources reside together as a niche. The localization of competition in niches actually implies a simple and indirect form of cooperation that allows complementary species to coexist and diverse ecosystems to thrive. Similarly, niching methods in EC penalize the fitness of individuals based on their genotypic or phenotypic similarities, thus forcing individuals to explore and reside in different parts of the search space. As a result, multiple distinct individuals that act of indirect cooperation can be produced in a single run. Formally proposed niching methods include crowding [19], deterministic crowding [56], fitness sharing [35], implicit sharing [28, 89] and resource-based fitness sharing [45, 32].

These two methods are perfect examples of how to apply credit assignment or niching to preserve population diversity. However, one of the drawbacks of population-based approaches is that they are not measuring the performance of subcomponents

[2]LCS in this dissertation refers to Michigan-style LCS.

as a whole. Without such a measurement, evolved individuals are not sufficient to consistently provide cooperative behavior, and the completeness of final solutions cannot be guaranteed [93, 117].

Team-based Approaches Team-based approaches overcome the disadvantage of population-based approaches by introducing the idea of teams[1]. Well-known team-based methods include GP Teaming [7], Cooperative Coevolutionary Evolutionary Algorithms (CCEA) [81], Individual Evolution (IE) [11, 79], Orthogonal Evolution of Teams (OET) [93], and Symbiotic Bid-Based Genetic Programming (SBB) [53, 54].

GP Teaming [7] has an explicit team representation: a population is subdivided into demes, which in turn are subdivided into fixed equal-sized teams of individuals. Both team and individual fitness are defined, but only teams are regarded as the objects of evolution: the members of a team are always selected, evaluated and varied simultaneously. The strong coupling between teams and their members eliminates the credit assignment problem, but it also misjudges the contribution of team members; as a result, good team members are at the risk of loosing reproductive opportunities because they might be teamed with free-riders or less fit individuals. Free-riders in the context of cooperation represents individuals who contribute little or nothing to the public good.

Cooperative CoEvolutionary Algorithms (CCEAs) [81] evolve each subsolution in a separate subpopulation without genetic exchange, except for fitness evaluation. To obtain the fitness of an individual, collaborators from other subpopulations are selected to form a complete solution (analogous to a team). The fitness of the com-

[1]Teams, groups and collaborations are interchangeable terms in this dissertation. They all refer to a collection of subsolutions.

plete solution (collaborative fitness) is evaluated, and returned as the fitness of the individual being evaluated. Using collaborative fitness as individual fitness can be problematic. The fitness of each individual might be incorrectly estimated due to the impact of other individuals in the collaboration. In addition, only based on the changes of fitness values, it can be really hard to reveal what the system is really doing; this is the so called Red Queen effect [106]. The other disadvantage of CCEAs is that fitness evaluation is completed on the team level, but selection is done on the individual level. Such a mismatch drives team members towards cooperation rather than optimization. To overcome the limitations, 't Hoen and de Jong enhanced CCEAs by a COllective INtelligences (COIN) framework [95]. COIN introduces a private utility function (i.e., an individual fitness function) and defines conditions that a private utility function has to meet, so that the optimization of the private utility functions leads to an increase of team performance.

Individual Evolution [11, 79], also known as the Parisian approach, evolves individuals in a single population. In each generation, only one team is formed by N individuals, which are selected deterministically (i.e. the best N individuals) or stochastically from the population. Fitness functions are defined for both individuals and the team. Individual fitness guides the evolution to optimize individuals. However, team fitness is used to adjust individuals' fitness depending on their respective contribution to the team. Individuals who improve team fitness will be rewarded; otherwise, they will be penalized. Fitness sharing is used to promote diversity in populations.

Orthogonal Evolution of Teams (OET) [93] treats a population in two ways: as a single population of M teams each with N members, and also as N independent

islands, each island for one particular type of member. Selection pressure is applied to both individuals and teams. To be specific, two individuals are selected from each of N islands. The $2N$ individuals will form two teams, which after crossover and mutation will produce two offspring teams to replace another two existing teams with worse performance. Individuals have to perform well in order to be selected as parents, and they also need to cooperate well with others to obtain high team fitness to prevent replacement. As confirmed by experiments, selection pressure on both levels optimizes the performance of individuals and teams.

Symbiotic Bid-Based (SBB) Genetic Programming [53, 54] exploits two populations: a symbiont population which contains individuals evolved by LGP and a host population which contains teams composed of individuals selected from the symbiont population. On the host level, three combinatorial search operators, which delete, add or change individuals from a selected team, are applied to search for effective individual combinations. Because the size and composition of hosts is always changing, SBB in fact tests the number of symbionts required for a host to accomplish the task at hand, which indirectly addresses the automatic problem decomposition. In addition, SBB is more efficient than most other CEAs in a sense that the search for the best host is conducted simultaneously on a set of hosts. Evolution of symbionts is driven by changes that occur on hosts: every time an individual in a host is selected for change, mutation happens, which will delete, add, change or swap instructions with a predefined probability in a symbiont's program.

Given the discussion above, it is clear that the use of teams explicitly expresses cooperation. All team-based approaches evaluate team performance according to a fitness function. This fitness function, similar to an individual fitness functions, will

indicate in which generation which team performs best, thus successfully guiding the search to approach the best team. It also puts pressure on individuals inside a collaboration, forcing them to develop different functionalities in a collaboration.

Hierarchical Approaches A desired property of CEAs is to allow collaborations to emerge through evolution. SBB reaches this goal by mutating the composition and size of teams. Another feasible way is to build collaborations using a bottom-up process: starting from simple basic elements, large complex components are constructed in a recursive fashion until the desired collaboration emerges; this in fact describes a hierarchical process for the construction of collaborations.

The first known hierarchical evolutionary algorithm was the Messy Genetic Algorithm (mGA) proposed by Goldberg *et al.* [34]. mGA is "messy" because it uses a messy coding: variable-length strings containing variable numbers of genes from the chromosome with respect to the problem being solved. Because of possible changes in the representation, the usual crossover operator can no longer be used. Instead, two messy operators, cut and splice, are implemented for this purpose. Cut divides an individual into two, while splice concatenates two individuals to one. The workflow of the mGA can be summarized as the partially enumerative initialization combined with two phases of selection: a primordial phase and a juxtapositional phase. Partially enumerative initialization provides all possible building blocks of a solution. The primordial phase is then executed to reduce building blocks to useful ones, whose combination will create optimal or near optimal individuals. The juxtapositional phase resembles the usual process of classical GAs by repeatedly invoking cut, splice and other genetic operators with certain probabilities to gradually build solutions from

useful building blocks. Another algorithm, called Hierarchical Genetic Algorithm, proposed a similar hierarchical framework but only used "splice" operators. For more details, please refer to [18].

The idea of composing complex structures out of simpler ones is analogous to the natural process of symbiogenesis, which creates new species from the genetic integration of symbionts. Watson and Pollack proposed the Symbiogenic Evolutionary Adaptation Model (SEAM) [104]. The heart of this model is to introduce a symbiotic combination operator to combine two individuals of arbitrary length to create a new offspring. Offspring will replace both parents if it dominates them, which indicates that the newly combined individual is a confirmed, better building block and can serve as a new start for future composition.

The Evolutionary Transition Algorithm (ETA) proposed by Lenaerts *et al.* [51] is another algorithm using the concept of symbiosis, but also embodies the concept of transition. Unlike SEAM in which an offspring replaces its parents immediately, ETA introduces an intermediate step, called induced phenotype. The induced phenotype is constructed by combining the genotypes of two individuals in a symbiotic relationship. Individuals are reproduced in three different ways. First, individuals will be selected and reproduced as in classical GAs. Second, in order to maintain useful links between individuals (i.e. individuals with good induced phenotype), both individuals and their symbiotic partners are replicated. This is a step toward transition. Finally, the real transition happens; if the fitness of the induced phenotype of an individual exceeds a predefined threshold (i.e. the induced phenotype is good enough), the induced phenotype will be upgraded to the genotype of a new individual at a higher level complexity.

In summary, hierarchical approaches exhibit the basic cooperative trait: putting independently evolved chromosome segments together to form a single solution. "Basic" here means that coadaptation between entities may not required. In addition, because they evolve segments of the chromosome, they have a special way to define and evaluate fitness. Nevertheless, the hierarchical method to construct complex solutions out of simpler ones may shed light on how to improve the problem decomposition ability of CEAs.

3.2.3 Limitations of Current CEAs

Table 3.1 summarizes features of the CEAs discussed in Sect. 3.2.2. The necessity to incorporate diversity preservation into CEAs is obvious, as all listed algorithms employ either niching or credit assignment (or both) to promote coexistence of individuals playing different roles in the population.

Team-based approaches generally outperform population-based approaches [54, 53, 93, 117]. The main reason is that team-based approaches agree on introducing teams as a new type of entity to represent a solution. As can be seen from Table 3.1, team fitness is defined in all team-based approaches. Team fitness models the interaction between coadapted subsolutions, and encourages individuals to cooperate [93]. Optimizing this measurement will produce highly fit teams. To be more specific, in the context of EC this optimization means team evolution. Please recall that in Sect. 3.2.1, we analyzed the importance of the evolution of collaborations, and suggested it as one of the desired features of CEAs. However, only a few of the algorithms take advantage of this measurement to optimize team performance. GP Teaming is the only one that considers team fitness and team evolution. Although OET and

Table 3.1: Comparison of CEAs in the literature

		Fitness		Evolution		Diversity		Problem
		Idv	Col	Idv	Col	Nch	CA	Decomposition
Population	Niching	√		√		√		Semi-auto
	LCS	√		√			√	Semi-auto
Team	GP Teaming		√		√	√		Manually
	OET	√	√	√			√	Manually
	CCEA		√	√			√	Manually
	IE	√	√	√		√	√	Manually
	SBB		√	√		√		Auto

Notes: Idv=Individual, Col=Collaboration, Nch=Niching, CA=Credit Assignment

SBB[2] test team fitness at survival selection, in our opinion these two algorithms are not performing team evolution. The reason is that selection of teams, including survival selection, does not equal to the evolution of teams. It does not result in good teams being prioritized for propagation, and does not exploit useful building blocks (good combinations of subsolutions) in existing teams.

To evolve cooperation, individuals not only need to exhibit the ability to cooperate, but must also be relatively successful at accomplishing their own distinct subtasks [93]. This implies that the evolution of individuals should be considered as well. Indeed, most CEAs include the evolution of individuals. However, they seem to not all

[2]Another reason that SBB is not considered to conduct team evolution is the uniform probability distribution used to select hosts for reproduction. Such a selection scheme does not correlate the chances of reproduction with fitness.

agree upon whether the performance of individuals should be accurately evaluated. For example, CCEAs use team fitness as individual fitness. As confirmed by previous studies [95, 93, 92, 106], with only a team fitness being defined, evolution will result in good teams but relatively poor team members, which prevents further improvement of team performance. Furthermore, treating individuals independently, such as evaluating or evolving them separately, will increase the flexibility and efficiency of algorithms. As shown in OET, optimized individuals can be reused to construct teams, which saves both computational resources and time.

From Table 3.1 we also notice that most CEAs deal with problem decomposition manually; the numbers or roles of subsolutions are determined *a priori*. LCS and Niching are marked as "semi-auto", because the number of subsolutions is not specifically defined, but is always confined by population size. Redundant or duplicate individuals are very likely to be found in solutions; normally a post-processing step is required. SBB is the only CEA which can automatically decompose problems without *a priori* knowledge. However, the way SBB changes team composition is rather stochastic: teams to be changed are randomly selected, and individuals to be deleted or added are also randomly selected. In addition, individuals are the only objects that can be added or deleted in teams. Apparently, SBB does not make good use of existing teams as potential building blocks. Such teams may not serve well as final solutions, but they might have high potential for containing valuable combinations of individuals; otherwise, they will be eliminated from the population. Again we argue that both, individuals and existing teams, should be regarded as reusable modules. Simpler and smaller modules could be reused to form larger modules with increased complexity. Through such a hierarchical method of constructing solutions,

as described in hierarchical approaches, the appropriate number of subcomponents in a solution might be automatically decided. That is to say, problem decomposition could be achieved through a bottom-up process.

In conclusion, all CEAs discussed in this section adapt various forms of diversity preservation mechanisms, but none of them satisfactorily addresses evolution on both individual and team levels and the automatic problem decomposition, especially in a hierarchical way.

3.3 Chapter Summary

Nature is a rich source for inspiration; one of the most fascinating one is how evolution shapes today's world. Briefly, natural evolution can be described as the changes, influenced by natural selection, in the heritable features within a population of reproducing individuals over generations; the consequence of natural evolution is that individuals become adaptive to their environment. This optimization character is appealing to computer scientists as it provides a feasible metaphor for solving optimization problems in computational settings.

The study of abstracting key principles of natural evolution into algorithms is called Evolutionary Computation (EC). Algorithms investigated in EC are normally employed to solve problems "involving chaotic disturbances, randomness, and complex nonlinear dynamics — that our traditional algorithms have been unable to conquer" [27]. Nevertheless, they are not entirely adequate for solving problems whose solution is in the form of interacting coadapted subcomponents [81]. Therefore, a type of new evolutionary algorithms called Cooperative Evolutionary Algorithms (CEAs) was

proposed.

CEAs evolve individuals representing potential subsolutions in one population. At the end of evolution, a set of individuals is returned as solution. Based on how final solutions are presented, CEAs are classified as population-based approaches or team-based approaches. Either way, they have to address the issues of (i) problem decomposition, (ii) evolution of collaborations, and (iii) diversity preservation. Those issues are consistent with the ones suggested by Potter and de Jong in [81], but with some extensions. Team-based approaches generally perform better than population-based approaches [53, 54, 93, 117], because collaborations are explicitly defined and evaluated. As a consequence, interactions between subsolutions are modeled, which promotes the emergence of cooperation. Unfortunately, none of the existing CEAs considers evolution on both individual and team levels to optimize their performance, which leads to either highly fit, but non-cooperative individuals or good collaborations with poorly performing individuals. We also found another limitation of current CEAs, their lack of ability to automatically decompose problems. The "composition" operator introduced in hierarchical approaches may shed light on how to build complex solutions out of simpler ones in a hierarchical way, with problem decomposition solved automatically.

Chapter 4

A Hierarchical Cooperative Evolutionary Algorithm

In Chapter 2, we introduced mechanisms suggested by biologists, especially the theory of group selection, to explain the evolutionary emergence of cooperation among unrelated individuals. In Chapter 3, we discussed evolutionary difficulties of classic Evolutionary Algorithms (EAs) and the limitations of Cooperative Evolutionary Algorithms (CEAs) when applied to search for multiple coadapted subcomponents in the solution of a targeted problem. In this chapter, we will propose a new evolutionary computation model to overcome the limitations of CEAs by going back to inspirations from nature. The motivation will be presented in Sect. 4.1. In Sect. 4.2 we will propose a new multilevel selection model to support the evolution of cooperation from the bottom up. Sect. 4.3 will focus on a hierarchical cooperative evolutionary algorithm which implements the model we propose. In Sect. 4.4, we will justify how our new model overcomes the limitations found in CEAs, and provide a few potential problem domains to demonstrate in examples how our model works.

4.1 Motivation

As we have said in Chapter 3, cooperation is often required in evolutionary computation to solve real-world problems. These problems often are too complex or too large to expect a single solution to solve them effectively, or sometimes they are structured in a way that a single solution cannot reasonably possess all necessary ingredients to solve every single subproblem. We want that a cooperative evolutionary algorithm should be able to solve such complex problems by evolving solutions in the form of interacting coadapted subcomponents, which are emerged from evolution rather than being designed manually.

In nature, cooperation has been observed everywhere. Through cooperation, individuals are able to increase their survival rate, or accomplish things they cannot reach individually. Mechanisms adopted by nature allow individuals who are engaged in cooperation to coadapt in spite of competition imposed by evolution, and to mediate conflicts of interest between individuals and their collaboration. Therefore, we have an existence proof of the evolution of cooperation. Among the mechanisms suggested by biologists (see Chapter 2 for details), group selection[1] is chosen to bring cooperation into evolution in computational settings because for two reasons. First, group selection theory unifies other alternative theories to explain the evolution of cooperation, such as kin selection and reciprocation [76]. Second, only group selection theory explicitly organizes individuals into a structure (i.e., groups), which is analogous to the collaboration structure required in CEAs.

Wilson's and Trauslen's group selection models demonstrated specifically how to

[1]Please note that it is not necessary for readers to agree with this theory as an explanation of biological phenomena in order to take advantage of their implications in artificial settings.

apply group selection theory in practice. However, when adapting their models to an EA context, one can notice right away that both models lack flexibility in terms of problem decomposition; individuals are always dispatched into groups of predefined size. Multilevel selection extends group selection from two levels to multiple levels, where levels are like "Russian matryoshka dolls" [114] nested one within another. If the bottom-up construction process is revealed, from previous discussion we know the issue of automatic problem decomposition would be resolved. However, it is not completely obvious how the two group selection models can be extended from the two-level structures to multiple levels, and how the two models explain the creation of hierarchical structure.

To address cooperation and automatic problem decomposition simultaneously, a possible alternative is to introduce into group selection models a new function similar to the symbiotic combination operator discussed in the hierarchical approaches of CEAs. This new function would be responsible for constructing hierarchical structures out of the most basic elements in a bottom-up fashion. However, the resulting hierarchical structures raise new questions, such as how many levels should be constructed and on which level should group selection apply.

Banzhaf [3] discussed the relationship between cooperation and competition in a simple artificial chemistry system. In that system, lower level entities are bonded together as a group by cooperative interactions. When such a group, which we can term a new entity, competes with less cooperative entities from lower-levels, it will take over the population in the end. The larger the difference among competing entities, the quicker the competition is settled; for example, a population with a group of 3 autocatalysts and 4 individual autocatalysts converges faster than a population with

2 groups of 3 each and 1 individual autocatalyst. The reason is that entities on different levels are allowed to compete against each other. This design leads to a very interesting extension. Suppose the fitness of entities now no longer depends on their size, but rather on how they maximize a specific goal of a problem; therefore, a group at a higher level would not necessarily have a higher fitness. When groups on different levels compete with each other, groups with higher fitness—regardless of their level—will be favored by selection, and hence are more frequently selected. In a sense, the level at which the selection should act is totally determined by the fitness of entities. We know evolution is parsimonious; higher level groups with lower fitness will be unstable, and will be eliminated from the population by competition. This is the reason to assume that hierarchies will not grow exponentially, but will stop growing at the most appropriate level required by the nature of the problem. Once the most stable level is decided, the best group structure will be found. That is to say, the problem decomposition is addressed at the same time.

In summary, the idea of group selection can be applied to encourage cooperation and adaptation; a function similar to the symbiotic combination operator along with the idea of selecting between levels described in [3] hierarchically creates a sophisticated solution out of simpler ones without predefined problem decompositions. Incorporating these three elements in Evolutionary Computation leads us to the possibility of inventing a new computational model for evolving solutions for decomposable problems, in which cooperation and problem decomposition will emerge as a result of evolution.

4.2 A Computational Multilevel Selection Model

With this motivation, we propose a new computational multilevel selection model, shown in Fig. 4.1.

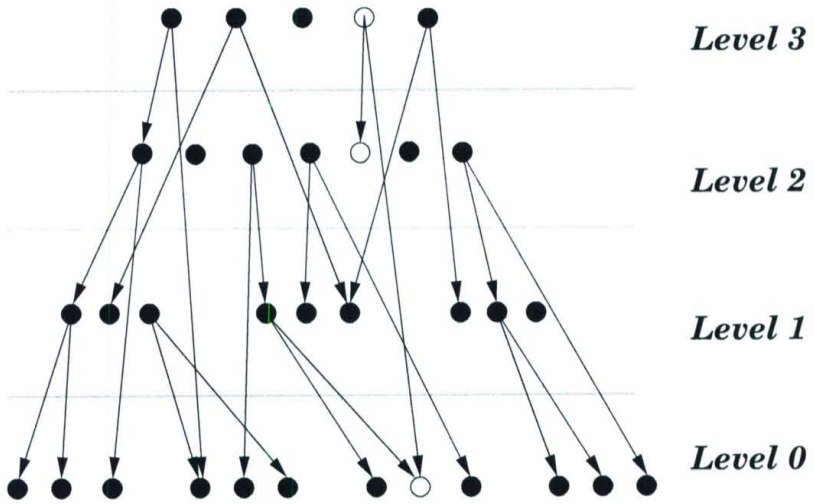


Figure 4.1: A new hierarchical model, which considers selection not only between individuals, but also between groups and between levels.

This model contains two types of entities. One is individuals, denoted by dots on level 0. Individuals are the most basic elements to compose the final solution of targeted problems. Individuals, for instance, can be circles in the concept learning problems or movement strategies in the robots coordination problem. They are independent, without being aware of the collaborative goal. Apparently, there is no cooperation at this level. The other type of entities are groups, represented by dots on level 1 and higher. They are compositions of existing individuals or groups.

Initially, only individuals exist in the framework. Groups and new levels are

created dynamically by a new operator called "**cooperation**". This operator utilizes the ideas of the symbiotic operator and the selection between levels [3] but with some extensions. When it is applied, individuals and groups on all levels, if any, are mixed together, from which two entities, either groups or individuals, are selected proportional to fitness to form a new group. For example, as highlighted in Fig. 4.1 by white circles, an individual on level 0 and a group on level 2 can cooperate to form a new group on level 3.

Hierarchy in the living world can be classified into two major types of biological hierarchies [61, 101]. One type is constitutive, in which entities are physically joined to each other within each level, as cells within a tissue; the other is aggregative, in which individuals are simply associated in a series of increasingly inclusive entities, such as organisms in a population. To accommodate aggregative hierarchies, reaction rules are introduced in cooperation operator. These rules, akin to chemical reaction rules, describe specifically under what conditions and what types of lower-level entities can be transformed to what types of entities on higher levels. As a result, groups normally have genotype definitions totally different from individuals. Unless specified, the default hierarchy in the model is constitutive.

Groups, once formed, will exhibit phenotypic traits. Group traits can be the same as individual traits, such as the cooperative trait shown in groups and individuals of Wilson's and Trauslen's models discussed in Chapter 2. At other times, group traits can be distinctive to individuals'. The transition from individual traits to group traits is possible, if group fitness is no longer defined proportional to or related to the average individual fitness [76]; that is to say, groups exhibit different behaviors when compared to individuals.

Groups in this framework are independent entities with their own fitness definition and heritable traits. Therefore, evolution should happen on group levels as much as on the level of individuals. Crossover and mutation are another two evolutionary operators defined on groups, besides the cooperation operator. Together the three operators, under the guidance of group fitness, explore and exploit promising regions in the group searching space, aiming to find new and better combinations of individuals, through combining, exchanging, adding or deleting individuals in groups. During evolution, the three operators take advantage of groups that already exist in the population. Those groups have passed the test of selection, so they are possibly good building blocks containing valuable combinations of individuals. By sampling, recombining or changing those good partial solutions, the possibility of constructing better groups with higher fitness is raised. Compared to always manipulating individuals, reusing existing groups in population accelerates evolution, as stated in [3].

Evolution on groups introduces selection pressures on group levels, which promotes fierce competition between groups. Only groups with good performance are able to seize the opportunity for future reproduction and cooperation. It also introduces selection between levels, as the three operators make their selection on group from all levels. Like mentioned before on page 77, the selection between levels makes sure that a new level emerges through evolution only when groups on this new level have an advantage in fitness.

The benefits of considering evolution on group levels are plain to see; it constantly optimizes group performance; therefore, it accelerates the overall search process and improves solution accuracy [93]. At the same time, it controls the growth of hierarchical structures, and addresses the issue of problem decomposition.

Individuals are the most basic building blocks in our model. In order to build better groups, they have to be optimized in the first place. This optimization is accomplished by evolution on the level of individuals, which is similar to the evolutionary process described in classical evolutionary algorithms, with one exception. Instead of selecting parent individuals directly from the individual level (i.e., level 0), our model first selects a group proportional to fitness from groups on all levels, from which an individual is selected as parent. Parents are crossed over or mutated to produce new individuals. Obviously, the idea of group selection applied here; the survival of an individual is now associated with the performance of its group. This implies that individuals have to give up their own interests and start to cooperate with others for the sake of striving for better group performance. Sometimes, individuals engaged in cooperation need to specialize on different roles. Group selection ensures that even though such roles are not assigned or unknown *a priori*, they will emerge through evolution because of the selection pressure on groups.

Our model requires that the fitness of individuals should be explicitly expressed. According to [92, 116], only considering collaboration fitness will lead to relatively good team performance, but the members themselves are relatively poor, which constrains a team's performance. One may argue that this is a limitation, as not every problem can be decomposed into subcomponents whose fitness can be easily evaluated. A possible workaround for such situations is to estimate individual fitness by using individual contributions. As suggested by Wolpert *et al.* [116], individual contribution can be calculated by first evaluating how a group would have performed if that individual was removed from the group and then assigning the resulting difference as the contribution of that individual. Since the contribution of individuals

greatly depends on whom they cooperate with, we may obtain different contributions for an individual by choosing different groups. Therefore, to estimate the fitness of an individual as accurately and fairly as possible, we can use the average individual contributions in all participated groups or use the individual contribution in the best performing group as individual fitness.

4.3 A Hierarchical Cooperative EA

The evolutionary algorithm implementing the above framework is shown in Fig. 4.2. In the initialization step (step 1 in Fig. 4.2), N_1 individuals are randomly generated and have their fitness evaluated. Reproduction on group levels (step 2 in Fig. 4.2) creates N_2 new groups every generation by applying the three evolutionary operators, i.e. cooperation, crossover and mutation, with a user-defined probability. Cooperation selects participants from both individuals and groups (i.e., from level 0 and above), while the other two operators only select from groups (i.e., from level 1 and above). Any selection schemes that are relevant to group fitness can be applied here, such as roulette wheel selection, tournament selection or ranking selection. Cooperation composes a new group from the selected entities according to appropriate reaction rules. Crossover exchanges individuals in two groups; one-point, two-point, homologous or even user defined crossover can be applied on groups. Mutation adds, removes (only when group size is greater than 2), or replaces individuals in selected groups. Once a new group is created, its group fitness and its validation are evaluated.

Although the selection between levels controls the growth of groups, an extra step of validation is necessary because of the existence of free riders. Free riders increase

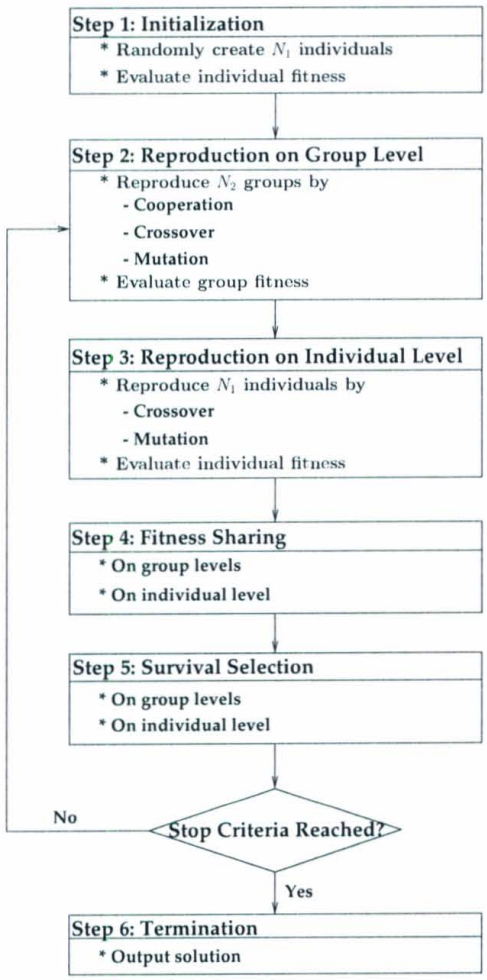


Figure 4.2: Outline of the hierarchical evolutionary algorithm with a population of N_1 individuals and N_2 groups.

group size without changing group fitness. Validation makes sure that every member in a group has a unique contribution towards the cooperative goal, which can either be checked explicitly or be considered as a part of group fitness. Groups should be penalized in fitness if they have the same performance as other groups but with a larger group size.

To produce offspring on the individual level (step 3 in Fig. 4.2), parent individuals are selected for crossover and mutation by the two-step procedure described in group selection theory. If no groups are currently available in the population, parents are selected directly from the individual level based on fitness. Any types of crossover and mutation pertaining to individual representations can be conducted on these selected parents. The fitness of N_1 newly produced individuals is evaluated. Please note that this algorithm evolves N_1 individuals and N_2 groups separately. The reason to keep a constant number of individuals in the population is because they are the most basic building blocks. Only when individuals have fully exploited their local environment, or have maximized their fitness, will it be possible to find optimal groups.

Preserving diversity (step 4 in Fig. 4.2) is mandatory on all levels, because the algorithm needs to maintain a set of different partial solutions so that all required subcomponents can be present in the final solution. Various niching mechanisms can be used, such as crowding, fitness sharing, implicit sharing, resource sharing or even user-defined niching schemes.

After the iteration on individual and group levels, the number of individuals and groups in the population was doubled. Therefore, new groups have to compete with groups in the current generation for N_2 positions in the next generation (step 5 in Fig. 4.2). Many survival selection strategies can be applied here, such as always

selecting the best N_2 groups among new and existing groups, or replacing parents with their offspring if offspring have better fitness. The same applies to new and existing individuals, but they compete for the N_1 positions in the next generation. Because individuals on level 0 are the most basic building blocks, they can participate in composing more than one group at different levels. When an individual is replaced by another, the changes can either be updated in all groups that contain this individual or not, depending on design.

The above steps (step 2 $\sim$ step 5) will be repeated until a predefined termination criterion is reached, e.g., the maximum number of generations, or a desired fitness, or accuracy. The algorithm finally will return the best performing group as the solution (step 6 in Fig. 4.2).

In summary, this new Hierarchical Evolutionary Algorithm (HEA) can be used to search multiple coadaptive subcomponents in a solution; it extends classic EAs by introducing group selection and evolution on group levels. Group selection favors individuals who cooperate and contribute in a group. Evolution on group levels optimizes groups, which in turn should help evolution on the individual level. In addition, because of the cooperation operator and the selection between levels, this algorithm is able to build solutions hierarchically, and decides the most appropriate depth of hierarchies and the size of a collaboration without human interference.

This algorithm only sketches a general workflow. Therefore, it fits well with many variations of evolutionary algorithms, such as Genetic Algorithm and Genetic Programming. For someone who wishes to apply this algorithm, they have the freedom to decide how to represent individuals and groups, how to measure individual and group fitness, and how to conduct cooperation, crossover and mutation on groups or

individuals.

4.4 Discussion

4.4.1 Issues Revisited

In Chapter 3 we pointed out three important issues that must be addressed if we wish to apply evolutionary computation to search for multiple coadapted subcomponents in a solution. These issues include problem decomposition, evolution of collaborations and diversity preservation. Here we need to return to these issues to find out how they are addressed by our multilevel selection model.

Problem Decomposition Our model addresses problem decomposition as an emergent property. Individuals involved in cooperation often assume different responsibilities or roles. Like other team-based CEAs, our model does not require to specify the roles individuals played *a priori*. The between-group selection pressure forces individuals to develop different roles (i.e., explore different areas in the search space) in order to optimize group fitness, because individuals with duplicated roles will result in a less cooperative group. Since the survival of individuals is strongly associated with the performance of their groups, individuals within less cooperative groups are unlikely to be awarded the opportunities of reproduction.

In addition, our model decides the number of subcomponents in a solution through evolution rather than prediction beforehand. Once the most basic elements[2] that con-

[2]Elements as such are application-specific; for example, they can be a classifier in classification systems, or a movement descriptor for a robot in a robot team.

stitute solutions are determined, our model repeatedly applies the three evolutionary operators (that is cooperation, crossover, and mutation) to create groups of various sizes. At the same time, the between-level selection pressure controls the size of groups from bloating, and forces groups to reach an optimal size at appropriate granularities. Furthermore, the mapping rules introduced by the cooperation operator allow very sophisticated composition relationships between entities; as a result, entities can be genotypically or phenotypically different from the entities who compose them.

Those features are obvious improvements of CEAs. Please recall that CCEAs and SBB are the only CEAs in the literature that consider problem decomposition. However, CCEAs depend on an accurate definition of evolutionary stagnation in order to dynamically adjust the number of species (corresponding to sub-problems). Normally, evolution stagnates when the fitness of the best collaboration does not make a specified improvement over a certain number of generations. How to measure the degree of improvements and to define the length of stagnation is not trivial. In addition, adding a new species into the population will discard previous computational efforts, because evolution has to start over again. SBB defines no selection pressure to ensure optimal group size, and no mapping rules for composing groups.

Evolution of Collaborations Our model falls into the category of team-based CEAs, as it explicitly defines "groups" (i.e. teams) to represent the collaborations of individuals. To avoid greed (i.e. always select the best individuals, such as in IE) or mediocrity (i.e. always select individuals randomly such as in SBB) when forming groups, our model selects entities proportional to their fitness. Without exception, the performance of every group is measured according to a group fitness function.

According to [92, 81], the absence of this function fails to model interactions between coadapted group members, and hence will fail to build groups composed of loosely coupled individuals working towards a common goal.

The novelty of our model is to treat both individuals and groups as evolvable objects. We noticed from the discussion in Sect. 3.2.3 that it is common for EAs to optimize individual performance through evolution. As an extension, our model defines a similar evolutionary process on groups; through repeated application of selection and the three evolutionary operators on groups, better groups are selected preferentially to produce offspring, which gives potentially useful building blocks (i.e. combinations of individuals) an opportunity to be exploited and reused; as a result, the performance of groups will be gradually optimized. In the end, the search for the best performing group should be accelerated, and the accuracy provided by the best performing group should be improved as well (please refer to Sect. 4.2 for more details).

Therefore, when compared to other CEAs, our approach is the only model that evolves both individuals and their collaborations.

Diversity Preservation We have said in Sect. 3.2.1 that niching and credit assignment are different techniques used to maintain population diversity. However, according to Potter and de Jong [81], both issues should be addressed by CEAs. In our model, niching is required on both individual and group levels, ensuring less overlap in functionality between entities in the population.

Credit assignment normally adjusts individual fitness based on their contribution in the cooperation. The reason is that individuals are very likely to have unequal

fitness due to different roles they play, and the likelihood that individuals will reproduce is solely determined by individual fitness. Through credit assignment, the contribution of each individual is reflected on individual fitness. Please note that in our model credit assignment is conducted implicitly; that is to say, there is no extra step to change individuals' fitness based on their contribution. The contributions of individuals are now reflected on their group fitness, as individuals who are dedicated in their roles and have unique contributions will boost group fitness, which in turn will give group members better opportunities to reproduce, despite their fitness value.

4.4.2 Potential Application Domains

Enough has been said for the moment about the model itself, now we will focus on some examples to illustrate the applicability of this model by customizing it on different problem domains. The first two examples will demonstrate how our model can facilitate the study of artificial life. The other two examples show how to apply the model to solve real-world problems, such as classification and multi-agent systems.

The Evolution of Cooperation Cooperation has always been a thorny issue for evolutionary theorists. Our multilevel selection model, like Wilson's and Traulsen's models (please refer to Chapter 2 for details), is another alternative to explain how cooperation can emerge and persist through evolution. Individuals will be players in the Prisoner's Dilemma game. Groups are structured by the cooperation operator in a way that cooperative individuals are able to interact more frequently with each other. At the same time, between-group selection helps cooperative individuals to propagate, even though within-group selection still favors selfish individuals. In the

case of the evolution of cooperation, groups are simply a collection of individuals, and have the same cooperative traits as individuals. In fact, our model is able to support the study of evolutionary transitions. With the help of carefully crafted reaction rules and/or group fitness definitions which are not proportional to individual fitness, our model can simulate user-specified evolutionary transition; that is to say, groups will exhibit different traits from individuals. This will be a useful means to gain insight into the process of evolutionary transitions.

Artificial Chemistry Artificial chemistry is a subfield of artificial life with the quest for understanding the origin and evolution of life starting from non-living molecules [20]. This extreme bottom-up approach requires the presence of a set of interaction rules and a set of molecules, so that complex systems can be built through the process of repeatedly applying interaction rules on corresponding molecules until the requirements are met. This abstract model is analogous to our hierarchical model; individuals and reaction rules in our model are equivalent to molecules and interaction rules in Artificial Chemistry, respectively. In addition, under the provision of reaction rules, interactions happening among individuals will cause the emergence of a complex systems, whose output is more than the sum of its constituents. This is exactly the desired property required by Artificial Chemistry. Therefore, our algorithm can be easily converted to an algorithm used in Artificial Chemistry.

Classification For most classification problems, due to a large volume of data sets and complex relationships between data attributes and output class labels, it is impossible to use only one classification rule or equation to classify all data instances accurately. Normally a set of classifiers is required. Individuals in our model will

represent classifiers, while groups are collections of individuals working cooperatively to classify the whole data set. The interaction between individuals should balance generalization and specialization of individuals, so that their groups have maximum coverage and accuracy but minimum misclassification errors and coverage overlaps between individuals. The between-level selection helps to keep groups compact by eliminating redundant individuals.

Multiagent Systems Multiagent systems are a subfield of distributed artificial intelligence. It aims at providing principles for construction of complex systems that involve multiple agents and mechanisms for coordination of independent agents' behavior [94]. Our model can be applied to evolve cooperative behavior of multiple agents which accomplish common goals together, such as executing search and rescue tasks together. For this application, individuals represent a set of movement instructions of a particular agent. Groups are formed on a higher level to control and coordinate the behaviors of multiple agents. The role each agent plays is not specified beforehand, rather it should emerge as a result of evolutionary pressure putting on group levels.

4.5 Chapter Summary

Group selection theory, or what is now being termed multilevel selection theory, has been widely accepted as an explanation for the evolution of cooperation observed in nature. Motivated thus, this chapter was dedicated to the incorporation of multilevel selection into evolutionary computation in order to transcend the limitations of existing CEAs. A new computational multilevel selection model was proposed, and

a hierarchical evolutionary algorithm implementing this model was sketched. A few simple examples were illustrated just to give readers some flavor of how this model can be adapted to fit various problem domains.

The advantages gained by this model include:

- Problem decomposition is automatically achieved by the bottom-up process described in this model. With the help of the cooperation operator and reaction rules, complex systems can emerge or transit from simply constituents.
- Cooperations among independent individuals are enhanced by group selection
- The evolution of groups optimizes group performance, which in turn should also optimize individual performance
- The between-level selection helps to decide the most appropriate level of hierarchies and the size of a collaboration without human interference
- There is no need to explicitly define credit assignment, as group selection strongly associates the survival of individuals to their groups'.

Because of those advantages, we expect the algorithm to evolve faster and find more accurate solutions. We also expect the structure of a solution and the roles played by subcomponents to emerge as a result of evolution, rather than being designed manually.

Chapter 5

Experiments on The N-player Prisoner's Dilemma Game

The N-player Prisoner's Dilemma (NPD) game [2] has been widely used to study the evolution of cooperation in social, economic and biological systems. It, as discussed in Chapter 2, has helped us to understand how cooperation arises and evolves in Wilson's and Traulsen's group selection models. One purpose of this chapter is to use the NPD game again to experimentally verify the feasibility of our proposed multilevel selection model in achieving the evolution of cooperation, before applying the model to complex computational tasks. Consequently, we will show in Sect. 5.1 how to adapt our model to fit the investigation of NPD games. Section 5.2 will first study the performance of our model in promoting cooperation under different parameter settings. We shall continue by investigating the contributions of the group selection and the cooperation operator to the evolution of cooperation. Finally our model and the improved Traulsen model (i.e. T2 discussed in Sect. 2.3) are compared in terms of robustness and sensitivity to parameter changes.

Most multilevel selection models in the literature focus on addressing the evolution of cooperation. There is, however, another aspect of multilevel selection theory — namely, it might be able to provide explanations for evolutionary transitions, which involve the creation of higher level complexes out of simpler elements. In order to be identifiable, these new complexes should exhibit heritable traits different from those of simpler elements. The other purpose of this chapter, therefore, is to explore whether our multilevel selection model can support evolutionary transitions. To this end, we investigate its ability to exploit the division of labor, as a crucial step in many of the major transitions [60] is the division of labor between components of an emerging higher level unit of evolution [31]. Examples include the separation of germ and soma cells in simple multicellular organisms, the appearance of multiple cell types and organs in more complex organisms, and the emergence of castes in eusocial insects [31]. In Sect. 5.3 we will first show how to adapt our model for the study of the division of labor. Then we shall investigate how our model helps independent individuals to transition to groups with totally different functionalities; in terms of division of labor, those are groups with members executing various skills with possibly different rewards.

5.1 Multilevel Selection Model on the NPD Game

5.1.1 Related Work

The NPD game is a simple, yet extensively used model to study the evolution of cooperation. In this game, players independently choose cooperative or defective actions, without knowing the other players' choices. Cooperators pay a cost, c, for

other players to receive a benefit, b. Defectors pay no cost and distribute no benefits. Costs and benefits are measured in terms of fitness. Obviously, in any NPD games with a well-mixed population, defectors obtain a higher fitness than cooperators. Without external help, natural selection will eventually drive cooperators to extinction. For a detailed description of the NPD game, see Sect. 2.3.1.

Group selection, which spatially structures the population into groups with various assortments of cooperators and defectors, can lead to the evolution and stability of cooperative traits in the NPD game [48, 82, 83, 120]. The experiments in Sect. 2.3 concluded that the success of group selection depends on effectively maintaining the variance in group composition. The higher the variance, the bigger the fitness difference between groups, therefore, the easier selection among groups can be conducted, and the less a group selection model is bound by parameters, such as group size, selection strengths, or the percentage of cooperators in a population [120].

Currently, investigations of most group selection models, even those conducted under the heading of multilevel selection, focus on selection acting on two levels, namely the group level and the individual level, assuming that two levels can be easily extended to multiple levels. However, multilevel selection is more complicated than selection on two levels; it has to consider not only how to produce group variance, but also how to define groups on each level, how to decide which level to select on, how to perform evolution on each level, and how to bring these levels together. This may explain why there are few computational multilevel selection models in the literature.

One example of a computational multilevel selection model was proposed by Chu and Barnes [10]. Their study concentrated on a model with three nested levels, as shown in Fig. 5.1. This model demonstrates the simplest case of multilevel selection.

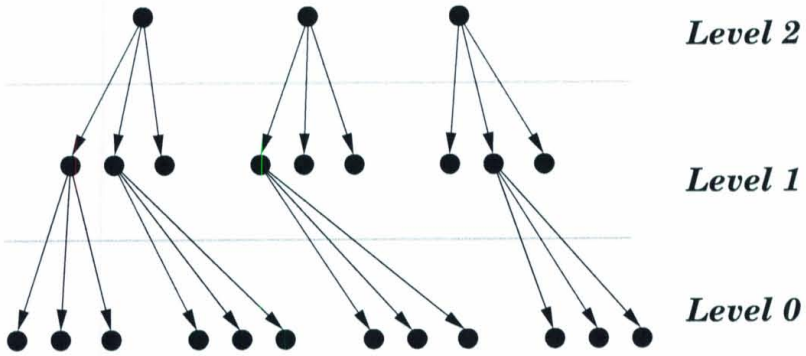


Figure 5.1: Schematic outline proposed by Chu and Barners [10] showing how to organize agents into levels. Not all agents are drawn at the lowest level.

Agents, denoted by black dots, are evolvable entities with both a genotype and a phenotype. The genome of an agent consists of n genes (n is limited to odd numbers), and each gene has a value of 1 or -1. The phenotype of an agent is either -1 when being a defector or 1 when being a cooperator, which is determined by the majority value of its genes; for example, if an agent has more genes with a value of -1, its phenotype will be -1. The genes of an agent on level $m+1$ are determined by the phenotype of n agents on level m ($m \geq 0$).

Except for the lowest level, at every level agents are subject to evolution. The level at which selection takes place is determined by the *level selection parameter* (*lsp*). If *lsp* is 0, selection always takes place at level 2; if it is 1, selection always happens at level 1; for any value between 0 and 1, level 2 is selected with a probability proportional to the value of *lsp*.

On each selected level, $10 \times n$ tournaments are held to evaluate the fitness of entities on that level. For any level between the highest and the lowest, tournaments are staged among n agents which compose the genome of a randomly selected agent

from the level immediately above. In each tournament, two agents out of n are picked to play against each other, and receive a reward (also known as the fitness value) according to the prisoner's dilemma pay-off matrix PD.

$$PD = \begin{pmatrix} CC & CD \\ DC & DD \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 3 & 1 \end{pmatrix} \tag{5.1.1}$$

If the highest level (which is level 2 in this case) is selected, all agents on the top level are involved in tournaments. To examine how fitness definitions on higher levels affect the model, three fitness functions were tested for agents on level 2. The first fitness function always assign a one-point reward to cooperators, no matter whom they confront in the tournament, the second fitness function considers the number of cooperators in the genome of an agent, and the last fitness function uses the expected pay-off of an agent as its fitness. Agents accrue fitness values from each tournament.

At the end of a tournament round, 90% of the agents who have the highest accrued fitness will reproduce. To guarantee a correct genotype and phenotype mapping between levels, the offspring will copy the entire hierarchy of the parent, and randomly replace an existing agent and its hierarchy. Mutations are inflicted randomly to agents on level 1 at a user-defined rate.

The authors made two valuable observations from their experiments: 1) the behavior of the model strongly depends on how fitness at higher levels is defined; 2) the selection on higher levels should occur at a higher frequency relative to lower level selection events in order to encourage cooperation. Surprisingly, they rejected the idea of multilevel selection, for the following two reasons. Firstly, in all experimental simulations, in which different mutation rate, level selection frequency and fitness definitions on level 2 were tested, their model can barely make cooperators gain dom-

inance in a population. Secondly, according to the authors, high selection frequency on higher levels means even higher replacement frequency for agents at lower levels, because any replacement of a single agent on level 2 immediately removes n agents at level 1, and n^2 agents on level 0, and such a replacement frequency is unrealistic in real biological systems.

We, however, argue that the above conclusion is drawn from a misunderstanding of multilevel selection and incorrect assumptions. First of all, group fitness is not correctly defined for agents on level 1. As stated by the authors, group selection dominates on level 1 when *lsp* is set to 1 (see page 4 in [10]). This implies that agents on this level are regarded as groups for agents on level 0. Unfortunately, fitness calculated by the pay-off matrix (see Eq. 5.1.1) does not measure their group performance, but only individual performance. Agents with phenotype of -1 (i.e., defectors) have higher fitness than agents with phenotype of 1 (i.e., cooperators). This is the reason why the population converges to defectors when selection happens frequently on level 1.

Second, multilevel selection is not about simply switching selection from one level to another. The idea of group selection has to be applied on each level, as group selection promotes fierce competition between groups, and hence cooperation within groups. The better a group performs, the greater the chances of this group and its members surviving and prospering in evolution. In other words, the reproduction probability of an agent depends on the fitness of its group, whereas such a selection force is missing in the Chu and Barnes' model. Although agents on level 1 and 2 do reproduce at a rate associated with their fitness, the process of accruing fitness is totally random, or is related to the frequency of an individual being randomly

selected. Thus, a less cooperative agent may end up with a higher fitness value, and hence produce more offspring. This contradicts the notion of group selection, and also explains why cooperators cannot dominate the population no matter on which level selection takes place.

Last but not least, their second reason to reject multilevel selection is based on a rigid and biologically unrealistic model. According to the design of this model, n agents exclusively compose an agent at the level immediately above, so the entire hierarchy of a selected agent has to be replaced in order to guarantee a consistent genotype and phenotype mapping between levels. It is this constraint that causes the high replacement frequency to occur at lower levels. In fact, such a composition requirement is not biologically sound; for example, different cells may share the same genes, but the death of a cell does not remove those genes from other cells. Such a replacement mechanism also evens out the variance between groups, which makes selection between groups harder.

In conclusion, Chu and Barnes' work can be considered a good initiative aimed at investigating the practicality of multilevel selection, but serious flaws exist in their model. Their work reveals that the notion of multilevel selection does not seem to be inherently difficult, but the actual implementation can lead to a number of complications; for example, the organization of the hierarchical structure (i.e. the connection between any two adjacent levels), and the definition of evolution. In the end, we are still facing the question of whether to embrace or reject the idea of multilevel selection, when it is applied correctly as a mechanism to promote cooperation through evolution.

5.1.2 Algorithm Customization

In this section, we attempt to approach the above question by examining our multilevel selection model when applied to the NPD game. Recall that our model involves many elements, including a cooperation operator, group selection, evolution on the individual level, evolution on group levels, and diversity maintenance. To focus on the effect of multilevel selection, we customize the model to only highlight the cooperation operator and group selection. Those two elements are responsible for organizing the population into a hierarchical structure required by multilevel selection and defining the selection on a structured population, respectively. They may create opportunities for cooperators to interact more frequently with each other, in order to obtain high fitness to survive the selection. We also simplify the evolution of individuals to asexual reproduction without mutation. Crossover and mutation on group levels, as well as diversity maintenance are temporarily not considered.

The execution of the cooperation operator and group selection requires both group fitness and individual fitness. However, individuals which participate in the NPD game cannot obtain their fitness unless they interact with others. To this end, we changed our model (see Fig. 4.1) somewhat to satisfy this requirement. As shown in Fig. 5.2, randomly initialized individuals (represented by the white dots inside the individual pool) are exclusively paired into groups on level 0. Those groups are the smallest unit in which the individual fitness can be evaluated. From level 0, the cooperation operator starts to build a hierarchical structure level by level.

The evolutionary algorithm shown in Algorithm 4 implements this customized model. It begins with initialization. N individuals, r percent of which are cooperators, are randomly created and exclusively paired into groups at level 0. The genome of

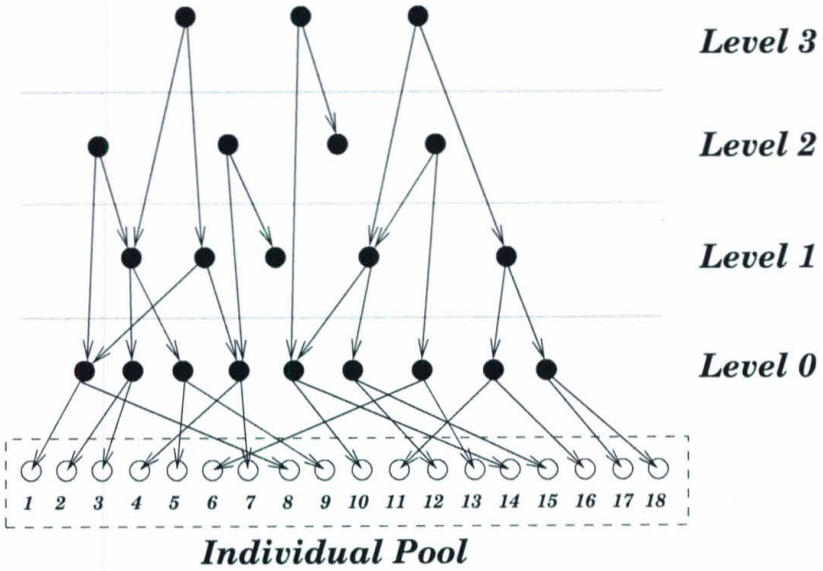


Figure 5.2: Our customized multilevel selection model for the NPD game.

individuals only contains one gene. This gene has two variants (alleles); one allele codes for cooperators, the other allele for defectors. When the former is expressed, the individual is said to be a cooperator; otherwise, it is a defector.

Groups at level 0 need to have their fitness evaluated right away. This group fitness can be easily calculated by averaging the individual fitness of its members. Group members, or individuals, possess a fitness determined by the following equations, depending on whether it is a cooperator (C) or a defector (D):

$$f_{C_i}(x) = base + w(\frac{b(n_i q_i - 1)}{n_i - 1} - c), \quad (0 \leq i < m) \tag{5.1.2a}$$

$$f_{D_i}(x) = base + w\frac{bn_i q_i}{n_i - 1}, \quad (0 \leq i < m) \tag{5.1.2b}$$

where m is the number of groups in the population, *base* the base fitness of cooperators and defectors, q_i the fraction of cooperators in group i, n_i the size of group i, b and c are the benefit and cost caused by the altruistic act, respectively, w is a coefficient. These

Algorithm 4: An EA based on our multilevel selection model.

```
1  P ← Initialize_Population(N, r);
2  Evaluate_Individual_Fitness (P);
3  Evaluate_Group_Fitness(P);
4  while population does not converge or max generation is not reached do
5      gp ← Conduct_Cooperation(P);
6      Evaluate_Individual_Fitness (gp);
7      Evaluate_Group_Fitness(gp);
8      Add_a_Group_to_Population(gp, P);
9      if Population_Size(P) > N' then
10         Remove_a_Group();
11     end
12     for i ← 0 to n do
13         idv ← Reproduce_an_Individual(P);
14         Replace_an_Individual(idv, P);
15         Update_Changes(idv, P);
16     end
17 end
```

are the same fitness functions used in the investigation of Wilson's and Trauslen's models. This fitness definition also implies that cooperation is not supported at the individual level, as cooperators always have lower fitness than defectors. However, groups with more cooperators will achieve higher group fitness.

In each generation, only one group is created by the cooperation operator. The cooperation operator, as we explained in Sect. 4.3, selects two groups proportional to their fitness, which automatically decides the levels to select on. This increases the complexity of groups as well as might cause new levels to appear in the hierarchical structure. To prevent hierarchical depth from ceaselessly growing, we assign every individual a unique number as its ID; no individuals with the same ID can exist within the same group. After fitness evaluation, a new group is added to the population P. If at this point the maximal number of groups, say N', is reached, another group has to be removed from the population inversely proportional to fitness.

We also reproduce n individuals asexually every generation. A group is first selected from groups on all levels, from which an individual is selected as parent. Both selections are proportional to fitness. Individuals in cooperative groups will have a higher probability to be selected and reproduced. Since cooperators within such groups are in the majority, they have a better chance to be selected by within-group selection, even though they have lower fitness than defectors in the same group. Each parent's genome further replaces the genome of a less fit individual in the individual pool (that is, the less fit individual still stays in the population, but with its genome changed.). Individuals in this pool are allowed to participate in composing more than one group. That is to say, each of those individuals may have multiple copies, but in different groups. Depending on group composition, it will have different fitness, and

the simplest way to determine its fitness is to average the fitness of all its copies (i.e., individuals with same ID).

After the replacement of an individual in the individual pool, this change needs to be communicated. We can update the genome of either all the copies of the replaced individual, or the only copy on level 0. We choose the former in order to ensure the same genotype appears in individuals with the same ID. The group fitness and individual fitness of affected groups need to be updated, accordingly. We repeat the process until a termination condition has been reached or the individual pool converges to either cooperators or defectors.

5.1.3 Discussion

When compared to Chu and Barnes' multilevel selection model, our model works differently in regard to the construction of the hierarchical structure. Chu and Barnes' model clearly defines the genotype and phenotype for agents, and also defines the mapping from genotype to phenotype for agents at any level, as well as the mapping from phenotype to genotype for agents between two adjacent levels. The latter specifies the bond between two levels, through which a hierarchical structure is built. This mapping, in fact, changes the nature of agents during the transition; an agent becomes a specific gene in the genome of an agent at the next higher level. This causes the unrealistic replacement frequency, and partly leads the authors to reject multilevel selection.

In contrast, our model only defines the genotype of individuals. The genotype of groups, though not explicitly defined, can be regarded as a collection of its members' genomes: the genome of a group is composed by the genomes of two entities

(individuals or groups) from lower levels. In this way, the between-level connection is established. Any entity in our model can be regarded as a potential building block, as long as it shows benefits to fitness. Once it is selected by the cooperation operation, it will contribute a copy of its genome to the new group. Since there is no phenotype-genotype mapping restrictions between groups and their constituents, when an entity is replaced, there is no need to replace its constituents. The replacement of a group in our model means the current combination of its members loses its competition advantage in evolution, which does not necessarily indicate that the genomes it got from lower level entities are useless. Indeed, some of them may still be good building blocks if in another composition or environment. Yet our connection shows the simplest way to bring levels together, which also seems more biologically reasonable to us. More complicated between-level mappings are left for future studies.

In respect to the question of when to start or stop building new levels, no specific mechanism was mentioned in Chu and Barnes' model, whereas the cooperation operator was introduced into our model for this purpose. Combining two existing groups will result in a new group with increased complexity, even creating a new level if such complexity has not been reached before. This new level can actually be a part of the system only when the fitness of its groups is improved. In other words, this operator is driven by fitness, which determines when to start or stop building a new level.

This operator also distinguishes our model from Wilson's and Traulsen's group selection models in terms of changing population dynamics. It is worth mentioning that our "cooperation" operator is the inverse of the "split" operator in Traulsen's model (see Algorithm 2 on page 32). As we concluded in Sect. 2.3, Traulsen's model is a better group selection model than Wilson's, because it is able to create high

variance between groups. This raises the following two questions: Does the "split" operator contribute to such a good result? Might the "cooperation" operator in our model serve the same purpose?

To answer these questions, we can analyze the two operators in the NPD game. Suppose we have three groups: g_1 with x_1 cooperators and y_1 defectors, g_2 with x_2 cooperators and y_2 defectors, and g_3 with $x_1 + x_2$ cooperators and $y_1 + y_2$ defectors. Group g_3 can be regarded as the group either after the cooperation of g_1 and g_2 or before the split into g_1 and g_2.

We know for any given group g with x cooperators and y defectors, the fitness for a cooperator is defined as

$$f_C(I) = \beta + \frac{b(x-1)}{x+y-1} - c, \quad (0 \leq i < x), \tag{5.1.3}$$

and the fitness of a defector is

$$f_D(I) = \beta + \frac{bx}{x+y-1}, \quad (0 \leq j < y), \tag{5.1.4}$$

So the fitness of group g can be calculated as,

$$\begin{aligned} f(g) &= \frac{x f_C(I) + y f_D(I)}{x+y} \\ &= \frac{x(\beta + \frac{b(x-1)}{x+y-1} - c) + y(\beta + \frac{bx}{x+y-1})}{x+y} \\ &= \frac{\beta(x+y) + x(b-c)}{x+y} \\ &= \beta + \frac{x}{x+y}(b-c) \end{aligned} \tag{5.1.5}$$

Therefore, the fitness of the group g_1, g_2, and g_3 are

$$f(g_1) = \beta + \frac{x_1}{x_1 + y_1}(b - c) \tag{5.1.6}$$

$$f(g_2) = \beta + \frac{x_2}{x_2 + y_2}(b - c) \tag{5.1.7}$$

$$f(g_3) = \beta + \frac{x_1 + x_2}{x_1 + x_2 + y_1 + y_2}(b - c) \tag{5.1.8}$$

The fitness difference between group g_1 and g_3 is

$$\begin{aligned} f(g_1) - f(g_3) &= \beta + \frac{x_1}{x_1 + y_1}(b - c) - \beta + \frac{x_1 + x_2}{x_1 + x_2 + y_1 + y_2}(b - c) \\ &= (b - c)(\frac{x_1}{x_1 + y_1} - \frac{x_1 + x_2}{x_1 + y_1 + x_2 + y_2}) \\ &= (b - c)\frac{x_1 y_2 - x_2 y_1}{(x_1 + y_1)(x_1 + y_1 + x_2 + y_2)} \end{aligned} \tag{5.1.9}$$

Similarly, the fitness difference between group g_3 and g_2 is

$$\begin{aligned} f(g_3) - f(g_2) &= \beta + \frac{x_1 + x_2}{x_1 + x_2 + y_1 + y_2}(b - c) - \beta + \frac{x_2}{x_2 + y_2}(b - c) \\ &= (b - c)(\frac{x_1 + x_2}{x_1 + y_1 + x_2 + y_2} - \frac{x_2}{x_2 + y_2}) \\ &= (b - c)\frac{x_1 y_2 - x_2 y_1}{(x_2 + y_2)(x_1 + y_1 + x_2 + y_2)} \end{aligned} \tag{5.1.10}$$

Since x_1, x_2, y_1, and y_2 are greater than 0, the order of sequence $f(g_1)$, $f(g_2)$, and $f(g_3)$ then depends on the relations between b and c and between $x_1 y_2$ and $x_2 y_1$. Therefore, All possibilities except $b - c = 0$ or $x_1 y_2 - x_2 y_1 = 0$ are shown in Table 5.1. When $b - c = 0$ or $x_1 y_2 - x_2 y_1 = 0$, the fitness differences between $f(g_1)$, $f(g_2)$, and $f(g_3)$ are zero.

Clearly under all circumstances, the fitness of group g_3 is in between the fitness of g_1 and g_2. That is to say, if the "split" operator is applied, one of the split group will have higher fitness than the original group; and if the "cooperation" operator is applied, the composed group will have higher fitness than one of the original groups.

Table 5.1: The ordered value-sequence of $f(g_1)$, $f(g_2)$, and $f(g_3)$

$b-c>0$	$x_1y_2 > x_2y_1$	$f(g_1) > f(g_3) > f(g_2)$
	$x_1y_2 < x_2y_1$	$f(g_2) > f(g_3) > f(g_1)$
$b-c<0$	$x_1y_2 > x_2y_1$	$f(g_2) > f(g_3) > f(g_1)$
	$x_1y_2 < x_2y_1$	$f(g_1) > f(g_3) > f(g_2)$

In either case, the overall group fitness is improved, as is the average percentage of cooperators in groups. On the other hand, the unequal fitness of involved groups implies that both operators introduce new groups with different compositions into the population, so that a good level of between-group variance can be sustained. Therefore, we can conclude that both operators are useful in promoting cooperation. However, the "split" operator should be more effective than the "cooperation" operator, as it yields a greater difference of fitness and of between-group variance among the groups involved.

5.2 Experiment 1: The Evolution of Cooperation

The first set of experiments investigates the performance of our model on the evolution of cooperation. To be specific, we are interested in whether the model can successfully remove all defectors from a population and how it performs under different parameter settings.

5.2.1 Experimental Setup

The investigations to be conducted here are similar to the ones conducted on Wilson's and Trauslen's models (refer to Sect. 2.4 for details), but here only focus on two parameters: the coefficient w and the initial fraction of cooperators r. Group size is no longer considered, as it becomes a self-adaptive parameter. We ran the algorithm 20 times, each with a generation size of 5000, a population of 200 individuals and maximum number of 20 groups at level 1 and above . Eq. 5.1.2a and Eq. 5.1.2b are used to calculate the fitness of cooperators and defectors within a group, respectively. Base fitness *base* is set to 10, benefit b to 5, and cost c to 1 in these two equations. The fitness of a group is defined as the average individual fitness of its members.

We measure the performance of the algorithm by the probability of fixation to cooperators $P_{fixation}$ and the average fixation speed $S_{fixation}$. In population genetics, fixation refers to the change in a gene pool from a situation where there exist at least two variants of a particular gene (allele) to a situation where only one of the alleles remains [109]. $P_{fixation}$ is computed as the number of runs whose populations converge to cooperators over 20 runs. $S_{fixation}$ is the average number of generations in 20 runs required to obtain a population with only one gene variant present in the population; or, to put it another way, when group fitness stops to change. As in the previous study, we also collect average variance ratio $Avg_{variance}$ (refer to Eq. 2.3.4) in each run. Please recall that the higher this ratio, the more prominent the effect of group selection.

5.2.2 Experimental Results

5.2.2.1 Investigation of different parameters settings

We first test the performance of the model when different numbers of cooperators are used to initialize the population. So we vary the value of r from $\{0.1, 0.3, 0.5\}$, and set $w = 1$. The results obtained over 20 runs are listed in Table 5.2.

Table 5.2: The effects of initial fraction of cooperators r

Settings	$P_{fixation}$	$S_{fixation}$	$Avg_{variance}$
$w = 1, r = 0.1$	1	390.9	0.419
$w = 1, r = 0.3$	1	341.6	0.430
$w = 1, r = 0.5$	1	312.6	0.440

We then test how selection pressure affects the model by changing the value of coefficient w from $\{0.1, 0.5, 1, 2, 5, 10\}$. The initial fraction of cooperators r is set to 0.5 at this time. The results are shown in Table 5.3.

From Table 5.2 and Table 5.3, we can see that the population converges to cooperators under all settings. The changes of r and w do not affect the performance very much. However, when less cooperators are present in the initial population, or when the selection pressure is either too weak or too strong, the population takes longer to converge. Please recall that strong selection means that the payoff is large compared with the baseline fitness; weak selection means the payoff is small compared with the baseline fitness [72]. When selection is too weak, the relative fitness difference between cooperators and defectors (i.e. the difference between their payoff) is very

small. For example, given a group with 9 cooperators and 1 defectors, when $w = 0.1$, the relative fitness difference between defectors and cooperators is only 0.156. As a result, the relative fitness between different groups should be small, too. Therefore, the effect of group selection is less obvious. When selection is too strong, though cooperative groups are favored by between-group selection, defectors have much higher fitness than cooperators. Consider the previous example: when $w = 10$, the relative fitness between defectors and cooperators is 15.6. Between-group selection cannot easily prevail over within-group selection. Therefore, it takes also longer under these circumstances to remove defectors from the population.

Table 5.3: The effects of selection pressure w

Settings	$P_{fixation}$	$S_{fixation}$	$Avg_{variance}$
$w = 0.1, r = 0.5$	1	441.05	0.418
$w = 0.5, r = 0.5$	1	292.45	0.431
$w = 1, r = 0.5$	1	312.6	0.440
$w = 2, r = 0.5$	1	370.75	0.455
$w = 5, r = 0.5$	1	644.55	0.456
$w = 10, r = 0.5$	1	1174.05	0.454

5.2.2.2 Functionalities of group selection and the cooperation operator

Group selection and the cooperation operator are two new concepts introduced in our algorithm. In order to get a clear picture of their contributions to the evolution of cooperation, we conducted another two experiments. In the first experiment, we

replaced group selection (line 11 to 12 in Algorithm 4) with individual selection. Individual selection selects individuals for reproduction based on their fitness values; that is the higher the fitness value, the higher is its probability of reproduction. Individual selection in this experiment is conducted directly on the individual pool.

We applied the model without group selection on all settings again, and the results are shown in Table 5.4. This time a $P_{fixation}$ value of 0 resulted on almost all settings.

Table 5.4: The performance of the model without group selection

Settings	$P_{fixation}$	$S_{fixation}$	$Avg_{variance}$
$w = 1, r = 0.1$	0	211.10	0.361
$w = 1, r = 0.3$	0	430.65	0.372
$w = 1, r = 0.5$	0	747.95	0.387
$w = 0.1, r = 0.5$	0.3	2137.67	0.401
$w = 0.5, r = 0.5$	0.05	1255.00	0.390
$w = 1, r = 0.5$	0	747.95	0.387
$w = 2, r = 0.5$	0	561.05	0.391
$w = 5, r = 0.5$	0	369.40	0.392
$w = 10, r = 0.5$	0	317.95	0.399

When $P_{fixation}$ was 0, $S_{fixation}$ referred to the fixation speed to defectors. Therefore, we can safely say that without group selection, our model can barely maintain cooperators in the population. Defectors now reproduce more often and gradually take over a population, because they have higher fitness than cooperators. We also noticed that

the population did converge to cooperators occasionally when selection pressure was very weak ($w = 0.1$ or $w = 0.5$). As we explained before, under weak selection the fitness difference between cooperators and defectors is very small. This, of course, gives cooperators an opportunity to be selected and reproduced.

In the second experiment, we removed the cooperation operator (line 3 in Algorithm 4) from the model, and ran the algorithm once again. As shown in Table 5.5, our model without cooperation was able to achieve the evolution of cooperation except

Table 5.5: The performance of the model without cooperation

Settings	$P_{fixation}$	$S_{fixation}$	$Avg_{variance}$
$w = 1, r = 0.1$	1	566.35	0.464
$w = 1, r = 0.3$	1	450.05	0.467
$w = 1, r = 0.5$	1	357.85	0.476
$w = 0.1, r = 0.5$	1	516.7	0.492
$w = 0.5, r = 0.5$	1	311.8	0.484
$w = 1, r = 0.5$	1	357.85	0.476
$w = 2, r = 0.5$	1	538.05	0.474
$w = 5, r = 0.5$	0.95	2458.47	0.480
$w = 10, r = 0.5$	0	N/A	0.500

when selection pressure was too strong. If $w = 10$, the population converged neither to cooperators nor to defectors within 5000 generations. If we compare the fixation speed obtained in this experiment with the ones shown in Table 5.2 and Table 5.3, we

will notice immediately that, given the same parameter settings, the population in a model without cooperation took longer to converge. For the reason we refer to the discussion on Page 107: The cooperation operator produces a composed group with a higher fitness value than one of the original groups. Hence, it changes group composition and increases the chances for putting more cooperators together, which in turn makes the role of group selection more prominent. Without such a push, evolution definitely is slowed down, especially when selection pressure is stronger. That is the reason why a population cannot converge if w is set to 10. Therefore, the cooperation operator in our model helps to encourage cooperation and accelerates evolution.

5.2.2.3 Performance comparison to the improved Traulsen model

So far, we have investigated our model on different parameter settings and further analyzed the functionality of group selection and the **cooperation** operator. However, we are not yet clear about the performance difference between our model (denoted as $W\&B$) and the improved Traulsen model ($T2$)[1]. A comparison between them is highly useful.

In $T2$, groups have a pre-defined size, which determines the maximum number of individuals a group can have. When this constraint is violated, the "split" operator is triggered with a certain probability. As we summarized in Sect. 2.3.5, the smaller the group size, the greater the group variance, and hence the easier it is to evolve cooperative groups. Hence, to challenge our model, we compare it to $T2$ with group size setting of 5.

[1]Experiments in Sect. 2.3 confirmed that the improved Traulsen model is the best model among the three investigated models.

First we ran two models 1000 times on a population with $r = 0.005$ (i.e. 199 defectors and 1 cooperator), and $r = 0.995$ (i.e. 1 defector and 199 cooperators), respectively. The results are shown in Table 5.6. The probability of fixation to

Table 5.6: Comparison of $W\&B$ and $T2$ when $r = 0.005$ and $r = 0.995$

	r=0.005		r=0.995	
	cooperators	defectors	cooperators	defectors
$W\&B$	0.544 (398.34)	0.456 (11.18)	1 (26.05)	0 (0.0)
$T2$	0.031 (233.16)	0.969 (7.54)	0.996 (22.03)	0.004 (253.5)

cooperators and to defectors are listed outside of the parenthesis, while the average fixation speed is inside of the parenthesis. Even though only 1 cooperator exists in the population at the outset, cooperators in our model have more than 50 percent chance of taking over the population, a probability that is 17 times greater than for $T2$. When r is set to 0.995, our model never converges to defectors, whereas Traulsen's model occasionally did so even under these conditions. This result indicates that it our model is resistant to loosing cooperators from the population.

We also compared the two models under different selection pressures with w varying from $\{0.01, 0.1, 0.5, 1, 2, 5, 10, 100\}$. Table 5.7 displays $P_{fixation}$ and $S_{fixation}$ (listed in parenthesis) obtained over 20 runs. Both models achieve the highest probability of fixation to cooperators under most settings, except for $w = 0.01$. When $w = 0.01$, which implies an extremely weak selection pressure, our model converges

Table 5.7: Comparison of $W\&B$ and $T2$ under various selection pressures

	$W\&B$	$T2$		$W\&B$	$T2$
w=0.01	1 (1379.79)	0.6 (1031.75)	w=2	1 (370.75)	1 (155.35)
w=0.1	1 (441.05)	1 (458.35)	w=5	1 (644.55)	1 (137.10)
w=0.5	1 (292.45)	1 (224.05)	w=10	1 (1174.05)	1 (126.80)
w=1	1 (312.60)	1 (207.55)	w=100	1 (1578.75)	1 (109.75)

to cooperators on all runs, but at the expense of a very slow fixation speed, while $T2$'s between-group selection strength can no longer easily outweigh the within-group selection strength, so that in some runs defectors dominate the population.

From the above two experiments, we can conclude that our model is able to promote the evolution of cooperation over a wider range of parameter values, i.e. our model is less sensitive to parameter changes. The reason is related to the different ways of changing group structures. Our model starts with small groups of only 2 individuals, from which larger groups are gradually built up. In contrast, the initial group size in Traulsen's model is 5. Although the "split" operator helps to reduce group size, it occurs at a relatively low rate. In addition, inserting offspring into split groups increases group size again. As we discussed before, between-group selection has a more pronounced effect on smaller groups than on bigger groups. Small groups can thus avoid eliminating cooperators too quickly from the population, which allows group selection to have enough time to play its role. This is the case especially when the selection pressure is too weak or the initial fraction of cooperators is too low. However, we also noticed that, given the same setting, the fixation speed of our model is

slower than that of improved Traulsen model. This confirms our theoretical analysis in Sect. 5.1.3. The cooperation operator is not as effective as the split operator when it comes to increasing the fitness difference among groups.

5.3 Experiment 2: Evolutionary Transitions

In Sect. 2.2, we introduced two types of multilevel selection: MLS1 and MLS2, and their relationship with evolutionary transitions. MLS1 takes place at a early stage of evolutionary transitions to promote the emergence of cooperation, and MLS2 happens at a later stage of evolutionary transitions to develop group traits, which are normally different from individual traits. In MLS1, group fitness is defined as the average fitness of individuals within a group, while in MLS2, group fitness is defined independent of the average fitness of its individuals. As transitions proceed, group fitness gradually becomes "decoupled" from individual fitness [65]. Once group fitness is decoupled, the transition has been achieved, and new complexes have been created that assume an existence of their own.

Obviously, the multilevel selection we demonstrated in the previous experiment belongs to MLS1: Group fitness is defined as the average individual fitness (i.e. individuals and groups share the same heritable trait), and individuals are the object of evolution. In the next experiment, we will show how our model can incorporate both MLS1 and MLS2 to support evolutionary transitions. The investigation uses the division of labor as an example. Division of labor is a group trait resulting from evolutionary transitions, where low level independent entities with specialized skills cooperate to increase the reproductive success of high level complexes. First, we ex-

amine the ability of our model to evolve groups fulfilling various numbers of skills from a population of independent individuals, when all skills receive the same reward. Second, we examine the dynamics within the model and the responses of individuals when different skills are given different rewards.

5.3.1 Experimental Setup

We adopt the extended N-player Prisoners Dilemma (NPD) game to study the division of labor. The NPD game is the classical setting for addressing the evolution of cooperation. Once cooperation is reached, all players possess the same cooperative trait, which is also the only trait required for cooperation. Even if such cooperation breaks down by loosing some individuals, the rest should still be capable of cooperating with others. Evidently, the game is not useful for investigating division of labor unless some extensions are made. Therefore, we first change the NPD game by attaching a new trait called "skill" to each player; then we redefine the goal of the NPD game: find N players who not only are willing to cooperate but also possess all required skills.

We also make the following three changes to our model. We first added a new attribute, "*skill_id*", for individuals to indicate the skill they possess. Next, a new genotype is defined for groups, which is represented by a Boolean list. Each position in the list is connected to a unique skill, so that the genotype of a group can keep track of all different skills of its members. A reaction rule is defined to specify the mapping between individual genotype and group genotype. When a skill is possessed by at least one cooperator in a group, the reaction rule will set the corresponding position in the genotype to true (we say it is activated); when the skill is no longer

possessed by any cooperator in that group, the rule will inactivate the position by setting it to false. Again, compared to groups in the evolution of cooperation, groups here require their members to develop different skills, not just to cooperate. As a result, groups exhibit more traits than simply the cooperative trait of individuals. Genetically, groups in our model are ready for evolutionary transitions.

Finally, we change the fitness definition on the group levels, as shown in Eq. 5.3.1.

$$g_{new}(y) = \frac{\sum_{i=0}^{n} f_{idv}(x_i)}{n} \times \frac{active_{pheno}(y)}{length_{pheno}(y)} \tag{5.3.1}$$

This revised fitness measures the performance of a group in two respects: (i) the average individual fitness of its n members and (ii) the percentage of activated skills in the genotype. The intention behind this fitness definition is straightforward; the first part encourages the appearance of cooperators as cooperators improve the overall individual fitness, and the second part rewards groups in which cooperators possess as many different skills as possible. Obviously, this group fitness is not defined as the average individual fitness, but it can be either proportional to average individual fitness, or completely "decoupled" from individual fitness, depending on the influence of the second term of the fitness function. According to Okasha [75], the former indicates the transition from MLS1 to MLS2, and the latter indicates that groups have fully emerged as discrete units. Both encourage evolution to reach transitions.

Since the purpose of this experiment is to study the division of labor, we no longer change the values of coefficient w and the initial fraction of cooperators r, but set them to 1 and 0.5, respectively. Other parameters, such as population size, generation size, the maximum number of groups in the population, and so on, use the same settings described for Experiment 1 in Sect. 5.2.1.

5.3.2 Varying Skills

The first part of this experiment tests whether our model is able to evolve a cooperative group that has all required skills, if all skills have the same fitness benefit.

We start the experiment with 5 different skills. At initialization, individuals independently choose to be a cooperator or a defector. In addition, they randomly set their *skill_id* with one of 5 skills, {1, 2, 3, 4, 5}. An individual with an attached skill will perform a specific task. The best performing group should contain only cooperators and should have all 5 skills presented in its phenotype. We then gradually increase the number of desired skills to 10, 15 and 20. For each setting, we ran the algorithm 20 times. The results are collected in Table 5.8.

Table 5.8: The performance of our model when individuals play various skills

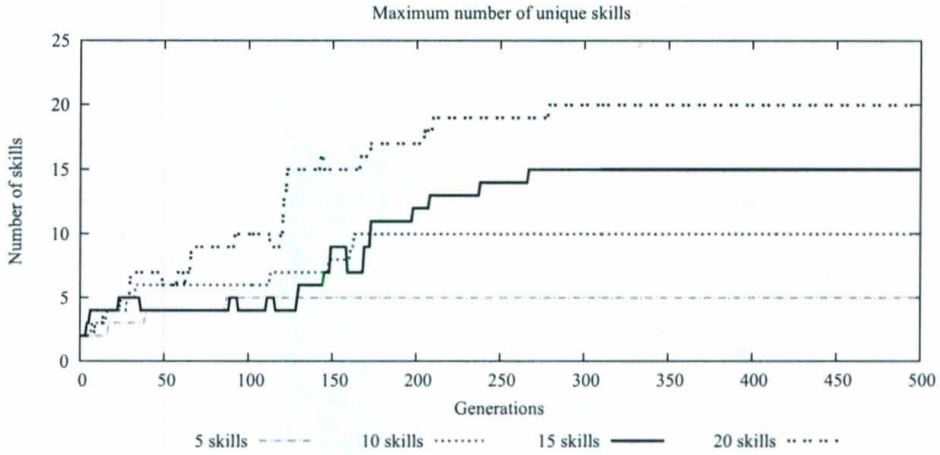
Settings	$P_{fixation}$	Activated Skills	Convergence Speed
$skills = 5$	1	5	96.3
$skills = 10$	1	10	181.55
$skills = 15$	1	15	247.60
$skills = 20$	1	20	301.25

The probability of fixation $P_{fixation}$ with a value of 1 is obtained under all settings, which indicates that defectors, despite a relatively high individual fitness, are eliminated from the population, whereas cooperators dominate the population eventually. MLS is the explanation for this result. More importantly, the best performing group for each setting develops all required skills through evolution. This demonstrates that

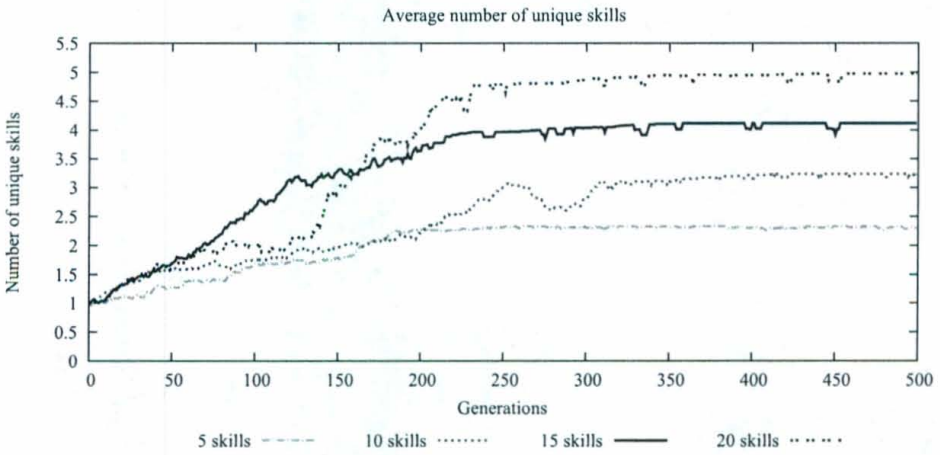
MLS2 is at work. It is not surprising to see that the larger the number of desired skills, the slower the population was to reach the maximum on group fitness. This is simply a reflection of the problem becoming harder when the number of desired skills is raised.

To get a better idea of how the division of labor develops through evolution, we select a typical run for each of {5, 10, 15, 20} skills for further analysis. Figure 5.3 depicts the maximum and average number of unique skills of all groups over 500 generations. Starting from at most 2 skills, the best performing group gradually evolves to perform more and more different skills until the number of desired skills was reached (see Fig. 5.3a). Such growth is due to the guidance provided by group fitness. Take the run for 20 desired skills for example. We collect the following information from this run: group fitness, number of activated skills, and percentage of cooperators in the best performing group, as well as percentage of cooperators in the population; see plot in Fig. 5.4.

Group fitness (refer to Eq. 5.3.1) is determined by average individual fitness and percentage of activated skills. We plot percentage of cooperators instead of average individual fitness in the best group, because (i) we can easily extrapolate average individual fitness from this percentage, and (ii) it also shows the fixation process in the best group. Figure 5.4 clearly shows how the percentage of cooperators and the number of activated skills affect the group fitness. Interestingly, we notice that the population converges to cooperators first, and then the best group develops all required skills. The same trend is also observed in other runs with 5, 10, 15 skills. This observation indicates that cooperators spread in the population before the evolutionary transition happens, a result confirming the discussion about the relationship



(a) Maximum number of unique skills



(b) Average number of unique skills

Figure 5.3: The changes of the maximum and average number of unique skills in a typical run.

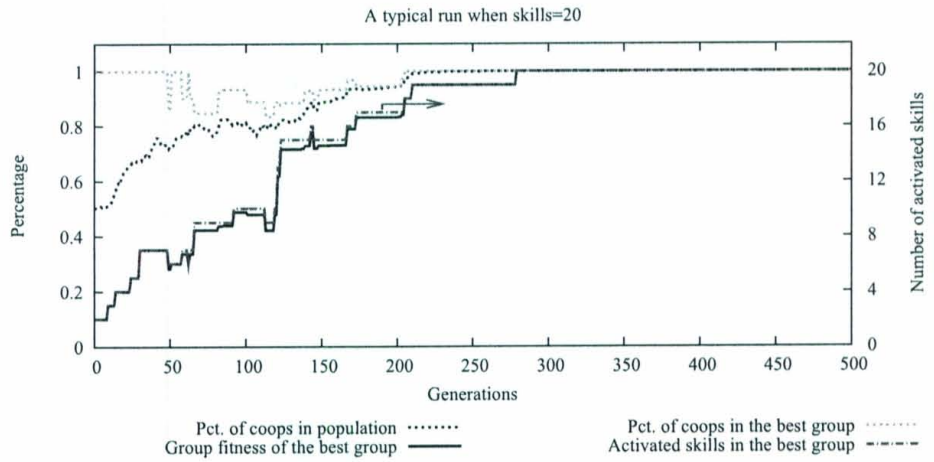


Figure 5.4: The changes of group fitness, percentage of cooperators and activated skills when 20 skills are set.

between MLS1 and MLS2. Group fitness, in turn, influences the execution of individual evolution and group evolution (i.e. cooperation operator). Since defectors yield no fitness benefit on group levels, they are eliminated from the population by group selection at reproduction; hence the percentage of cooperators in the best group and in the population increases steadily towards 1. As shown in Fig. 5.3b, the average number of activated skills never approaches to the number of desired skills. This implies that the population maintains groups with various skills. They are potential building blocks, out of which the cooperation operator is able to test different combinations of existing groups, and gradually hone in on optimal groups with all required skills.

In summary, our model is able to successfully evolve groups with all desired skills for the extended NPD game; or we can say that our model is able to evolve groups to engage in the division of labor between equally rewarding skills.

5.3.3 Varying Rewards

The second part of this experiment continues the exploration of whether or not our model can evolve the division of labor, but this time with unequally rewarded skills. The different rewards put extra pressure on accomplishing this task, as they encourage individuals to specialize on the most rewarded skills while avoiding the less rewarded skills.

To distinguish skills with different reward, we refer to the "leader/follower" situation described by Goldsby *et al.* [36]. Individuals whose *skill_id* is set to 1 are appointed to the leader of that group, while individuals performing other skills are simply followers. Leaders receive different rewards than followers, but followers, no matter what specific skills they have, receive no other rewards. A coefficient, α, is used to control how much reward a leader can receive. Coefficient α basically is a multiplicative of the individual fitness; that is, the individual fitness of a leader is calculated as the product of α and the individual fitness obtained by Eq. 5.1.2a or Eq. 5.1.2b.

We vary the value of α in the range of {0.5, 2, 4, 8, 64} on each of {5, 10, 15, 20} skills, and run the model on each setting 20 times. The performance is summarized in Table 5.9. Clearly for each setting the population converges to cooperators as a result of MLS1, and the best performing group is composed of cooperative individuals with all required skills as a result of MLS2.

Because group fitness can hardly converge in this experiment, the convergence speed $S_{converge}$ is judged by the stabilization of $P_{fixation}$ and $S_{activated}$. Fig. 5.5 displays a typical run when the number of desired skills is set to 5 and coefficient α is set to 8. Although the percentage of cooperators in the population and the number of activated

Table 5.9: The performance of our model when leaders are assigned with various rewards

Settings		$P_{fixation}$	Activated Skills	Convergence Speed
skills=5	$\alpha = 0.5$	1	5	90.45
	$\alpha = 2$	1	5	145.35
	$\alpha = 4$	1	5	193.00
	$\alpha = 8$	1	5	238.10
	$\alpha = 64$	1	5	330.00
skills=10	$\alpha = 0.5$	1	10	152.2
	$\alpha = 2$	1	10	232.40
	$\alpha = 4$	1	10	379.05
	$\alpha = 8$	1	10	488.00
	$\alpha = 64$	1	10	607.75
skills=15	$\alpha = 0.5$	1	15	196.60
	$\alpha = 2$	1	15	313.80
	$\alpha = 4$	1	15	531.50
	$\alpha = 8$	1	15	696.55
	$\alpha = 64$	1	15	950.55
skills=20	$\alpha = 0.5$	1	20	314.80
	$\alpha = 2$	1	20	407.35
	$\alpha = 4$	1	20	586.85
	$\alpha = 8$	1	20	902.35
	$\alpha = 64$	1	20	1394.75

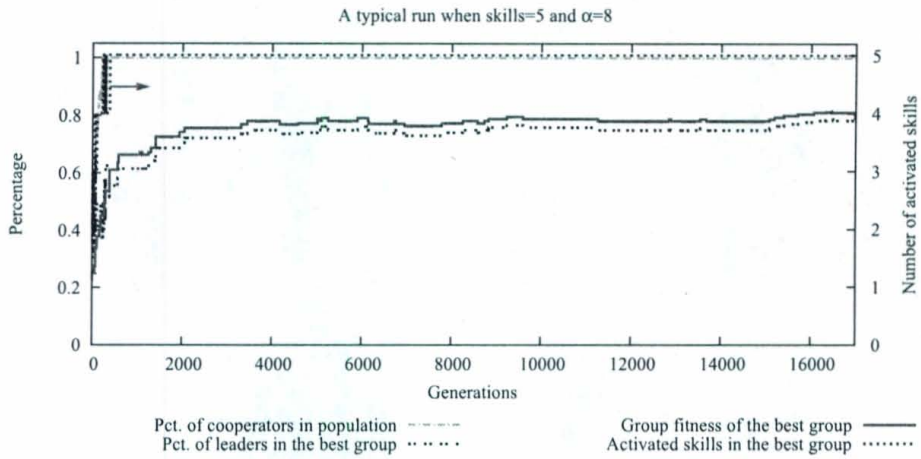


Figure 5.5: A typical run when the number of skills=5 and α=8.

skills in the best group converge quickly (around generation 350), group fitness and the percentage of leaders in the best group never stop increasing. After generation 350, the percentage of leaders is the only factor that changes group fitness. Leaders in this case receive much higher rewards than followers, and maximizing this percentage at the same time maximizes the group fitness. Therefore, both values are constantly improving. Because there is no upper bound on group size, the cooperation operator keeps creating larger groups with more leaders; therefore an equilibrium distribution of different skills can hardly be reached.

To facilitate the investigation on how different rewards affect the division of labor, we restrict the maximum group size to 20. We plot in Fig. 5.6 the percentage of leaders in the best performing group collected from a typical run when α is set to each of $\{0.5, 2, 4, 8, 64\}$. When α is set to 0.5, 5% of 20 individuals, which is only 1 individual, play the role as a leader, while when α is set to 2, 55% of the group, that is 11 individuals, choose to be a leader; similarly, 15 out 20 individuals (75%) become

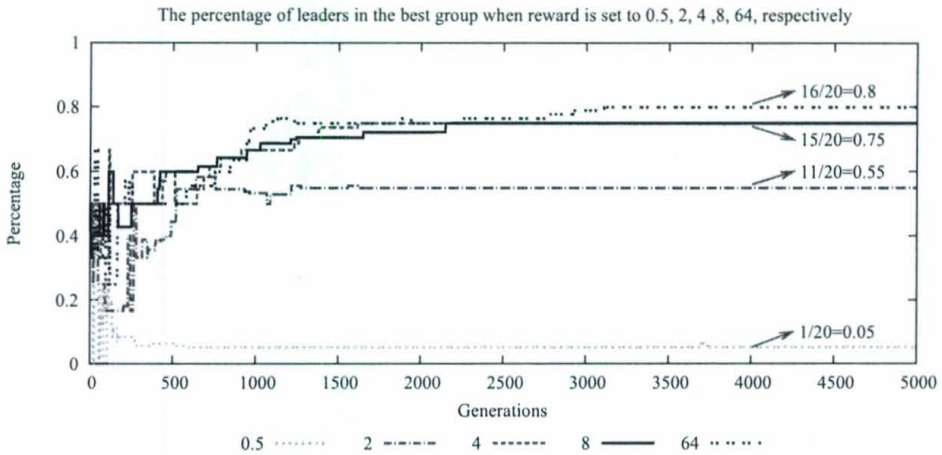


Figure 5.6: The percentage of leaders in the best group when α is set to 0.5, 2, 4 ,8, 64, respectively.

the leader when α is 4 or 8, and 16 leaders (80%) when α is 64.

When α is less than 1, leaders are in fact receiving a penalty, not a reward. Very naturally, individuals avoid becoming a leader, but because of selection pressure at the group level, a leader must be present in a group. Therefore, the best group ends up with only 1 leader, which maximizes the group fitness. In contrast, when α is greater than 1, individuals strive to be leaders because of the positive reward. An α value of 64 shows another extreme distribution of different skills. Driven by such a significant reward, the best group only has 4 individuals as followers, playing the other 4 skills, respectively, while all other individuals are leaders. The higher the reward, the greater the number of leaders in a group, and the slower the population converges (see $S_{converge}$ column in Table 5.9).

This experiment perfectly shows the adaptability of our model in response to changes in group selection pressure, and the importance of selection pressure on group

levels in developing division of labor. Selection pressure eliminates defectors from a population, adjusts the distribution of skills according to the received reward or penalty, and forces all skills to be present even though some of them have lower fitness than others.

5.4 Chapter Summary

In this chapter, we first investigated the capability of our computational multilevel selection model to evolve cooperation on the NPD game. The experiments confirmed that the evolution of cooperation can be promoted in our model under a wide range of selection pressures and initial fractions of cooperators. When compared with the improved Traulsen model, cooperation in our model more easily emerges and can be sustained, and is less affected by parameter changes. The experiments also highlight the essential roles of group selection and cooperation operator in encouraging and accelerating the process to reach cooperation.

The second experiment investigated the transition ability of our model on the extended NPD game for achieving division of labor. Compared to the first experiment, this experiment defines a reaction rule to map individual genotype to group genotype, and redefines group fitness to specify the new trait groups have to adapt to. The results demonstrate that groups with all required skills can emerge from a population of independent individuals, no matter whether the skills are equally rewarded or not.

The two sets of experiments highlight the importance of multilevel selection. Guided by fitness definitions, selection puts sufficient pressure onto the population to ensure that appropriate adaptations, such as cooperation or division of labor, ap-

pear on different levels. Also the experiments showed the flexibility of our model in switching between two types of multilevel selection. To achieve the evolution of cooperation, only MLS1 is required, since MLS1 propagates cooperators in a population. However, to achieve evolutionary transitions, both MLS1 and MLS2 are necessary, each at a different stage. MLS1 happen first; only when participating individuals are willing to cooperate, will evolutionary transitions occur. MLS2 forces complexes to evolve adaptations, which gradually develop into new group traits. Our experiments thus confirm the findings of Okasha [74, 75, 76].

In conclusion, the experiments conducted in this chapter validate the feasibility of multilevel selection in promoting cooperation in spite of the competition introduced by evolution, and the possibility to achieve at least a very simple type of evolutionary transitions.

Chapter 6

Experiments on String Covering Problems

The discussion in Chapter 5 helped us to understand how cooperation emerges from evolution in our multilevel selection model. In fact, the mechanisms employed by our model are strong enough that it reaches the evolution of cooperation easier than other group selection models, and it even obtains a simple type of evolutionary transitions, a more advanced topic that builds on the evolution of cooperation. In this chapter, we will concentrate on the other aspect of our model, namely, problem decomposition; that is, how to determine an appropriate number of subcomponents and the precise role each will play. We are curious whether our model is able to produce good decompositions without *a priori* knowledge, during which cooperation is also required to assist the algorithm determine the role of each subcomponent. To this end, the Hierarchical Evolutionary Algorithm (HEA) introduced in Chapter 4 will be applied to string covering problems. String covering problems are a typical decomposable problem, providing a relatively simple environment in which the emergent decomposition

properties of our model can be studied.

This chapter is organized as follows. Sect. 6.1 describes string covering problems, and the data sets used in this study. Sect. 6.2 discusses the customization of HEA when applied to string covering problems, such as defining individual fitness, group fitness and evolutionary operators. Sect. 6.3 investigates the performance of HEA, and compares the results with those obtained from a classic Evolutionary Algorithm (EA), a Cooperative Co-evolutionary Algorithm (CCEA), and an Individual Evolutionary Algorithm (IEA). We choose these three control algorithms, because we would like to show how well HEA extends classic EAs to achieve cooperation, and how good the solutions of HEA are when compared to the results of the CCEA and IEA, in which problem decomposition is completed manually.

6.1 The String Covering Problem

The string covering problem [81] aims to discover a **set** of N binary strings that matches as strongly as possible another set of K binary strings, where K is typically much larger than N. The N and K binary strings are called match set (M) and target set (T), respectively. Strings in M and T have the same length. The matching strength S between binary string x and y of length l is determined by the number of bits in the same position with the same value, as follows:

$$S(x,y) = \sum_{i=1}^{l} \begin{cases} 1 & \text{if } x_i = y_i, \\ 0 & \text{otherwise.} \end{cases} \tag{6.1.1}$$

Examples in [80] have shown that the matching strength associated with one string in a match set will be warped by a change in another string. That is to say, strings

in a match set are interdependent. The goal of any algorithm for string covering is to find a composite solution, which is M, consisting of multiple interdependent subcomponents, which are matching strings. Obviously, the string covering problem is a typical decomposable problem.

In addition, this composite solution has to satisfy two contradictory criteria: generalization and specialization. If every string in T has to be matched by at least one string in M, the strings in M must contain patterns shared by multiple target strings. Hence, M must have the capacity to generalize. However, on the other hand, matches should be as strong as possible (i.e. to maximize Eq. 6.1.1); so the match set also needs to be specific. The size of M affects the balance between generalization and specialization. Given M has to cover all strings in T, the larger the size of M, the more specific M can be; the smaller the size of M, the more general M must be. An optimal M should minimize the size, but maximize the matching strength.

The string covering problem is an excellent test application for investigating Cooperative Evolutionary Algorithms (CEAs), because of the following reasons: 1) it is a typical decomposable problem; 2) it provides a relatively simple environment in which the emergent decomposition properties of CEAs can be studied [80]; 3) researchers are able to construct artificial string covering problems with known optima in different fitness landscapes; 4) it is practical, as the implemented algorithms and the experimental findings can be easily applied to similar application domains, such as other instances of set covering problems, or to model a number of complex processes from nature, such as self-nonself discrimination in the immune system [89].

In fact, string covering problems have been used by several CEAs in the literature. Forrest *et al.* [28, 89] designed an artificial immune system for pattern recognition

in which the patterns were hidden in target strings of string covering problems. A genetic algorithm with implicit fitness sharing was a central component of this system, where implicit fitness sharing was responsible for maintaining diversity in a matching string population in order to cover different patterns in target sets. Potter *et al.* [81] used the string covering problem to investigate CCEAs on locating and covering multiple environmental niches, finding an appropriate level of generality, and evolving an appropriate number of species.

Our research also uses string covering problems as a test bed for the same purpose. Four target sets are generated by the four schemata shown in Fig. 6.1, respectively. Each schema contains at least two 64-bit string templates with a fixed region (marked

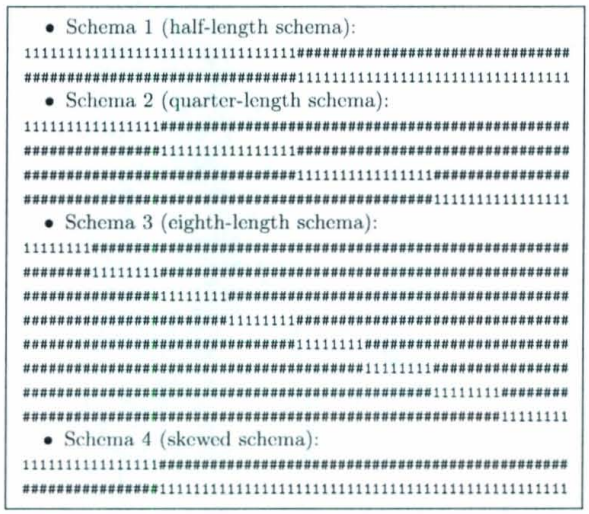


Figure 6.1: The four target sets used in this study are generated from above schemata.

by 1's) and a variable region (marked by #'s). Target strings created by a string template will share the same fixed region, but have randomly decided 0's and 1's in

the variable region. The first 3 schemata were borrowed from research work by Potter *et al.* [81]. We generate 200 target strings from each single string template; hence in total there are 400 target strings in target set 1, 800 in set 2 and 1600 in set 3. The fourth schema is a new schema we introduced for the purpose of this study. We created a skewed data distribution for target set 4 by generating 200 strings from the first template and 20 from the second. Based on the design of these four schemata, we can easily estimate that the optimal solution for a target set should contain the same number of matching strings and the same patterns as the corresponding schema.

6.2 Algorithm Customization

6.2.1 Representation

Individuals are represented as 64-bit strings on the alphabet {1, 0, *}. Thus, the evolutionary algorithm employed by HEA is a genetic algorithm. Compared to {1, 0} used in previous work [28, 81, 89], "*" is a new symbol called "don't care". It represents either "1" or "0" in a position, whose value is not shared by most strings in a target set. This change allows us to easily assess the accuracy of solutions and their level of generality. Each individual is also assigned a unique ID. Since the hierarchy created in this study is constitutive, groups are simply represented as a list of individuals. No mapping rules are defined.

6.2.2 The Fitness Functions

We define the individual fitness function in the following way:

$$f(x) = \alpha \times Ratio(x) \times Cvrg_{\text{idv}}(x), \tag{6.2.1}$$

where *Ratio* shows the percentage of non-"*" symbols in the representation of individual x, $Cvrg_{\text{idv}}$ the string coverage of individual x, calculated by the number of target strings covered by individual x over the cardinality of the target set, and α is a weighting coefficient. If we say that an individual covers a target string, we mean that every non-"*" symbol in an individual has the same value on the same position in a target string. Basically, the individual fitness function is looking for a specific individual (individuals with more non-"*" symbols), but at the same time with a high string coverage.

Once we know how the performance of individuals is measured, our intention of conceiving the four different target sets is obvious. For the first three target sets, solutions are becoming harder to find as the number of subcomponents increases and fixed regions become progressively shorter with respect to the variable regions of the string templates. Target set 4 is the only set whose solution contains subcomponents with **unequal** fitness, because of the unequal length in the fixed regions and its skewed data distribution. Therefore, target set 4 presents a more difficult problem than the others.

Group fitness is defined as

$$g(y) = \frac{\sum_{i=0}^{n} f(x_i)}{n} \times Cvrg_{\text{grp}}(y), \tag{6.2.2}$$

which considers the average individual fitness in group y and the string coverage, $Cvrg_{\text{grp}}$, of group y, where n is the group size. In this experiment, and also the ex-

periment shown in the next chapter, we include average individual fitness as a part of group fitness for purely engineering reasons (the discussion of transition from MLS1 to MLS2 for this experiment is left for future work). A group with high coverage may have resulted from very general individuals (individuals with less non-"*" symbols) with high string coverage, i.e., individuals with low fitness. By adding average individual fitness, group fitness now favors groups whose individuals cooperate to provide maximum coverage, while each individual is optimized to specialize its role in the cooperation.

6.2.3 Algorithmic Description

The Hierarchical Evolutionary Algorithm (HEA) applied to the string covering problems, as shown in Algorithm 5, is completed in 5 steps.

Initialization Lines 1 and 2 initialize the population with N randomly created individuals, each with a unique ID. They become the lowest level in the hierarchical structure. Individuals are independent and competitive with each other, without being aware of collaborative goals.

Evolution on group levels Lines 4 to 8 determine the evolution on group levels. In every generation, up to m new groups are created by three evolutionary operators, cooperation, crossover and mutation, each applied with a user-defined probability. Cooperation mixes together individuals and groups on all levels, from which two are selected to form a new group. If individuals with the same ID exist in a group, only one individual is kept. Crossover and mutation, in contrast, select parents from groups only. One-point crossover is used; two groups exchange individuals starting from a randomly selected crossover point in each group. Mutation randomly adds, removes,

Algorithm 5: The hierarchical evolutionary algorithm

```
1  P ← Initialize_Population(N);
2  Evaluate_Individual_Fitness (P);
3  while population does not converge or max generation is not reached do
4      for i ← 0 to m do
5          gp ← Reproduce_a_Group(P);
6          Evaluate_Group_Fitness(gp);
7          Add_a_Group_to_Population(gp, P);
8      end
9      for i ← 0 to n do
10         idv ← Reproduce_an_Individual(P);
11         Evaluate_Individual_Fitness (idv);
12         Add_an_Individual_to_Population(idv, P);
13     end
14     Niching_on_Individuals ();
15     Niching_on_Groups ();
16     P' ← Survival_Selection (P, N, M);
17     P ← P';
18 end
```

or replaces an individual in a group. For simplicity, we only consider removing individuals in this study. The probability of selection, unless specified, in all cases is proportional to fitness. Once a new group is created, its fitness is evaluated. The validation of a group should also be checked before it is added into the population. A group is valid if there are no members having exactly the same contribution, which in this study is the same coverage. This eliminates free riders from a group and prevents group size from increasing unnecessarily.

Evolution on the individual level Lines 9 to 13 ensure that no more than n offspring with new IDs are produced on the individual level. To select a parent individual for reproduction, a group has to be selected first based on its fitness, from which an individual is subsequently selected. Roulette wheel selection is employed in the between-group and within-group selection. Crossover on individuals randomly selects a position on each individual and exchanges the following l bits, where l is less than the length of the chromosome. Bit-flip mutation is then conducted. Finally, fitness of new individuals is evaluated.

Niching Lines 14 and 15 conduct niching on individual and group levels. Preserving diversity is mandatory on individual and group levels with the purpose of maintaining all subcomponents in final solutions. A revised fitness sharing method is used here. We first establish a niche for each individual, which includes individuals whose genotypic Euclidean distance from the focal individual is within a sharing radius. That is to say, individuals in the same niche are similar in genotype. If this individual does not have the best fitness in the niche, its fitness has to be reduced by the following equation

$$f(x) = f(x) * \frac{Avg_{\text{distance}}(x)}{Radius} \tag{6.2.3}$$

where Avg_{distance} is the average distance between individual x and others in its niche, and *Radius* is the sharing radius. The advantage of considering this fitness sharing is that it preserves diversity but not at the expense of the best individuals in each niche. Niching on groups is conducted in a similar way, except that the similarity between groups is judged by the overlap in string coverage. Groups are penalized if they share string coverage with others. However, if a group covers new strings that never appear in the coverage of the best group from the previous generation, it will be rewarded on fitness. Therefore, group fitness is adjusted by Eq. 6.2.4.

$$g(y) = g(y) + \left(\frac{Cvrg_{\text{grp}}^{\text{new}}(y)}{K} - \frac{Avg_{\text{overlap}}(y)}{Cvrg_{\text{grp}}(y)} \right) \tag{6.2.4}$$

where $Cvrg_{\text{grp}}^{\text{new}}$ is the new string coverage of group y, K the size of the target set, Avg_{overlap} the average overlapped string coverage of group y, and $Cvrg_{\text{grp}}$ the string coverage of group y.

Survival selection The population size is expanded after the evolution on individual and group levels. The survival selection on Lines 16 and 17 reduces the population to its original size by saving the best N individuals and M groups to the next generation. The individuals and groups are sorted with respect to their fitness values, adjusted by niching. After survival selection, fitness of individuals and groups will be restored to their original value, so they can compete fairly with new individuals or groups created in the next generation. Please note, unlike the design in chapter 5, the copies of the replaced individuals will not be updated, i.e., when an individual is replaced by another individual on individual level, its copy on group levels will not be replaced. This is another way to maintain the diversity in the population.

The above steps are repeated until a predefined termination criterion is reached, e.g. the maximum number of generations, or a desired fitness or accuracy.

6.3 Experiments

In order to test whether cooperation can indeed emerge through the evolutionary process defined by HEA, and solutions can indeed be built hierarchically from smaller independent subcomponents, we ran experiments on the four target sets discussed in Sect. 6.1. For each set, we expect the algorithm to return a match set whose matching strings have the exact patterns shown in the string templates of the corresponding schema. We compare the results with those produced by the three control algorithms: a classic EA, a CCEA, and an IEA.

6.3.1 Experimental Setup

The three control algorithms use the same chromosome structure as HEA, and also apply the same crossover and mutation operators to change individuals. The difference lies in the evolutionary process, hence the way to define fitness. Both the classic EA and IEA use Eq. 6.2.1 as individual fitness. Because of a change in the alphabet used to encode genes, we redefine the matching strength function as follows:

$$S(x,y) = \sum_{i=1}^{l} \begin{cases} 2 & \text{if } x_i = y_i = \text{“1”}, \\ 1 & \text{if } x_i = \text{“\#”}\ , \\ -1 & \text{if } x_i \neq \text{“\#”} \text{ and } x_i \neq y_i. \end{cases} \tag{6.3.1}$$

where x and y are strings of a match set and target set, respectively. This equation replaces the match function used in the fitness function of the CCEA defined in [81], which is shown in Eq. 6.3.2.

$$MS(M) = \frac{1}{K} \sum_{j=1}^{K} Max(S(x_1, y_j), \ldots, S(x_N, y_j)) \tag{6.3.2}$$

where K is the target set length, and N is the match set length. This fitness function averages the largest match strength between match set M and target set T.

IE is the only algorithm with an explicit team fitness, or global fitness in terms of IEAs. We define the global fitness, $f_g(t)$, of the collaboration M at generation t as follows:

$$f_g(t) = \#\text{ of target strings covered by } M. \tag{6.3.3}$$

In order to reflect the quality of individual x in a collaboration, we measure the changes in global fitness in two successive generations, shown in Eq. 6.3.4 where $f_l(x)$ indicates the fitness of individual x.

$$g_1(x) = f_l(x) \times \left[\frac{f_g(t)}{f_g(t-1)}\right], \forall x \in M. \tag{6.3.4}$$

It is very likely that individuals in a collaboration in generation t cannot provide full coverage. Therefore, if an individual outside of a collaboration covers β new data samples, we reward it based on Eq. 6.3.5.

$$g_2(x) = (f_l(x^{best}) - f_l(x)) \times \frac{\beta}{f_g(t)}, \forall x \cap \overline{M} \neq \emptyset. \tag{6.3.5}$$

where $f_l(x^{best})$ is the fitness of the best individual in the population. Once $g_1(x)$ and $g_2(x)$ are calculated, feedback will be given to individuals as follows:

$$f_l(x') = \begin{cases} f_l(x) + \lambda_1 g_1(x) & \text{if } x \in M, \\ f_l(x) + \lambda_2 g_2(x) & \text{if } x \cap \overline{M} \neq \emptyset, \\ f_l(x) & \text{otherwise.} \end{cases} \tag{6.3.6}$$

where $f_l(x')$ is the fitness of individual x after adjustment, λ_1 and λ_2 are user defined parameters.

We ran HEA and the three control algorithms on a PC with an AMD $Turion^{TM}$ 64×2 CPU at 1.6GHz and with 2 GB of RAM. The parameter settings are shown in Table 6.1. The four target sets are denoted as ts1, ts2, ts3, and ts4, respectively.

Table 6.1: Parameter settings

Parameter	Classic EA	CCEA	IEA	HEA
Run	50	50	50	50
Generation	1000	1000	2000	2000
Number of groups	N/A	N/A	1	10
Cooperation rate	N/A	N/A	N/A	0.5
Crossover rate	0.95	0.95	0.95	0.95
Mutation rate	0.05	0.05	0.05	N/A
Group size	N/A	ts1/4:2	ts1/4:2	N/A
		ts2:4	ts2:4	
		ts3:8	ts3:8	
Fitness coefficient	ts1/4:4	N/A	ts1/4:4	ts1/4:4
	ts2:16		ts2:16	ts2:16
	ts3:64		ts3:64	ts3:64
Niching radius	N/A	N/A	ts1/2:0.7	ts1/2:0.7
			ts3:0.5	ts3:0.5
			ts4:0.9	ts4:0.9

We measured the performance of all algorithms by convergence time and average number of mismatched bits. Convergence time is the number of seconds an algorithm

needs to find the best solution. In our evaluations this is a better indicator for evolutionary speed than the number of fitness evaluations, given the fact that the algorithms conduct fitness evaluation differently (e.g. CCEAs only consider fitness on the collaboration level), which cause differing amounts of time to complete. To calculate the average number of mismatched bits, we first count the number of different bits between each string template used by a target set and the closest matching string returned by an algorithm, and then average over all string templates.

6.3.2 Evaluating HEA and Control Algorithms

We ran HEA and the three control algorithms on all four target sets. Table 6.2 shows average performance of the algorithms over 50 runs. Standard deviation of convergence time and of average number of mismatched bits are enclosed in brackets.

In order to allow a fair comparison, given the same target sets, all algorithms have the same amount of individuals in their population, as this number has direct implications for the evolutionary speed and solution quality. Species size in CCEA is calculated by dividing the total population size by the number of species. We also ran another set of experiments on CCEA using this number as the size of species, and kept the results for reference.

For target sets 1 and 4, HEA always found a match set with **2** match strings, which perfectly matched all target strings; in other words, there was no mismatch in either case. However, it took a longer time for runs to converge on target set 4 than for runs on target set 1, because exploring and maintaining multiple match strings with unequal fitness is more difficult. For target sets 2 and 3, all match sets returned by HEA contained **4** and **8** match strings, respectively. Because both sets obtained

Table 6.2: Performance of four algorithms on target sets 1, 2, 3, and 4

(a) Performance of four algorithms on target set 1

Algorithms	*Target set 1*		
	# of Idv.	Convergence Time (Sec.)	Mismatch Bits
HEA	**20**	**1.539 (0.509)**	**0.000 (0.000)**
Classic EA	20	0.595 (0.169)	32.00 (0.000)
CCEA[1]	20 (10×2)	1.658 (0.289)	0.617 (0.462)
CCEA[2]	40 (20×2)	2.236 (0.397)	0.600 (0.536)
IEA	20	3.171 (1.091)	0.717 (0.429)

(b) Performance of four algorithms on target set 2

Algorithms	*Target set 2*		
	# of Idv.	Convergence Time (Sec.)	Mismatch Bits
HEA	**20**	**10.828 (3.239)**	**0.095 (0.256)**
Classic EA	20	2.140 (1.113)	24.017 (0.207)
CCEA[1]	20 (5×4)	19.664 (9.246)	0.758 (0.350)
CCEA[2]	80 (20×4)	23.132 (7.429)	0.592 (0.282)
IEA	20	7.560 (1.529)	0.708 (0.378)

Table 6.2: Continued.

(c) Performance of four algorithms on target set 3

Algorithms	*Target set 3*		
	# of Idv.	Convergence Time	Mismatch Bits
HEA	**40**	**20.665 (6.801)**	**0.088 (0.145)**
Classic EA	40	12.743 (3.717)	12.225 (5.298)
CCEA[1]	40 (5×8)	289.060 (57.951)	0.950 (0.270)
CCEA[2]	320 (40×8)	367.540 (144.319)	0.467 (0.183)
IEA	40	59.323 (18.991)	0.504 (0.268)

(d) Performance of four algorithms on target set 4

Algorithms	*Target set 4*		
	# of Idv.	Convergence Time	Mismatch Bits
HEA	**20**	**3.020 (1.664)**	**0.000 (0.000)**
Classic EA	20	0.350 (0.372)	32.000 (0.000)
CCEA[1]	20 (10×2)	5.258 (2.170)	1.983 (0.517)
CCEA[2]	40 (20×2)	5.970 (2.839)	1.867 (0.472)
IEA	20	1.981 (0.987)	0.983 (0.160)

low average number of mismatched bits and relatively high standard deviation, we further collected the median as extra evidence, which was 0 on both sets. The three numbers together indicate that most runs returned correct match sets. Convergence time on the two target sets was longer because the difficulty of the problem increased.

We now compare the four algorithms on the first three target sets. The classic EA can only find one matching string out of many, as all matching strings for a target set are equally good in terms of specialization and coverage. That the algorithm fails this task is no surprise, because we know that it has a strong tendency to converge. Given the same population size, CCEA outperforms IEA only on simple target sets, but not on hard ones. It cannot beat HEA on any of the three sets because of two limitations. First, CCEA does not maintain diversity within species; the way fitness is defined only helps to preserve diversity between species. In our experiments, the algorithm converged at generation average 88.8 for target set 1, given 40 individuals, at generation average 125.467 for set 2 on 80 individuals, and at generation average 263.226 for set 3 on 320 individuals. Once species have lost their diversity, the algorithm stops exploring the search space. Second, individual fitness depends on exactly who is in a collaboration, so it does not accurately measure the performance of individuals. As a result, the search will drift to suboptimal solutions. Increasing population size, though improving accuracy somewhat, provides little help to overcome these limitations. HEA also performs better than IEA on all test runs. The difference between the two algorithms is the choice of group selection. IEA only composes a single group by selecting the best n individuals from the population (where n is the group size), while HEA forms more than one group with various sizes and compositions, and considers the evolution on group levels. From this perspective, it increases the possibility

of finding a solution faster.

Target set 4 is a new set we introduced in this study, which requires algorithms to optimize and maintain multiple subsolutions with unequal fitness simultaneously. It has proven to be the hardest case among the four sets; the classic EA converged to a string with 48 1's despite the low fitness, as searching for such a string is much easier than searching for a string with 16 1's; CCEA and IEA both obtained the highest average number of mismatched bits on this set. In contrast, HEA found a perfect match set very quickly.

Table 6.3 shows the statistical comparison of HEA over the two control algorithms w.r.t. convergence time and average mismatch bits, using the two-tailed t-test with 98 degrees of freedom at a 0.05 level of significance. Since the p-value is less than

Table 6.3: The T-test results between HEA and the two control algorithms.

Target set		CCEA[1]	CCEA[2]	IEA
ts1	Time	0.227	4.807E-07	6.287E-08
	Mismatch	8.053E-08	7.061E-07	4.802E-10
ts2	Time	2.346E-05	2.499E-09	1.787E-05
	Mismatch	2.758E-10	2.155E-07	3.568E-08
ts3	Time	1.00E-21	8.007E-14	3.286E-12
	Mismatch	3.156E-16	1.815E-10	2.741E-08
ts4	Time	5.108E-05	6.856E-05	0.015
	Mismatch	4.218E-19	1.891E-19	8.405E-25

0.05 (except the convergence time of CCEA[1] on ts1), we can conclude that on string

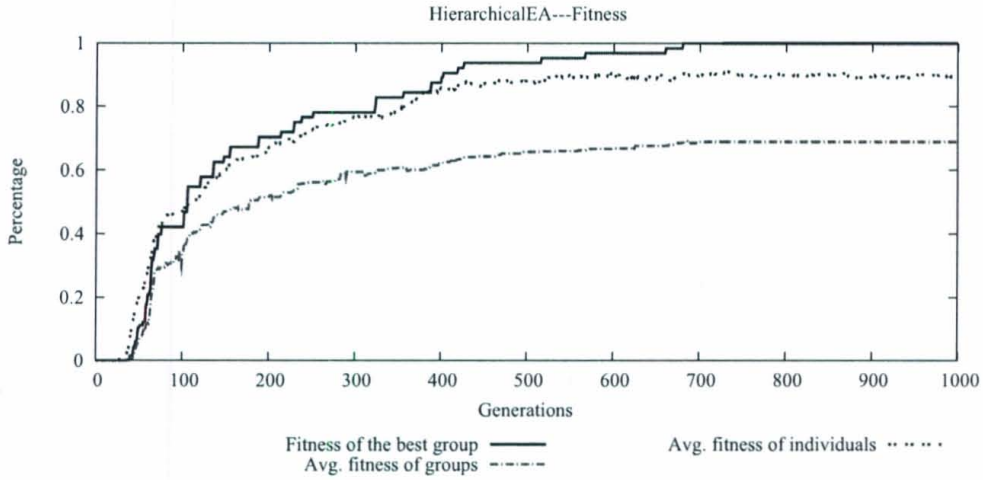
covering problems, HEA with the emergent problem decomposition property achieved a significant improvement on accuracy and evolutionary speed when compared to CCEA and IEA, which required predefined problem decomposition.

6.3.3 Looking Inside of HEA

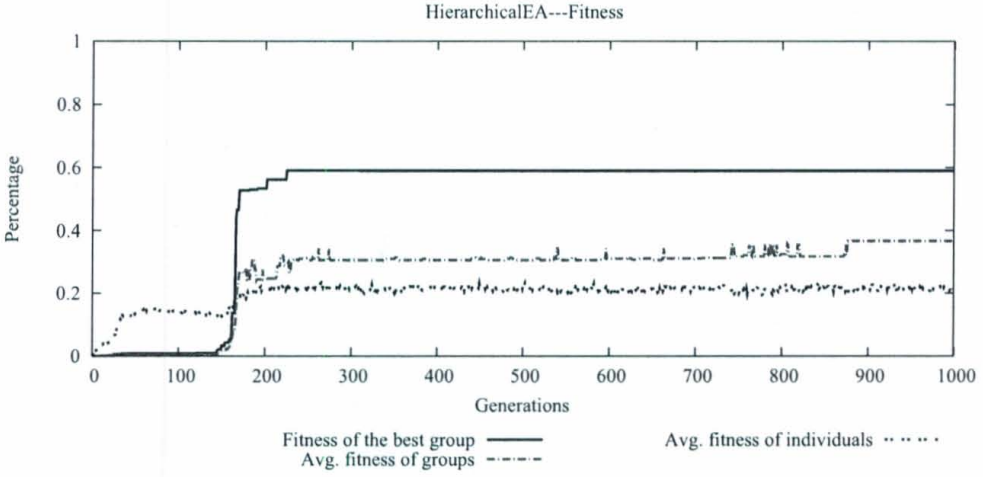
In order to get a better idea why HEA achieves automatic problem decomposition while at the same time evolves faster and finds more accurate solutions than the other cooperative EAs, we investigate the algorithm by examining its performance in a typical run on target sets 2 and 4. We choose these two sets because they represent two different situations, namely equal and unequal fitness of subcomponents in a solution. Target sets 1 and 3 are not discussed here because they share the same features with target set 2.

Fitness is always a good place to start investigations as it reflects how evolution proceeds. Fig. 6.2(a) and (b) depict fitness related information in a typical run on target sets 2 and 4, respectively. We show the fitness of the best group, average fitness of individuals and average fitness of groups. Individual fitness and group fitness by definition are affected by coverage and specialization (the number of non-"*" symbols in the representation). Therefore, we also show average specialization of individuals and of the best group in Fig. 6.2(c) and coverage of the best group in Fig. 6.2(d).

As we can see clearly in Fig. 6.2(a) and (b), average individual fitness and group fitness improve steadily due to the evolution happening on individual and group levels. As a result, the fitness of the best group increases constantly on both sets. To be more specific, HEA optimizes the coverage first (see Fig. 6.2(d)), because increasing the coverage will improve both individual and group fitness. However, the different

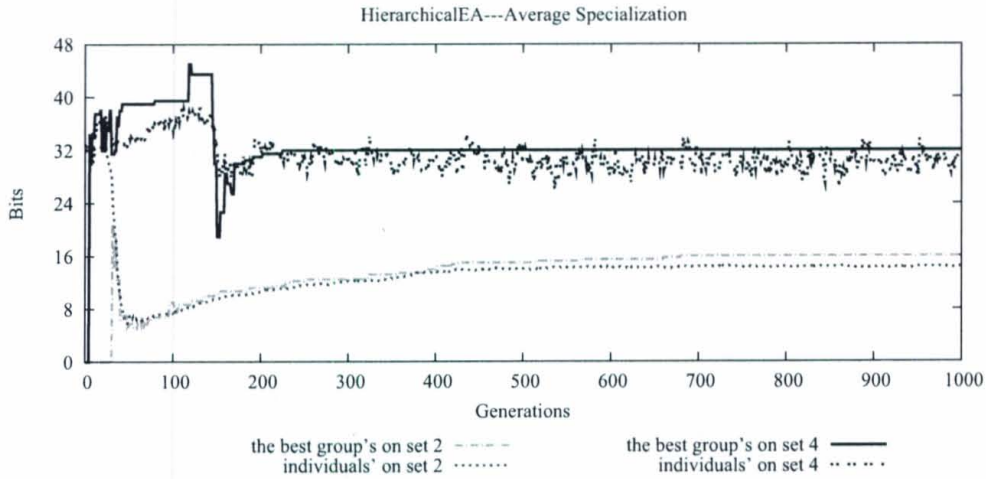


(a) Fitness changes on target set 2.

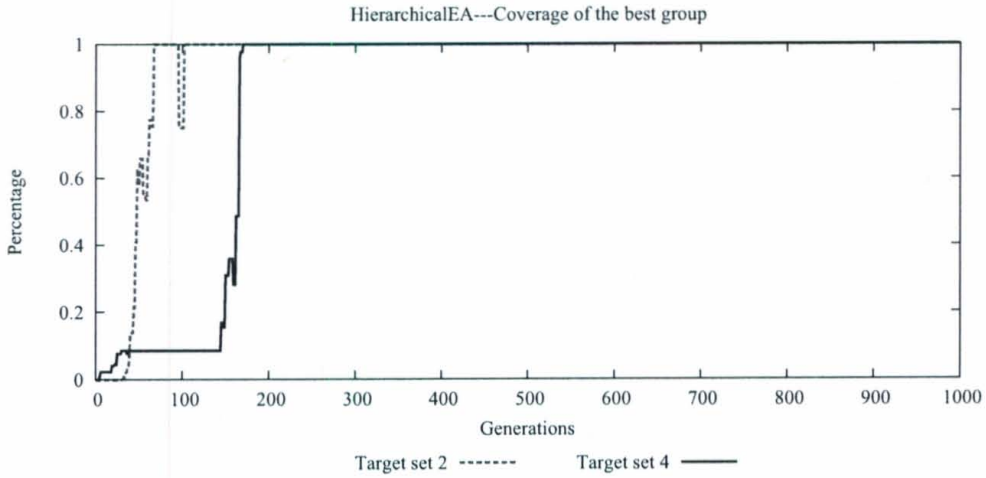


(b) Fitness changes on target set 4.

Figure 6.2: Typical runs on target sets 2 and 4.



(c) Average specialization on target set 2 and 4.



(d) Coverage of the best group on target set 2 and 4.

Figure 6.2: Continued.

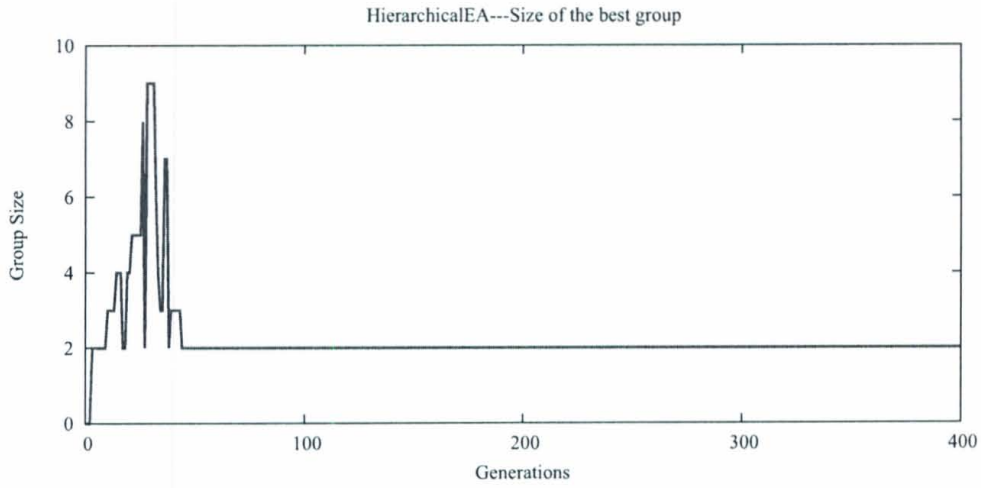
properties of the two sets cause HEA to behave differently at this stage. Individuals are randomly generated at the initialization, so their average specialization on both sets is around 32 at the outset (see Fig. 6.2(c)). For target set 2 which requires 16 1's in all match strings, the specialization has to drop in order to maximize the coverage. For target set 4, the specialization is increased first to optimize the coverage of the match string with 48 1's, and then decreased to optimize the coverage of the one with 16 1's. After coverage hits 1 (i.e. coverage has been optimized), HEA then optimizes the second part of the individual fitness, so we see that the specialization on both sets increases.

The run on target set 4 demonstrates very well the contribution of group selection to encourage cooperation, regardless of an individual's fitness. As shown in Fig. 6.2(c), during the first 200 generations, no matter whether the average specialization of the best group is moving towards 48 or 16, the average specialization of individuals always keeps a distance. This implies that even though HEA optimizes the matching string with 48 1's in the first place, a few individuals covering target strings with 16 1's have managed to stay in the population. The reason is because such individuals provide new coverage to their group (i.e. they increase group fitness), hence the group and the individuals inside are more frequently selected and optimized. Therefore, they gradually dominate the population (the average specialization of the best groups and individuals begin to drop). Similarly, after the coverage hits 1, the specialization of individuals with 48 1's continues to increase, despite their low fitness and the domination of high fitness individuals with 16 1's. This experiment also showed how HEA avoids, with the help of group selection, manually distributing credits to individuals based on their contributions to the team.

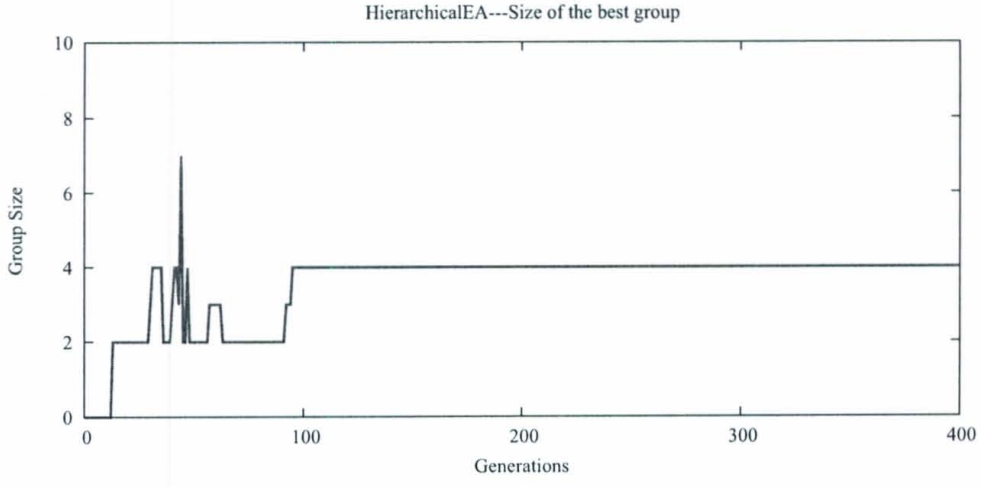
The process of searching for the structure of a solution on the four target sets is shown in Fig. 6.3. We can easily see that the HEA is able to return a solution with the correct number of subcomponents, even though that number was not known *a priori*. Driven by the between-level selection introduced in [3], groups are maintained in the population if they show advantages in fitness; otherwise, they are eliminated. Therefore, we observe the size of the best group changing till the best size is found. We also notice that the group size fluctuates at the beginning of the evolution. This is because individuals during that time have similar coverage and fitness; small changes in group composition and size easily affect the group fitness.

6.4 Chapter Summary

This chapter investigated the emergent decomposition property of our model on string covering problems whose fitness landscapes have multiple equal or unequal fitness peaks. As indicated by the experiments, without *a priori* knowledge, HEA in a bottom-up process always found the solutions with correct number of subcomponents, each covering different patterns hidden in the data sets. That is to say, the structure of a solution and the roles played by their subcomponents emerge as a result of evolution, rather than being designed by hand. When compared with the three control algorithms, especially the CCEA and IEA who decompose problems manually, the solutions produced by HEA are more accurate and require less search time. The next chapter will further study the evolutionary dynamics of HEA, and the ability of HEA to tackle real-world problems that require a substantial degree of cooperation.

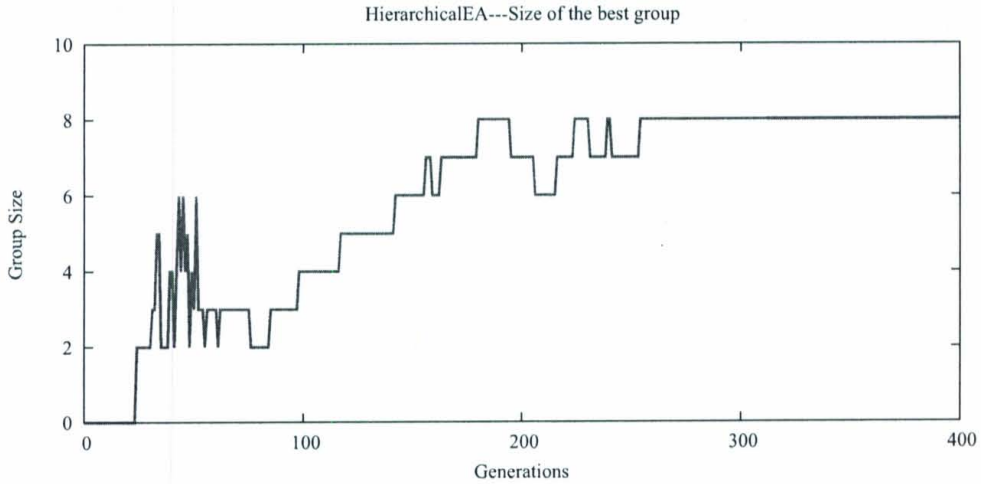


(a) The size of the best group on target set 1

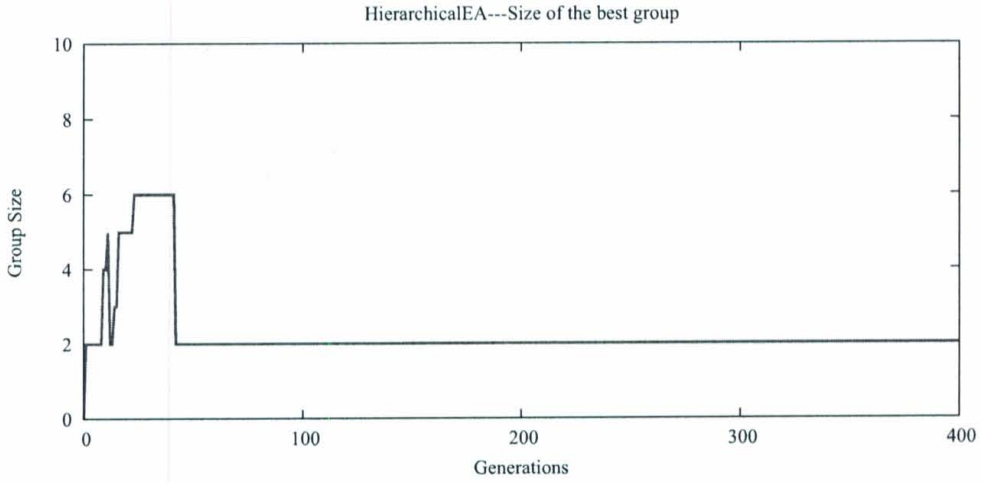


(b) The size of the best group on target set 2

Figure 6.3: Hierarchically finding subcomponents in the solution for all target sets



(c) The size of the best group on target set 3



(d) The size of the best group on target set 4

Figure 6.3: Continued.

Chapter 7

Experiments on Classification Problems

Chapter 5 and Chapter 6 have investigated the cooperation and the emergent problem decomposition properties of our computational multilevel selection model, respectively. Because these investigations were conducted on two toy problems, which are the N-player prisoner's dilemma game and the string covering problem, we were be able to obtain a good understanding of how mechanisms, such as group selection, cooperation operator, and between-level selection, benefit or enhance evolution to encourage cooperation and to achieve automatic problem decomposition. This chapter will continue the effort, but on a more complex problem domain: real-world classification problems. We are interested in the applicability of our model to such practical problems, in which the relationship between subcomponents of a solution is complex and difficult to understand.

In Sect. 7.1, we briefly introduce classification problems and explain the reasons for choosing them as a case study. Sect. 7.2 focuses on the customization of HEA, such as

representations, fitness function definitions and implementation details. Sect. 7.3 describes 7 classification problems with different features, such as non-linearity, skewed data distribution and large feature space. The experimental setup including parameter settings is also listed here. In Sect. 7.4, experiments are undertaken to further understand the role of group selection, and the adaptability of the model in terms of solution accuracy and complexity on datasets with various level of difficulty. The results are compared with outcomes from traditional LGP, one population-based CEA (XCSR), and two team-based CEAs (OET and SBB). The training time of those algorithms is not compared, because the results are either unavailable or incomparable (for example, SBB reduces the size of the training data by sampling.) Sect. 7.5 summarizes the observations derived from these investigations.

7.1 Classification Problems

Classification is probably the most studied data mining task, and possibly the one with the greatest practical relevance [58]. For example, with the help of classification, we may be able to predict who will or will not renew a service contract, or who is or is not a loyal customer when given a related data set. Essentially, classification refers to an algorithmic procedure for assigning a given piece of input data into one of a given number of categories [108]. Each input data sample contains a set of predictor attributes, and a category attribute, which is also known as goal attribute or class label. From the perspective of matching learning, classification is an extension of concept learning; it produces a particular enumeration of patterns (or models), which are a combination of conditions on predictor attributes to describe and distinguish

different values in goal attributes.

We choose classification problems as another application of HEA for the following reasons. First of all, classification has been successfully applied to a wide range of real-world problems; to cite some of them: computer vision, pattern recognition, bioinformatics, natural language processing [108]. The potential benefits of progress in classification are immense since the technique has great impact on other areas, both within data mining and in its applications. Secondly, classification is also a very active research area in Evolutionary Computation (EC) [121]. A classification problem can be formulated as a search problem by considering it as a search for a good pattern in the space of patterns. Evolutionary Algorithms (EAs) can be more powerful when compared with traditional search techniques, because they involve search with a "population" of solutions, not a single solution which might have to backtrack. Last but not least, for most real-world classification problems, especially multiple class classification problems, due to the high volume of data sets and the complicated relationships between predictor attributes and goal attributes, it is impossible to use only one pattern to classify all data instances accurately. A feasible approach is to use multiple but simpler classifiers which co-adapt to balance the detection rate and the false alarm rate of final solutions. Because of this, problem decomposition is impossible to assess prior to runs. In addition, individual classifiers have much stronger correlations than individual matching strings in string covering problems. Therefore, classification problems should better showcase the ability of HEA in modeling the interactions between coadapted patterns and automatic problem decomposition.

7.2 Algorithm Customization

Both classification problems and the string covering problem (Chapter 6) can be trivially expressed as instances of the set covering problem [110], as the ideal solution in both case should be a minimum sized pattern set that covers all input data samples. Therefore, the same workflow described in Algorithm 5 in Chapter 6 can be applied to classification problems. However, classification problems have to satisfy an extra constraint: During the training phase the class label of input data and classifiers should match as well. In order to accommodate this difference, implementation details, such as fitness definition, individual evolution, group evolution, and niching, need to be changed. This section describes these necessary changes.

7.2.1 Representation

In this experiment we use Linear Genetic Programming (LGP) [8] to evolve classification patterns hidden in data sets, such that the patterns are represented by individuals in the format of a linear sequences of C instructions. All instructions apply an operator on one or two registers R_i, or on a constant I_l which refers to the value of attribute l in a data sample; the result is assigned to a destination register R_j; for example, $R_j = R_j + I_l$ or $R_j = R_j \times R_i$. R_0 is defined as the output register, holding the final program output. When R_0 is greater than 0, we say that an input data instance should belong to the class specified by an classifier. In other words, individuals in LGP transform data in a high dimensional space into a specific value or a range of values in a low dimensional space according to different class labels, as demonstrated in Fig. 7.1.

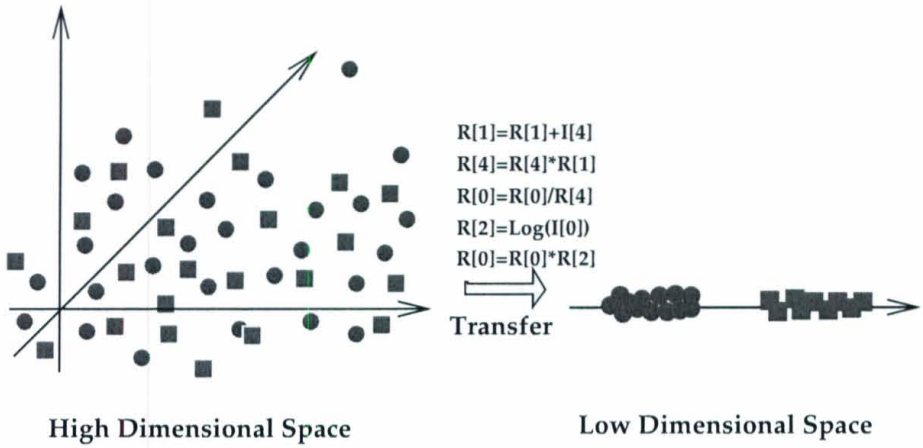


Figure 7.1: Transformation functions as classifiers. A transformation function is an equation or a program which transforms data in a high dimensional space into a specific value or a range of values in a low dimensional space according to different class labels.

Even though this type of representation is not as easy to comprehend as traditional IF-THEN classification rules evolved by GAs or Tree-based GP, it involves more operators and has more flexible structures, both of which will greatly enhance its discriminative power resulting in higher classification accuracy [121]. In addition, as suggested by Brameier and Banzhaf [7], introns, which are non-effective instructions with no influence on the calculation of the output for all possible inputs, are detected and eliminated prior to fitness evaluation. Skipping the execution of non-effective codes, without any doubt, speeds up the evolutionary process.

Following the implementation of SBB [53, 54], the operation set used in this experiment includes 7 arithmetic operators: *addition* (+), *subtraction* (−), *multiplication* (×), *division* (÷), *cosine* (cos), *logarithm* (ln), *exponential* (exp), and 1 conditional

operator *if*, which inverts the value of the first operand if it is smaller than the second operand; for example $if(R[x] < R[y])\ then\ (R[x] = -R[x])$.

7.2.2 Fitness Function

HEA evolves two types of entities: individuals and groups. Individuals in the context of classification are binary classifiers whose chromosome contains a program evolved by Linear GP and a class label. During the training phase, if an individual's program returns a value greater than 0 on a given input data example and the class labels of the individual and an input data instance match, this individual is said to accurately classify the input data; otherwise, this individual misclassifies the input data. The individual fitness is defined as the following:

$$f_i = TPR_i \times (1 - FPR_i)^2 \tag{7.2.1}$$

where TPR_i and FPR_i are the true positive rate (TPR) and false positive rate (FPR) of individual i, respectively. The FPR is given more weight here to encourage low misclassification errors.

Groups are compositions of existing individuals and groups. Again the cooperation operator is constitutive, so no mapping rules are defined. Group fitness is defined as:

$$g_j = \frac{\sum_{i=0}^{n} f_i}{n} \times \frac{C_{covered}}{C_{total}} \times \frac{N_{correct}}{N_{total}} \times \sqrt{\frac{2 + GS_j}{2 \times GS_j}} \tag{7.2.2}$$

where the first term is the average individual fitness of group j, the second term is the class coverage, which is the number of classes covered by group j over the total number of classes in the training dataset, the third term is the data coverage, defined as the number of correctly classified data samples by group j over the total number

of training data samples, and the last term is a normalized term to control the size of group j (GS_j). Here, the first term requires individuals to perform at their best, the second and third terms together rate classification accuracy, and the last term shows another way to control free riders in groups: using the group fitness definition. For example, if group i and group j obtain the same classification accuracy, but group i has a larger group size then group j, we can conclude that group i contains individuals that made no contributions on group levels; those individuals are free-riders. Group i, hence, is penalized with lower fitness by using the last term. Please recall that in the string covering experiments group size was controlled by an extra validation step. Obviously, this fitness function is a measure of how group members collaboratively increase overall data coverage on **all** classes and individually maximize their own classification accuracy with as few members as possible.

7.2.3 Algorithm Description

HEA applied on classification problems follows the same steps shown in Algorithm 5 in Chapter 6. The implementation details are highlighted below.

Initialization The population is initialized with N individuals, each with a unique ID. Class labels from training datasets are randomly assigned to individuals as their class labels.

Evolution on group levels In every generation, up to m new groups are created by cooperation, crossover and mutation with a user-defined probability. Crossover exchanges individuals in two groups; individuals with the same class labels are exchanged with priority; otherwise, arbitrarily selected individuals are exchanged. Mutation adds or removes an individual from a group. The individual being added is

selected from the individual pool (i.e. individuals on the lowest level). The probability of selection is, unless specified otherwise, proportional to fitness. Once a new group is created, its fitness is evaluated. If individuals with the same ID appear in a new group, only one individual is kept.

Evolution on individual levels No more than n offspring with new IDs are reproduced on the individual level every generation. The two-step selection is followed to choose parent individuals. Between-group selection is proportional to fitness. However uniform selection, as opposed to roulette wheel selection suggested by the model, is employed in within-group selection, because individuals all contribute to achieve cooperation despite their fitness. Crossover exchanges randomly selected program segments between two parents, while mutation copies, deletes, adds, swaps, and changes instructions in an individual's program with predefined independent probabilities.

Niching The revised fitness sharing discussed in Sect. 6.2.3 is used here. Due to the special characteristic of classification problems, the similarity of two individuals is measured on their phenotypes; that is, we consider the number of data examples shared between two individuals, as similar individuals will have similar data coverage. If an individual does not have the best fitness in a niche, its fitness has to be reduced by the following equation

$$f(x) = f(x) * \left(1 - \frac{Avg_{\text{overlap}}(x)}{TP(x)}\right) \tag{7.2.3}$$

where Avg_{overlap} is the average data overlap between individual x and others in its niche, and TP is the true positive of individual x, i.e., the number of data samples correctly classified by individual x. Individuals with different class labels have no need to share because there is no overlap. Niching on groups is conducted in a similar way, except that groups only share fitness with ones having the same set of class

labels. This method attempts to save the best groups of various granularity in the population.

Survival selection Survival selection reduces the population to its original size by saving the best n individuals and m groups to the next generation. Subsequently, surviving individuals and groups reset their fitness values to their original values before niching.

7.3 Experimental Setup

The complexity of classification tasks depends on various data characteristics, such as the separability of classes, dimensionality of the feature space, sparseness of available samples, and the number of classes. To examine how HEA performs when encountering complexity, we, following the experimental approach of SBB [53, 54], selected seven datasets from the UCI data repository [29]: The ANN Thyroid Disease (Thyroid), Cleveland Heart Disease (Heart), Statlog Shuttle (Shuttle), Bupa Liver Disorder (Bupa), Pima Indians Diabetes (Pima), Original Breast Cancer Wisconsin (Cancer), and KDD Census Income (Census) datasets. Detailed information about these datasets is summarized in Table 7.1. The first three datasets have at least three classes, while the rest only have two. Shuttle, Thyroid and Census also have unbalanced class distributions, where the data distribution for minor classes is as low as less than 0.01%. This property is ideal to demonstrate how group selection can help maintain individuals for both, minor and major classes. Bupa and Pima are two datasets known for poor class separability; a rate of error in the region of 30% has been observed across a wide range of machine learning algorithms [54]. Other

Table 7.1: Summary of the datasets used in the evaluation. '*' indicates a dataset has no separate test set; therefore we divided the dataset into partitions of 90% for training and 10% for test. The value in parentheses following the name of the dataset indicates the number of features.

Type	Dataset		Data Distribution							
			Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Total
Multi-class	Thyroid (21)	Training	93	191	3488	-	-	-	-	3772
		Test	73	177	3178	-	-	-	-	3428
	Heart (13)*	Training	148	50	33	32	12	-	-	275
		Test	16	5	3	3	1	-	-	28
	Shuttle (9)	Training	34108	37	132	6748	2458	6	11	43500
		Test	11478	13	39	2155	809	4	2	14500
Two-class	Bupa (6)*	Training	181	131	-	-	-	-	-	312
		Test	19	14	-	-	-	-	-	33
	Pima (8)*	Training	451	242	-	-	-	-	-	693
		Test	49	26	-	-	-	-	-	75
	Cancer (10)*	Training	413	217	-	-	-	-	-	630
		Test	45	24	-	-	-	-	-	69
	Census (41)	Training	187141	12382	-	-	-	-	-	199523
		Test	93576	6186	-	-	-	-	-	99762

reasons to select those datasets are that they have established performance levels [53, 54, 96, 115] in other CEAs, and they all represent real world data rather than artificial data. Therefore, the results obtained on those datasets should make us sufficiently confident to judge whether or not the new computational multilevel selection framework is applicable to solving real-world problems.

Before starting, some additional data pre-processing is performed. We first map categorical values into numeric values by using the order in which they appear. Missing values are replaced by the value of the nearest data sample (measured by euclidean distance) for the relevant attribute.

50 runs were performed on each dataset. 10-fold cross-validation was used to assess datasets denoted by '*' in Table 7.1; that is, five runs per partition. This helps to minimize validation errors when no separate test dataset is provided, and makes sure that the performance comparison is fair between our algorithm and control algorithms. Introns in individual programs were removed by the method described in [6]. 8 registers were used and initialized by the mean value of a randomly selected input attribute. JAVA parallel programming dispatched individual fitness evaluation and niching to 15 threads running on 15 CPUs.

Parameters shared by all experiments are shown in Table 7.2. Parameters specific to each dataset were as follows: the maximum number of generations in runs for Thyroid and Cancer is $2,000$, and $10,000$ for the rest. The population contains 30 individuals and 20 groups for Cancer, 60 individuals and 20 groups for Thyroid, Bupa, Pima and Census, and 140 individuals and 30 groups for Heart and Shuttle.

Two indicators were employed to measure the performance of HEA on classification tasks: overall detection rate (ODR) and average detection rate (ADR) [53]. ODR is

Table 7.2: Parameterization of multi-class classification problems. $ProgramSize_{max}$ refers to the maximum program length, $P_{coopgp}, P_{xovergp}, P_{mutgp}$ for group cooperation, crossover, and mutation probability, $P_{xoveridv}$ for individual crossover, $P_{del}, P_{add}, P_{mut}, P_{swap}, P_{copy}$ for deleting, adding, changing, swapping, and copying instructions at individual mutation, and R_{gp}, R_{idv} for group and individual niching radius.

Parameters	Value	Parameters	Value
$ProgramSize_{max}$	48	P_{coopgp}	0.5
$P_{xovergp}$	0.8	P_{mutgp}	0.3
$P_{xoveridv}$	0.8	P_{del}, P_{add}	0.6
P_{mut}, P_{swap}	0.6	P_{copy}	1
R_{gp}	0.5	R_{idv}	0.4

defined as the number of data samples the final solution correctly classified over the total number of test data; ADR is defined as the average detection rate on all classes. ADR is independent of class distribution; hence it is a good supplement to ODR, especially for datasets with unbalanced data distribution. Take the Thyroid dataset as an example: Only 2% of the test data are from class 1. Even if a final solution missed all data samples in class 1, its ODR could still reach as high as 98%.

7.4 Evaluation

This section presents the experiments for highlighting the effects of group selection on HEA and comparing the performance of HEA with the traditional LGP, XCSR, OET, and SBB algorithms. The results of the control algorithms, presented in the format of box plots/violin plots, were gathered from [53, 54, 96, 115]. A violin plot is a combination of a box plot and a kernel density plot which shows the probability density of the data at different values. Box plots allow us to compare two result sets without knowing their underlying statistical distributions. They even can verify the statistical significance of differences between the result sets; if the notches of two boxes do not overlap, the median of the two datasets differ at the 0.95 confidence interval. The detection accuracy mentioned in [53, 96, 115], and the multi-class detection rate or the score in [53, 54] are equivalent to ODRs and ADRs in our experiments, respectively.

7.4.1 Understanding Group Selection

One of the key concepts of HEA is to associate the survival of individuals to the performance of their group. This encourages the emergence of cooperative groups,

because only through cooperation will individuals seize the opportunity to reproduce offspring. To examine if this key point plays the same role in computational settings, we compared HEA with a control algorithm, called CtrlHEA, which functions the same way as HEA, except that parent individuals are selected directly from the individual pool, rather than from groups. That is to say the CtrlHEA does not consider group selection at the reproduction stage.

We run the two algorithms 50 times on the Thyroid dataset, which has 3 classes with an imbalanced data distribution. The mean classification accuracies and average class coverage (CLS) are shown in Table 7.3. The ADR values clearly show the per-

Table 7.3: The average classification accuracies and class coverage of HEA and CtrlHEA on the Thyroid dataset over 50 runs. Standard deviations are listed inside of parentheses.

	ODR	ADRs	CLS
HEA	0.978 (0.008)	0.954 (0.031)	3 (0.0)
CtrlHEA	0.931 (0.025)	0.595 (0.070)	2.02 (0.141)

formance difference between the two algorithms. The low ADR obtained by CtrlHEA implies that it is not able to cover all classes; on average it covers 2.02 classes out of 3. In fact, CtrlHEA rarely includes a classifier in groups to cover data examples from class 1, the minority class. Our further investigation shows that individuals evolved for class 1 normally start with a low TPR and high FPR, i.e. a relatively low individual fitness, because of very scarce training data. Therefore, if competing against individuals who classify data for major classes in the same population, they

are less favored given fitness proportional selection. In other words, the growth of fitness progresses very slowly on such individuals. As a result, the chances of them existing in the best group are slim.

However, if we allow group selection at the reproduction stage, the outcome is different. Individuals are randomly selected from a group; individuals, despite their fitness, are facing equal reproduction opportunities. Because individuals evolved for minority classes can provide additional data coverage, their appearance in a group will improve group fitness, which in turn would increase their probability of being selected and reproduced. Consequently, the fitness of weak individuals that possess unique contributions is improved much quicker in a group than in a population. This experiment also demonstrates the importance of individual optimization. Only when the space of individuals has been properly explored, will it be possible to build a good solution from them.

7.4.2 Classification Accuracy

We first evaluate the performance of HEA on the four two-class datasets. Average classification accuracies and class coverage of the best groups collected from 50 runs are summarized in Table 7.4. The violin plots of average ODRs and ADRs are shown in Fig. 7.2.

It is evident from the 4th column of Table 7.4 that the best groups evolved by HEA successfully covered both classes, even the minority class (class 2) in Census dataset. XCSR, on the contrary, indiscriminately labeled almost all instances in class 2 to class 1, resulting a low ADR (around 0.504) [53].

We first compare the box plots of ODRs obtained by HEA and traditional LGP

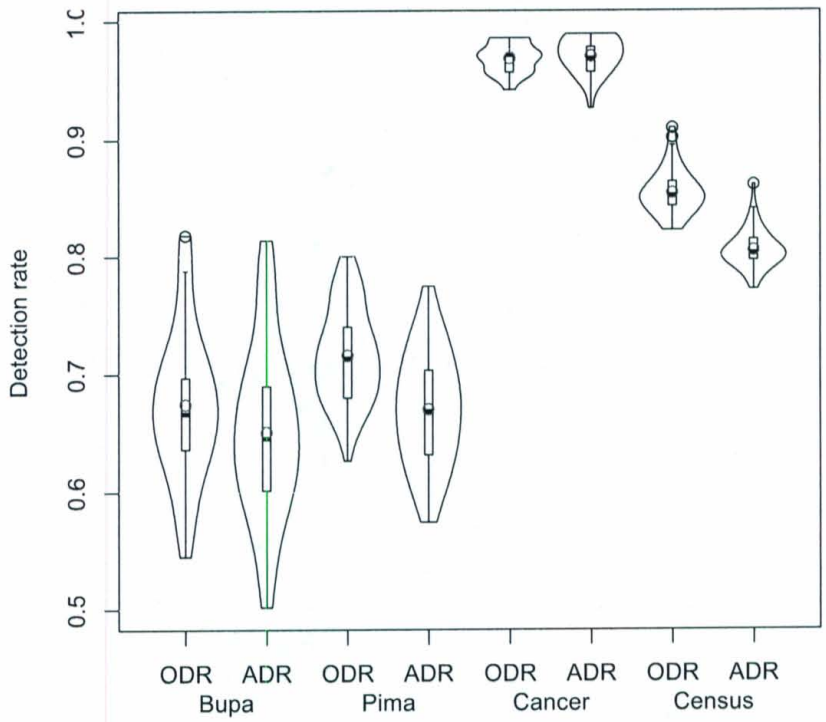


Figure 7.2: Violin plots of ODRs and ADRs obtained by HEA on the four two-class datasets over 50 runs. Each box indicates the lower quartile, median, and upper quartile. The horizontal lines at the end of whiskers represent the maximum/minimum values. Points outside of the boxes represent outliers to whiskers of 1.5 times interquartile range, and points inside of the boxes show the mean values of ODRs or ADRs.

Table 7.4: The average classification accuracies and class coverage of HEA on the four two-class datasets over 50 runs. Standard deviations are listed inside of parentheses.

Dataset	ODR	ADR	CLS
Bupa	0.675 (0.063)	0.651 (0.072)	2 (0.0)
Pima	0.716 (0.040)	0.670 (0.049)	2 (0.0)
Cancer	0.968 (0.013)	0.970 (0.015)	2 (0.0)
Census	0.854 (0.019)	0.805 (0.016)	2 (0.0)

on Cancer dataset (See Fig. 7.2 and Fig. 7.3 for details). The minimum ODR in HEA is larger than the upper quartile (UQ) value in traditional LGP, which implies the notches of the two boxes are impossible to overlap. We can conclude that the HEA outperforms traditional LGP on this dataset at the 0.95 confidence interval. In the case of the other datasets, we compare box plots of ADRs produced by HEA and SBB (See Fig. 7.2 and Fig. 7.4 for details).

On the Census dataset, HEA outperforms SBB at the 0.95 confidence interval, given the fact that the two boxes do not overlap. On the Bupa dataset, HEA and SBB have the same maximum and UQ ADRs, but HEA has higher minimum, lower quartile (LQ), and median values. On the Pima dataset, HEA has higher values on all statistics except the maximum value. That is to say on both datasets SBB's graph is generally lower than HEA's graph; in addition, the ADRs in SBB have larger variability than HEA because of a longer interquartile range. Overall, it appears that HEA performs better and more stable than SBB. However, because that the boxes overlap and the notches are not shown, the significance of the differences cannot be

checked.

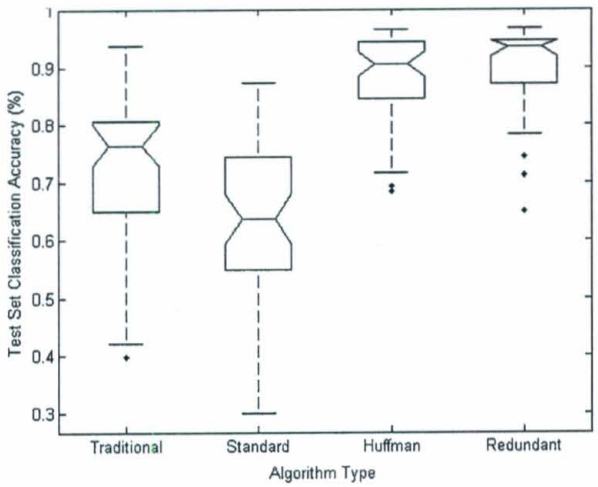


Figure 7.3: The box plot of ODR obtained by the traditional LGP on the Cancer dataset (denoted as "traditional"). From "Introducing Probabilistic Adaptive Mapping Developmental Genetic Programming with Redundant Mappings," by G. Wilson and M. Heywood, Genetic Programming and Evolvable Machines, 8(2):187-220, 2007. Reprinted With Permission.

We then increase the difficulty of the problem by feeding HEA three more datasets with multiple classes. The results are detailed in Table 7.5 and plotted in Fig. 7.5. All three datasets have skewed data distributions, especially Shuttle in which class 5 only has 6 out of a total 43,500 data examples. However, such skewness apparently affects XCSR most; the low ADR value implies that XSCR is not able to detect data samples of the rare classes. In contrast, HEA and SBB have comparable values on ODRs and ADRs. In fact, the class coverage of HEA shown in Table 7.5 clearly indicates that best groups evolved by HEA identified data from all classes.

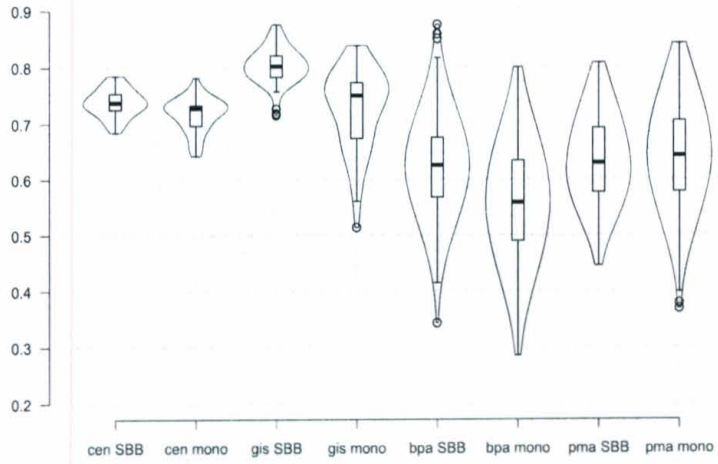


Figure 7.4: Violin plots of ADR obtained by SBB on the Census (denoted as "cen SBB"), Bupa (denoted as "bpa SBB"), and Pima (denoted as "pma SBB") datasets. From "Symbiosis, Complexification and Simplicity under GP," by P. Lichodzijewski and M. I. Heywood, In M. Pelikan and et al., editors, GECCO '10: Proceedings of the 12th Genetic and Evolutionary Computation Conference, pages 853-860, ACM, 2010. Reprinted With Permission.

As we expected, the performance of HEA on the Heart dataset is better than traditional LGP at the 0.95 confidence interval if we compare their plots in Fig. 7.5 and Fig. 7.6. HEA also performs better than OET (see Table 7.5) on this dataset.

Table 7.5: The average classification accuracies and class coverage of HEA on the three multi-class datasets over 50 runs. Standard deviations are listed inside of parentheses. Results shown for SBB and XCSR are cited from [53], and OET from [96]. The best values from the three approaches are shown in bold.

Dataset		ODR	ADR	CLS
Thyroid	HEA	**0.978** (0.009)	**0.950** (0.041)	3 (0.0)
	SBB	0.960	0.935	
	XCSR	0.976	0.924	
Shuttle	HEA	**0.999** (0.001)	**0.983** (0.020)	7 (0.0)
	SBB	0.967	0.953	
	XCSR	0.982	0.416	
Heart	HEA	**0.744** (0.043)	0.688 (0.072)	5 (0.0)
	OET	0.568 (0.030)		

For the Thyroid and Shuttle datasets, the ODRs and ADRs produced by XCSR and SBB approaches were collected from [53]. We listed them in Table 7.5 as reference for comparison. Clearly HEA outperforms SBB and XCSR on both datasets with respect to either ODRs or ADRs. To find out if the differences are statistically significant, we then compare their box plots. The box plots of SBB and XCSR (see Fig. 7.7) were drawn using the normalized ODR and ADR values. For fair com-

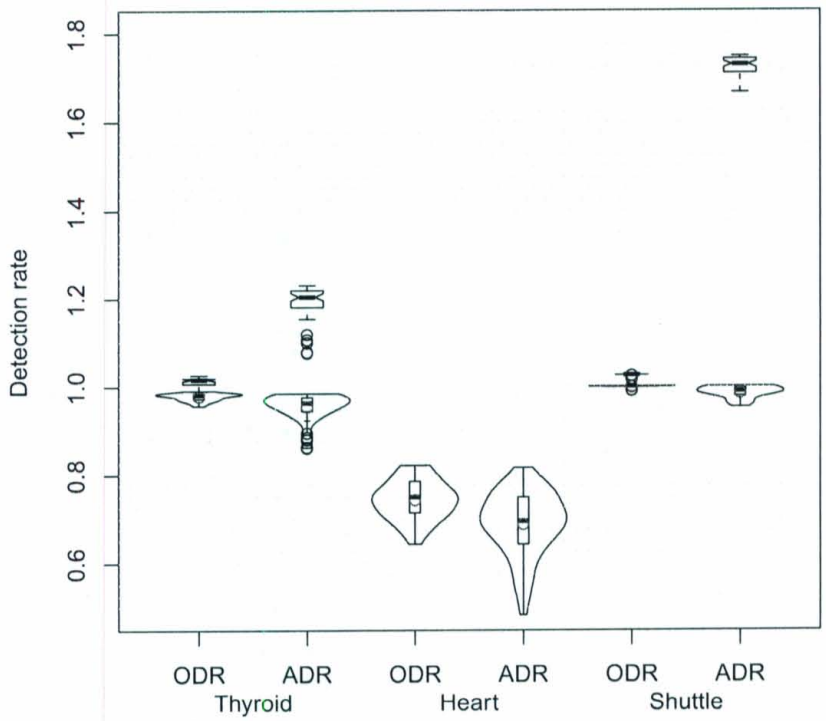


Figure 7.5: Violin plots of ODRs and ADRs of the best groups obtained by HEA on the three multi-class datasets over 50 runs. The box plots depict the distribution of normalized ODRs and ADRs on the Thyroid and Shuttle datasets.

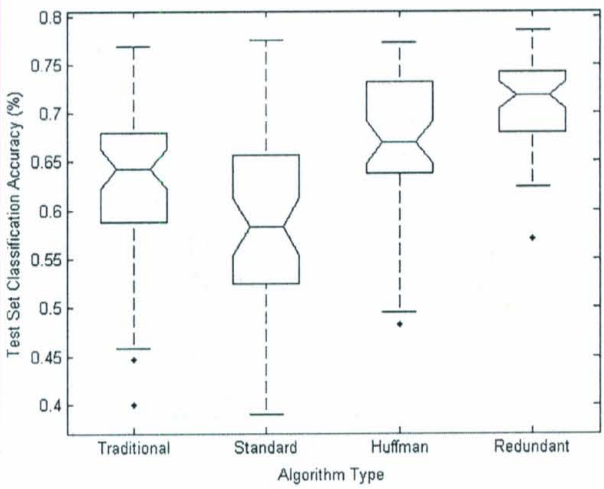
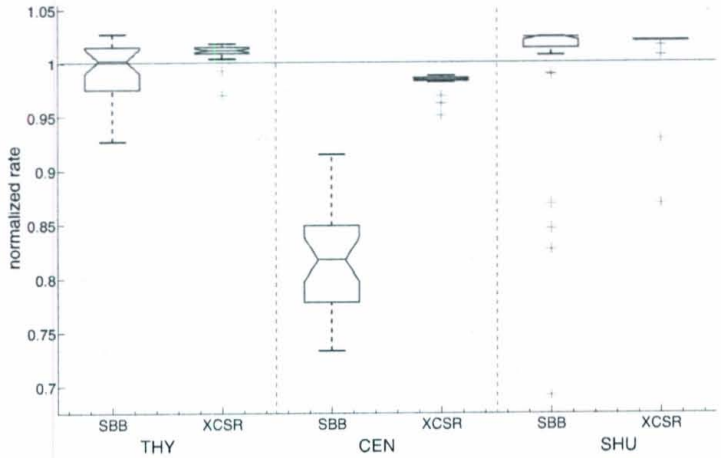
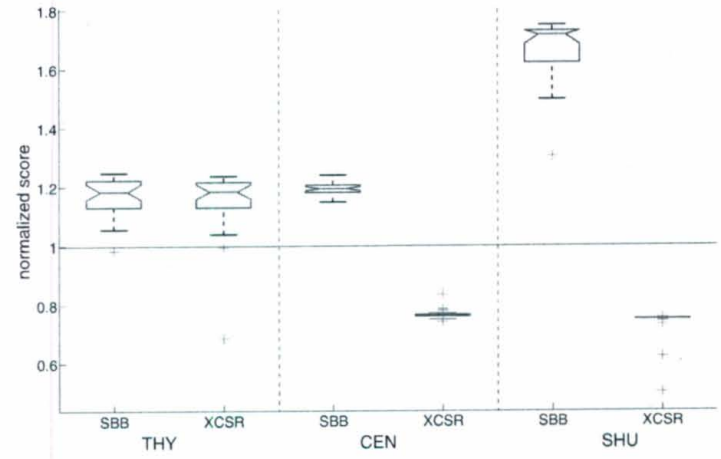


Figure 7.6: The box plot of ODR obtained by the traditional LGP on the Heart dataset (denoted as "traditional"). From "Introducing Probabilistic Adaptive Mapping Developmental Genetic Programming with Redundant Mappings," by G. Wilson and M. Heywood, Genetic Programming and Evolvable Machines, 8(2):187-220, 2007. Reprinted With Permission.



(a) Normalized ODR.



(b) Normalized ADR.

Figure 7.7: Box plots of normalized ODR and ADR obtained by SBB and XCSR on the Thyroid (denoted as "THY") and Shuttle (denoted as "SHU") datasets. From "Managing team-based problem solving with symbiotic bid-based genetic programming," by P. Lichodzijewski and M. Heywood, In C. Ryan and M. Keijzer, editors, GECCO '08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation, pages 363-370, ACM, 2008. Reprinted With Permission.

parison, the same normalization procedure (see [53] for details) was applied to ODR and ADR values in HEA. Box plots based on the transformed values are depicted in Fig. 7.5. HEA outperforms SBB and XCSR on Shuttle at the 0.95 confidence interval on both ODR and ADR values. However, with respect to the Thyroid dataset, all three box plots on ADRs have very close maximum values, but again HEA has the highest median, and the shortest interquartile range; for example the LQ value of HEA is aligned with the median of SBB and XCSR. Similar patterns are observed on the ODR box plots as well. Because the HEA results are highly clustered, it is difficult to tell whether their box plots overlap or not. However, we can safely conclude that HEA performs at least as good as SBB and XCSR on Thyroid.

In conclusion, HEA, in terms of classification accuracies, outperforms SBB on Census and Shuttle at the 0.95 confidence interval, and performs slightly better than or at least as good as SBB on the Bupa, Pima, and Thyroid. It excels SBB in stability (i.e, low variance of the distribution of classification accuracy) on all datasets. One of the reasons that SBB has a diverse distribution over accuracies may be that a uniform probability selection scheme is used for within-group and between-group selection. Uniform probability selection does not distinguish individuals and groups based on their performance (fitness). Therefore, the optimization opportunities are spread over all individuals and groups. HEA performs better than XCSR on skewed datasets, such as Thyroid, Shuttle, and Census, because XCSR, as we stated before, lacks a measurement of group performance. HEA also exceeds single binary classifiers evolved by the traditional LGP on performance, because binary classifiers only focus on one class at a time, and ignore correlations with other classes.

7.4.3 Solution Complexity

HEA builds solutions hierarchically out of simple subcomponents without the need to specify in advance their structure. We are interested to know how complex solutions are, especially when compared to solutions returned by SBB. In this investigation, we use group size to represent solution complexity. More sophisticated measurements, such as the number of unique attributes utilized by an individual and the number of effective instructions per individual [54], will be left for future work.

Figure 7.8 plots the average number of individuals in the best groups from 50 runs on the four two-class datasets. The solution complexity of SBB obtained on

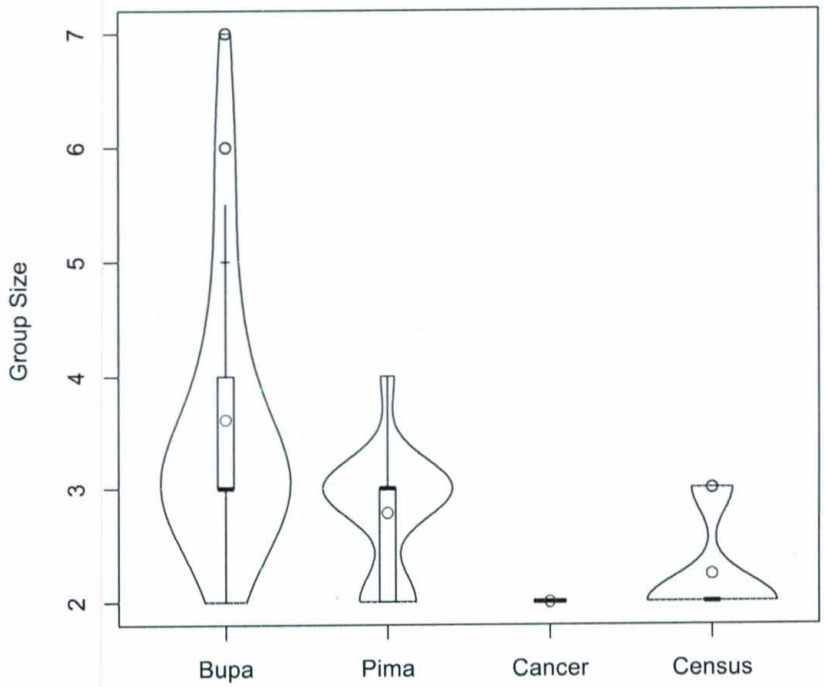


Figure 7.8: Solution complexity of best groups obtained by HEA on the four two-class datasets over 50 runs.

Bupa, Pima, and Census can be found at Fig. 7.9. HEA has the same solution

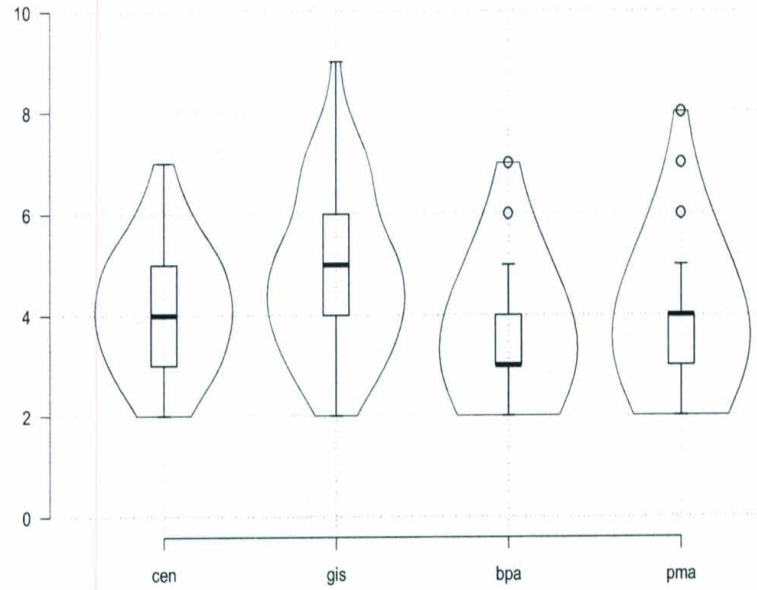


Figure 7.9: Solution complexity of the best groups obtained by SBB on the Bupa (denoted as "bpa"), Pima (denoted as "pma") and Census (denoted as "cen") datasets. From "Symbiosis, Complexification and Simplicity under GP," by P. Lichodzijewski and M. I. Heywood, In M. Pelikan and et al., editors, GECCO '10: Proceedings of the 12th Genetic and Evolutionary Computation Conference, pages 853-860, ACM, 2010. Reprinted With Permission.

complexity as SBB on Bupa. However, for the other two datasets, SBB tends to find more complicated solutions with larger numbers of individuals than HEA (at 0.95 confidence interval); for example, the median values obtained by SBB on Pima and Census are 4, while they are 3 and 2 for HEA, respectively.

On Thyroid and Shuttle, solution complexity of HEA is significantly lower than

SBB or XCSR (see Table 7.6). It found the most compact groups on all runs, in which only one individual is used to classify every single class.

Table 7.6: The average solution complexity of HEA, SBB and XCSR on the three multi-class datasets over 50 runs. Standard deviations are listed inside of parentheses. Results shown for SBB and XCSR are cited from [53]. The best values among the three approaches are shown in bold.

	HEA	SBB	XCSR
Thyroid	**3**(0.0)	9.5(0.9)	881.2(14.3)
Shuttle	**7**(0.0)	10.0(0)	644.8(39.4)
Heart	6.667(1.361)		

We thus can conclude HEA beats SBB in terms of solution complexity. The obvious reason is that HEA explicitly expresses how to control group size in a group fitness function. It is particular noteworthy that HEA automatically keeps the solution complexity in proportion to the separability of a dataset. For highly separable datasets, HEA returns the smallest group with each member being responsible for one class, without wasting extra computational resources. For poorly separable datasets such as Heart, Bupa and Pima, however, HEA tends to evolve large groups in which one data class is covered by more than one individual. These results clearly demonstrate the good problem decomposition ability of HEA; the appropriate number of subcomponents and their roles emerge through evolution without human interference. The driving evolutionary force behind this effect is the between-level selection, which controls the hierarchical structure by screening out invalid levels and groups.

7.5 Discussion

So far we have demonstrated how to implement the new computational multilevel selection framework using LGP to solve classification problems. Please also recall the experiments conducted on string covering problems in Chapter 6. The findings of these two studies confirm that HEA is able to improve solution accuracy and simplify solution complexity as compared to other approaches in the literature. However, the following issues should be given special consideration before HEA is applied to new problems:

1. Evolutionary Transition. As shown in Chapter 5, our model has the potential to be extended to an evolutionary transition model, in which groups, depending on their levels, become a new complex organism functioning differently from their components. Even though not demonstrated in these two studies, we believe our model will be useful to solve real-world problems whose subcomponents have more complicated interactions, such as agents in multi-agent systems. Detailed transition rules can be defined to change the genotype or phenotype of a new organism, thus expressing various functions.

2. Niching. With no exception, our model requires the use of niching or similar techniques to maintain different partial solutions in a population, from which a full solution can be built. Designing an appropriate niching scheme, nevertheless, can be very tricky as it is strongly correlated with specific problems. Canonical fitness sharing, resource sharing or crowding are always good starting points. However, one important thing to remember about fitness sharing is that it diminishes the fitnesses of all individuals within a niching radius, including

the best one in the niche. We are then faced with the risk of losing potentially good individuals; if they are closely surrounded by others, their fitness may degrade much faster than less optimal individuals with no neighbors.

3. Group fitness definition. After multiple trials on different group fitness definitions, we advise to consider at least two factors: average individual performance and overall group performance. Missing either of them will cause the evolution to drift to suboptimal solutions.

4. Cooperation measurement. Evidence in biology and social science suggests that excluding or punishing free-riders can maintain cooperation. In the same way any implementation of our model should measure how much individuals cooperate in a group. Removing free-riders yields compact groups and savings on computing resources. In Chapter 6, an individual's contribution is judged by the number of new strings it provided to its group. In this chapter, an individual's contribution was indirectly assessed jointly by the group size and overall data coverage in the group fitness function; free riders increase group size without improving coverage. The Shapley value [87] of game theory is also an interesting approach to determine the contributions of individuals in a collaboration.

5. Parameterization. The framework extends evolution to group levels; therefore, we need to specify values for new parameters, namely the cooperation, crossover and mutation rates for reproducing groups, the niching radius for groups and individuals, and the number of groups in a population. Like any other EA, there are no universally optimal parameter settings that suit every problem. Based on our experiments, we suggest high cooperation and crossover rates, but a

relatively low mutation rate, as cooperation constructs new groups and crossover discovers all possible individual combinations. The number of individuals and groups in a population will vary depending on specific problems. Complex problems normally need a large individual pool in order to preserve all potential subcomponents. A group pool is normally smaller than an individual pool, and its size increases as the individual pool grows, but at a smaller rate.

7.6 Chapter Summary

In this chapter, we moved the investigation of our computational multilevel selection model to solve more practical and complex problems: real-world data classification. Such problems are complex because after decomposition the interdependencies between subproblems are difficult to understand. The Hierarchical Evolutionary Algorithm was applied on 7 classification tasks, whose datasets reflect different features, such as non-linearity, skewed data distributions, and a large feature space. The results, when compared to traditional GP, OET, XCSR and SBB, demonstrate that this approach improves solution accuracy and consistency, and simplifies solution complexity. In particular, HEA automatically keeps the solution complexity in proportion to the difficulty of the datasets. For highly separable datasets, HEA returns the smallest group with each member being responsible for one class; However, for poorly separable datasets, HEA tends to evolve larger groups in which one class is covered by more than one individual. This observation clearly demonstrates the good problem decomposition ability of our model. In addition, this chapter also shows that our model can be easily adapted to different classes of evolutionary algorithms, and different

application domains.

Chapter 8

Conclusion and Future Work

In this chapter we will summarize our work and the contributions made to the evolutionary computation and artificial life communities. We will also give an outline of future work that could be derived from this dissertation.

8.1 Summary

Evolution, driven by the force of natural selection, demonstrates an optimization characteristic. Without exception, Evolutionary Computation (EC), which mimics natural evolution, also inherits this character and hence is applied widely to solve optimization problems. However, EC may fail to solve decomposable problems, whose solution are in the form of multiple coadapted subcomponents; in other words, because of its strong tendency to converge, EC is not suitable for evolving a set of individuals that work cooperatively.

Surprisingly enough, despite this seeming conflict between evolution and cooperation, cooperation has been observed everywhere in our hierarchically organized

biological world. For example, genes cooperate in genomes, chromosomes in cells, and cells in multicellular organisms. The reason is that cooperation is needed for evolution to construct new levels of organizations [70]. Through cooperating in these organizations, the constituents can increase their chances of survival.

Biologists have proposed several theories to explain the evolution of cooperation, including kin selection, reciprocation, group selection and social learning. Among these, group selection has been embraced by a growing number of biologists, in spite of longstanding controversy. In fact, group selection unifies kin selection and reciprocation [73, 112]; it is also compatible with the selfish-gene theory. Group selection theory suggests that individuals are divided into groups, and the emergence of cooperation is due to the selection pressure exerted on groups: between-group competition facilitates within-group cooperation. Therefore, it sheds light on integrating cooperation into artificial evolution. The primary aim of this dissertation was to extend classic artificial evolutionary models to multilevel hierarchies, so that the principles of group selection theory could be applied on each level to allow cooperation to emerge and be sustained.

Most multilevel selection models in the literature take the hierarchical structure as given. The biological hierarchy, on the contrary, has developed gradually: simpler, smaller components appeared before more complex, composite systems. Therefore, the new computational multilevel selection model we propose defines a bottom-up process, where entities on new levels are created with the help of the cooperation operator in the framework of predefined reaction rules. Hence, new entities will possess new traits due to their genotypic or phenotypic differences. Evolution is performed on each level to optimize the traits of the entities on that level. Selection pressure from

higher levels forces individuals on lower levels to cooperate. The between-level selection determines which level to select and controls the growth of hierarchical structure. As a result of these features, the model shows an emergent property: the appropriate structure required to reach a predefined cooperation goal, i.e., the number of individuals and the role each individual plays in the cooperation, will be automatically developed during evolution. We believe this model evolves faster and performs better than other current proposals in Cooperative Evolutionary Algorithms.

The intention of the model is twofold. First, the model can be used by computer scientists and engineers to solve real-world cooperative problems. To this end, we first presented a hierarchical evolutionary algorithm that implemented the model we proposed. We then validated the cooperation and problem decomposition capability of this algorithm within the context of string covering problems. Finally, we applied the algorithm for Multi-Class Classification (MCC). When compared to string covering problems, MCC is much more complicated, as the number of classifiers in a desired solution is unknown and it is very difficult to understand the interdependencies among those individuals. This real-world application is a better showcase of the emergent problem decomposition and cooperation properties of our model. The experiments conducted on both problems demonstrated that our model evolves faster to find more accurate solutions than other cooperative evolutionary algorithms.

Second, the model can be used by researchers in artificial life to study the evolution of cooperation and related issues. As a step towards this goal, we confirmed by experiments the feasibility of this model to evolve cooperation. Our findings revealed that cooperation emerges and persists more easily in our model than in Wilson's or Trauslen's models. The reason is that different mechanisms were employed to enhance

the effect of group selection, mainly the bottom-up process and the cooperation operator. In addition, multilevel selection also provides explanations for evolutionary transitions. We hence studied division of labor using our multilevel selection model; division of labor is a commonly observed group trait resulting from an evolutionary transition. As demonstrated by the experiments, groups with all required skills transit successfully from a population of independent individuals, no matter whether skills are equally rewarded or not. Our experiments also confirmed that both type 1 and type 2 multilevel selection are relevant to evolutionary transitions.

8.2 Contributions

Through discussions and experiments, our comprehension of multilevel selection theory, especially its working mechanisms and its role in promoting cooperation, developing transitions and building up hierarchies has deepened. We claim the following conceptual and practical contributions, hoping that our findings and understandings are some help to those also interested in studying, modeling, and designing computational multilevel selection models.

8.2.1 Conceptual Contributions

- Clarified the concept of group selection. During the literature review, we noticed that some research work mistakenly equates the idea of group selection with the idea of selection between groups. In fact, group selection incorporates not only between-group selection, but also a two-step selection procedure at individual reproduction: a group is selected first, from which an individual is

then selected for reproduction. Associating the survival of an individual with its group propagates cooperators within groups, and eliminates the need for credit assignment required by Cooperative Evolutionary Algorithms.

- Suggested to consider evolution on every level of the hierarchical structure. When evolution is conducted on each level, it means that the fitness of collaborations is defined to look after the interdependences between the constituents, selection is applied on each level to encourage entities below to cooperate, and adaptation is developed on every level in response to dynamic environmental change.

- Confirmed between-group selection as an unignorable force in computational settings with respect to promoting cooperation. Such selection models the coadaptation and interaction between individuals. The resulting selection pressure also forces individuals in cooperation to develop different roles when necessary, and mediates the conflict of interest between individuals and their collaboration.

- Added to the mechanisms to create hierarchical structures. The cooperation operator is a means of forming groups, such as cells sticking together to form multicellular organisms. Mapping rules state under what conditions which actions must be taken; this includes triggering conditions, entities before mapping, and entities after mapping. The transformed entities are genotypically and phenotypically more complex than entities before mapping, and they become the entities on a new, higher level.

- Showed the integration of two types of multilevel selection in one multilevel selection model. The experimental results confirmed the prediction of Okasha

[75, 78] on the relevance of both types of selection in evolutionary transitions.

- Identified critical issues that every cooperative evolutionary algorithm must address: problem decomposition, evolution on multiple levels, and diversity preservation. This is consistent with the issues suggested by Potter and de Jong [81], but with an extension made to the evolution on multiple levels.

8.2.2 Practical Contributions

- Proposed a computational multilevel selection model. The core element of this model is the computational implementation of multilevel selection theory. This model also attempts to capture key mechanisms employed by nature to create hierarchical structures. The two features together describe a process in the model which is analogous to the process of constructing sophisticated solutions out of simpler ones. Therefore, with proper adaptations, this model is useful for computer scientists and engineers to solve decomposable problems in different domains. In fact, this model overcome the limitations of existing cooperative evolutionary computation models. Researchers in artificial life could use this model to better understand the nature of multilevel selection and to investigate implications of multilevel selection. One of the implication is the evolution of cooperation. Experiments can be designed on this model to simulate the evolution of cooperation by multilevel selection under various conditions, such as different population structures, interaction constraint, or population composition. Cooperation is the first step to achieve evolutionary transitions, which further leads to the diversification of life and the hierarchical organization of the living world. Therefore, this bottom-up hierarchical model can also be used to study evolu-

tionary transitions, which is another implication of multilevel selection, and the creation of hierarchical structures.

- Designed a hierarchical evolutionary algorithm based on the new model. This algorithm is targeted to solve problems whose solution is in the form of multiple coadapted subcomponents. When compared to other cooperative evolutionary algorithms in the literature, this algorithm adequately addresses the issues of problem decomposition, evolution on collaboration levels and diversity preservation. Consequently, it evolves faster and returns more compact, accurate results than others. Since this algorithm describes a general approach for evolving cooperation by an evolutionary algorithm, it can be applied to a variety of domains and is not limited to any particular implementation of evolutionary algorithms. As shown in this dissertation, both genetic algorithms and genetic programming can be used to instantiate this algorithm.

8.3 Future Work

This thesis leads to a number of opportunities for future research. The following are possible areas for further investigation that could prove profitable to computer science and engineering and also researchers in artificial life:

8.3.1 Computer Science and Engineering

- Heterogeneous representations. Our model evolves individuals in one population, which implies all individuals have the same representation: they use the same chromosome structure and accept the same input information. This re-

quirement becomes a restriction on the model when subcomponents of a solution need to be represented differently. Therefore, we would like to explore a remedy for this limitation, but at the same time without sacrificing problem decomposition as an emergent property.

- Diversity preservation. As emphasized many times, diversity preservation is critical to the success of the Hierarchical Evolutionary Algorithm (HEA). This dissertation adopted a revised fitness sharing for maintaining different partial solutions in a population. However, this niching strategy still requires the definition of a niching radius and a fitness adjustment equation, both of which are decided by a trial and error process. This is a limitation that prohibits ones from applying HEA to solve problems in other domains. More investigations should be conducted on diversity preservation strategies and also on the dynamics caused by each strategy.

- Applications for multi-agent systems. Another possible application of the hierarchical evolutionary algorithm is to evolve cooperative behavior for multiple agents so that they could work as a team. Agents are autonomous and intelligent: they operate without central control, and are able to interact with their surrounding environment to decide their next move. Therefore, the interdependencies among agents are harder to model. Furthermore, quite commonly in many multi-agent systems the fitness of agents cannot be implicitly defined. Hence, the algorithm needs the ability to deal with endogenous fitness, which emerges from actions and interactions over the course of an agent's lifetime.

- Applications on problems with sophisticated solution structures. One advantage

of this model is the flexibility in defining group structure by using reaction rules, as shown in the division of labor example. However, no reaction rules were defined in the applications of the algorithm to string covering problems and multi-class classification problems; solutions for both problems are a simple combination of individuals. It would be interesting to find an application domain in which mapping rules are required to bridge the structural difference between subcomponents and the final solution, and to test the strengths and weaknesses of the algorithm on that domain.

8.3.2 Artificial Life

- Population structure. Our model treats populations as well mixed and unstructured. Individuals are dispatched into groups, in which the interactions between individuals take place randomly. However, real populations are not well mixed. Spatial topology or social networks imply that some individuals interact more frequently than others [70]. It would be interesting to find out whether or not the multilevel selection theory could promote cooperation on structured population. One challenge left to face is how to define the group boundary. The SkillWorld in P2P networks [39] is a good potential problem to test this issue.

- Time scale. Our model avoids on purpose the discussion of the time scale problem concerning the evolution taking place at each level of the hierarchy. As a matter of fact, entities on different levels evolve at different rates. Salthe [85] pointed out that the higher one goes from level to level, the longer it takes for the process to continue, or cycle, or go to completion when viewed from a fixed scale. The difference in the rates of processes are one of the fundamental

sources of hierarchical structure in nature [1]. "Our" or "an" artificial hierarchy should consider the impact of time scale as part of the model.

- Evolution of individuality. The evolution of individuality [9] is a different research topic from the evolution of cooperation, but also depends on the explanation of multilevel selection theory. The central question it tries to answer is how groups become individuals. Individuality is a complex trait, yet a series of stages may exist allowing evolution to get from one kind of individual to another; for example, Michod [64] listed the steps involved in the transition from unicellular to multicellular life. These steps, according to the author, can be applied more generally to other evolutionary transitions. One possible extension of our model is to consider the steps suggested by Michod to study the evolution of individuality, which may lead to better understanding evolutionary transitions.

Bibliography

[1] T. F. H. Allen and T. B. Starr. *Hierarchy: Perspectives for Ecological Complexity.* The University of Chicago Press, Chicago, IL, USA., 1982.

[2] R. Axelrod. *The Evolution of Cooperation.* Basic Books, 1984.

[3] W. Banzhaf. On the dynamical of competition in a simple artificial chemistry. *Nonlinear Phenomena in Complex Systems*, 5(4):318–324, 2002.

[4] W. Banzhaf, P. Nordin, R. Keller, and F. Francone. *Genetic Programming — An Introduction.* Morgan Kaufmann, San Francisco, CA, USA, 1998.

[5] M. E. Borrello. The rise, fall and resurrection of group selection. *Endeavour*, 29(1):43–47, March 2005.

[6] M. Brameier and W. Banzhaf. A comparison of linear genetic programming and neural networks in medical data mining. *IEEE Transactions on Evolutionary Computation*, 5(1):17–26, February 2001.

[7] M. Brameier and W. Banzhaf. Evolving teams of predictors with linear genetic programming. *Genetic Programming and Evolvable Machines*, 2(4):381–407, December 2001.

[8] M. F. Brameier and W. Banzhaf. *Linear Genetic Programming.* Springer, New York, 2007.

[9] L. Buss. *The Evolution of Individuality.* Princeton University Press, New Jersey, 1987.

[10] D. Chu and D. J. Barnes. Group selection vs multilevel selection: Some example models using evolutionary games. In *CEC 2009: Proceedings of the IEEE Congress on Evolutionary Computation*, pages 808–814. IEEE, 2009.

[11] P. Collet, E. Lutton, F. Raynal, and M. Schoenauer. Individual GP: an alternative viewpoint for the resolution of complex problems. In W. Banzhaf, J. Daida, and et al., editors, *GECCO '99: Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation*, volume 2, pages 974–981. Morgan Kaufmann, 1999.

[12] D. M. Craig. Group selection versus individual selection: An experimental analysis. *Evolution*, 36(2):271–282, 1982.

[13] J. Damuth and I. L. Heisler. Alternative formulations of multilevel selection. *Biology and Philosophy*, 3(4):407–430, 1988.

[14] C. R. Darwin. *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life.* London: John Murray, 1st edition, 1859.

[15] C. R. Darwin. *The Descent of Man and Selection in Relation to Sex.* Murray, London, 2nd edition, 1871.

[16] R. Dawkins. *The Selfish Gene.* Oxford : Oxford University Press, 1976.

[17] R. Dawkins. *The Extended Phenotype.* Oxford : Oxford University Press, 1982.

[18] E. D. de Jong, D. Thierens, and R. A. Watson. Hierarchical genetic algorithms. In X. Yao, E. K. Burke, and et al., editors, *PPSN VIII: Proceedings of 8th International Conference on Parallel Problem Solving from Nature*, volume 3242/2004 of *LNCS*, pages 232–241. Springer Berlin / Heidelberg, 2004.

[19] K. A. de Jong. *Analysis of the Behaviour of a Class of Genetic Adaptive Systems.* PhD thesis, Dept. of Computer and Communication Sciences, University of Michigan, USA, 1975.

[20] P. Dittrich, J. Ziegler, and W. Banzhaf. Artificial chemistries — a review. *Artificial Life*, 7(3):225–275, 2001.

[21] L. A. Dugatkin. *Cooperation Among Animals.* Oxford: Oxford University Press, 1997.

[22] E. Dunn, G. Olague, and E. Lutton. Parisian camera placement for vision metrology. *Pattern Recognition Letters, Special issue: Evolutionary computer vision and image understanding*, 27(11):1209–1219, 2006.

[23] A. Eiben and J. Smith. *Introduction to Evolutionary Computing.* Natural Computing Series. Springer, 1st edition, 2003.

[24] J. A. Fletcher and M. Zwick. N-player prisoner's dilemma in multiple groups : A model of multilevel selection. In E. Boudreau and C. Maley, editors, *Artificial Life VII: Proceedings of the 7th International Conference on Artificial Life Workshop*, pages 86–89, 2000.

[25] J. A. Fletcher and M. Zwick. The evolution of altruism: Game theory in multilevel selection and inclusive fitness. *Journal of Theoretical Biology*, 245:26–36, 2007.

[26] D. B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press, 1995.

[27] D. B. Fogel. What is evolutionary computation? *IEEE Spectrum*, 37(2):26, 28–32, Feb. 2000.

[28] S. Forrest, R. E. Smith, B. Javornik, and A. S. Perelson. Using genetic algorithms to explore pattern recognition in the immune system. *Evolutionary Computation*, 1(3):191–211, 1993.

[29] A. Frank and A. Asuncion. UCI machine learning repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, 2011.

[30] D. J. Futuyma. *Evolution*. Sunderland, Massachusetts: Sinauer Associates, Inc., 1st edition, 2005.

[31] S. Gavrilets. Rapid transition towards the division of labor via evolution of developmental plasticity. *PLoS Comput Biol*, 6(6):1–10, June 2010.

[32] S. Goings and C. Ofria. Ecological approaches to diversity maintenance in evolutionary algorithms. In *ALife '09: IEEE Symposium on Artificial Life*, pages 124–130. IEEE, 2009.

[33] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, MA, USA, 1989.

[34] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems*, 3:493–530, 1989.

[35] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In J. J. Grefenstette, editor, *ICGA 1987: Proceedings of the 2nd International Conference on Genetic Algorithms*, pages 41–49, Cambridge, MA, 1987. Lawrence Erlbaum Associates.

[36] H. J. Goldsby, D. B. Knoester, J. Clune, P. K. McKinley, and C. Ofria. The evolution of division of labor. In G. Kampis, I. Karsai, and E. Szathmáry, editors, *ECAL 2009: Proceedings of the 10th European Conference on Artificial Life, Part II*, volume 5778/2011 of *LNCS*, pages 10–18. Springer Berlin / Heidelberg, 2011.

[37] C. J. Goodnight. Experimental studies of community evolution i: The response to selection at the community level. *Evolution*, 44(6):1614–1624, 1990.

[38] C. J. Goodnight and L. Stevens. Experimental studies of group selection: What do they tell us about group selection in nature? *The American Naturalist*, 150(S1, Multilevel Selection: A Symposium Organized by David Sloan Wilson):S59–S79, 1997.

[39] D. Hales. Emergent group level selection in a peer-to-peer network. *Complexus*, 3(1-3):108–118, 2006.

[40] W. D. Hamilton. The genetical evolution of social behavior. *The Journal of Theoretical Biology*, 7(1):1–16, 1964.

[41] F. Heylighen. Evolution, selfishness and cooperation. *Journal of Ideas*, 2(4):70–76, 1992.

[42] J. Holland and J. Reitman. Cognitive systems based on adaptive algorithms. *ACM SIGART Bulletin*, 63:49–49, 1978.

[43] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, USA, 1975.

[44] J. Horn. *The Nature of Niching: Genetic Algorithms and the Evolution of Optimal, Cooperative Populations.* PhD thesis, University of Illinois at Urbana-Champaign, USA, 1997.

[45] J. Horn. Resource-based fitness sharing. In J. J. M. Guervós, P. Adamidis, and et al., editors, *PPSN VII: Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, volume 2439/2002 of *LNCS*, pages 381–390. Springer Berlin / Heidelberg, 2002.

[46] J. Horn and D. E. Goldberg. Natural niching for evolving cooperative classifiers. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, editors, *Proceedings of the 1st Annual Genetic Programming Conference*, pages 553–564. MIT Press, 1996.

[47] D. Hull. Units of evolution: A metaphysical essay. In R. Jensen and R. Harré, editors, *The Philosophy of Evolution*, pages 23–44. Brighton, England: Harvester Press, 1981.

[48] G. Ichinose and T. Arita. Cooperation achieved by migration and evolution in a multilevel selection context. In *ALife '07: IEEE Symposium on Artificial Life*, pages 236–242. IEEE, 2007.

[49] J. R. Koza. *Genetic Programming: On the Programming of Computers by means of Natural Evolution.* MIT Press, 1992.

[50] T. Lenaerts. *Different Levels of Selection in Artificial Evolutionary Systems.* PhD thesis, Vrije Universiteit Brussel, Belgium, 2003.

[51] T. Lenaerts, A. Defaweux, and J. van Hemert. The evolutionary transition algorithm: Evolving complex solutions out of simpler ones. In R. Chiong, editor, *Nature-Inspired Algorithms for Optimisation*, volume 193 of *Studies in Computational Intelligence*, pages 103–131. Springer, 2009.

[52] R. C. Lewontin. The unit of selection. *Annual Review of Ecology and Systematics*, 1:1–18, 1970.

[53] P. Lichodzijewski and M. Heywood. Managing team-based problem solving with symbiotic bid-based genetic programming. In C. Ryan and M. Keijzer, editors, *GECCO '08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pages 363–370. ACM, 2008.

[54] P. Lichodzijewski and M. Heywood. Symbiosis, complexification and simplicity under GP. In M. Pelikan and J. Branke, editors, *GECCO '10: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pages 853–860. ACM, 2010.

[55] E. Lieberman, C. Hauert, and M. A. Nowak. Evolutionary dynamics on graphs. *Nature*, 433:312–316, 2005.

[56] S. W. Mahfoud. Crowding and preselction revisited. In R. Männer and B. Manderick, editors, *PPSN II: Proceedings of the 2nd International Conference on Parallel Problem Solving from Nature*, pages 27–36. Elsevier, 1992.

[57] S. W. Mahfoud. *Niching method for genetic algorithms.* PhD thesis, University of Illinois at UrbanaChampaign, USA, 1995.

[58] O. Z. Maimon and L. Rokach. *Data mining and knowledge discovery handbook.* Springer, 2005.

[59] J. Maynard Smith. Group selection and kin selection. *Nature*, 201:1145–1147, 1964.

[60] J. Maynard Smith and E. Szathmáry. *The Major Transitions in Evolution.* Oxford University Press, 1995.

[61] E. Mayr. *The Growth of Biological Thought: Diversity, Evolution, and Inheritance.* Belknap, Cambridge, Massachusetts, 1982.

[62] J. McCarthy. *LISP 1.5 Programmer's Manual.* The MIT Press, 1962.

[63] R. Michod. *Darwinian Dynamics, Evolutionary Transitions in Fitness and Individuality.* Princeton University Press, 1999.

[64] R. E. Michod. Evolution of individuality during the transition from unicellular to multicellular life. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 104:86138618, 2007.

[65] R. E. Michod and A. M. Nedelcu. On the reorganization of fitness during evolutionary transitions in individuality. *Integrative and Comparative Biology*, 43(1):64–73, 2003.

[66] T. Miconi. *The Road to Everywhere: Evolution, Complexity and Progress in Natural and Artificial Systems.* PhD thesis, The University of Birmingham, U.K., 2007.

[67] M. Mitchell. *An Introduction to Genetic Algorithms.* MIT press, 1996.

[68] L. Mui. Towards realistic models for evolution of cooperation. MIT LCS Memorandum, 2002.

[69] W. M. Muir. Group selection for adaptation to multiple-hen cages: Selection program and direct responses. *Poultry Science*, 75(4):447–458, 1996.

[70] M. A. Nowak. Five rules for the evolution of cooperation. *Science*, 314:1560–1563, 2006.

[71] M. A. Nowak and K. Sigmund. Evolution of indirect reciprocity by image scoring. *Nature*, 393:573–577, 1998.

[72] M. A. Nowak and K. Sigmund. How populations cohere: Five rules for cooperation. In R. M. May and A. McLean, editors, *Theoretical Ecology: Principles and Applications*, pages 7–16. Oxford: Oxford University Press, 2007.

[73] S. Okasha. Why won't the group selection controversy go away? *British Journal for the Philosophy of Science*, 52(2001):25–50, 2001.

[74] S. Okasha. Recent work on the levels of selection problem. *Human Nature Review*, 3(2003):349–356, 2003.

[75] S. Okasha. Multi-level selection and the major transitions in evolution. *Philosophy of Science*, 72:1013–1028, 2005.

[76] S. Okasha. *Evolution and the Levels of Selection.* Oxford University Press, 2006.

[77] S. Okasha. The levels of selection debate: Philosophical issues. *Philosophy Compass*, 1(1):7485, 2006.

[78] S. Okasha. Units and levels of selection. In S. Sarkar and A. Plutynski, editors, *A Companion to the Philosophy of Biology*, chapter 8, pages 138–156. Oxford: Blackwell, 2008.

[79] G. Olague, E. Dunn, and E. Lutton. Individual evolution as an adaptive strategy for photogrammetric network design. In C. Cotta, M. Sevaux, and K. Sörensen, editors, *Adaptive and Multilevel Metaheuristics*, volume 136 of *Studies in Computational Intelligence*, pages 157–176. Springer, 2008.

[80] M. A. Potter. *The Design and Analysis of a Computational Model of Cooperative Coevolution.* PhD thesis, George Mason University, USA, 1997.

[81] M. A. Potter and K. A. de Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, 2000.

[82] S. T. Powers, A. S. Penn, and R. A. Watson. Individual selection for cooperative group formation. In F. A. e Costa, L. M. Rocha, and et al., editors, *ECAL 2007:*

Proceedings of 9th European Conference on Artificial Life, volume 4648 /2007 of *LNCS*, pages 585–594. Springer Berlin / Heidelberg, 2007.

[83] S. T. Powers, A. S. Penn, and R. A. Watson. The efficacy of group selection is increased by coexistence dynamics within groups. In S. Bullock, J. Noble, R. Watson, and M. A. Bedau, editors, *Artificial Life XI: Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems*, pages 498–505. MIT Press, 2008.

[84] I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme und Prinzipien der biologischen Evolution.* Frommann-Holzboog, Stuttgart, Germany, 1973.

[85] S. N. Salthe. *Evolving Hierarchical Systems: Their Structure and Representation.* Columbia University Press, NY, 1985.

[86] H. P. Schwefel. *Numerical Optimization of Computer Models.* John Wiley & Sons, New York, NJ, USA, 1981.

[87] L. S. Shapley. A value for n-person games. In H. Kuhn and A. Tucker, editors, *Contributions to the Theory of Games, Volume II*, volume 28 of *Annals of Mathematical Studies*, pages 307–317. Princeton University Press, 1953.

[88] H. Simon. A mechanism for social selection and successful altruism. *Science*, 250:1665–1668, 1990.

[89] R. E. Smith, S. Forrest, and A. S. Perelson. Searching for diverse, cooperative populations with genetic algorithms. *Evolutionary Computation*, 1(2):127–149, 1993.

[90] E. Sober and D. S. Wilson. *Unto Others: The Evolution and Psychology of Unselfish Behavior*. Harvard University Press, 1999.

[91] D. Song, M. I. Heywood, and A. N. Zincir-Heywood. Training genetic programming on half a million patterns: An example from anomaly detection. *IEEE Transactions on Evolutionary Computation*, 9(3):225–239, 2005.

[92] T. Soule. Cooperative evolution on the intertwined spirals problem. In C. Ryan, T. Soule, and et al., editors, *EuroGP 2003: Proceedings of the 6th European Conference on Genetic Programming*, volume 2610/2003 of *LNCS*, pages 434–442. Springer Berlin / Heidelberg, 2003.

[93] T. Soule and P. Komireddy. Orthogonal evolution of teams: A class of algorithms for evolving teams with inversely correlated errors. In R. L. Riolo, T. Soule, and B. Worzel, editors, *Genetic Programming Theory and Practice IV*, volume 5 of *Genetic and Evolutionary Computation*, chapter 8, pages 79–95. Springer, 2006.

[94] P. Stone and M. Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robotics*, 8(3):345383, 2000.

[95] P. 't Hoen and E. de Jong. Evolutionary multi-agent systems. In X. Yao, E. K. Burke, and et al., editors, *PPSN VIII: Proceedings of 8th International Conference on Parallel Problem Solving from Nature*, volume 3242/2004 of *LNCS*, pages 872–881. Springer Berlin / Heidelberg, 2004.

[96] R. Thomason and T. Soule. Novel ways of improving cooperation and performance in ensemble classifiers. In H. Lipson, editor, *GECCO '07: Proceedings*

of the 9th Annual Conference on Genetic and Evolutionary Computation, pages 1708–1715. ACM, 2007.

[97] A. Traulsen and M. A. Nowak. Evolution of cooperation by multilevel selection. In *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, volume 103, pages 10952–10955. National Academy of Sciences, 2006.

[98] A. Traulsen, N. Shoresh, and M. A. Nowak. Analytical results for individual and group selection of any intensity. *Bulletin of Mathematical Biology*, 70(5):1410–1424, 2008.

[99] R. L. Trivers. The evolution of reciprocal altruism. *The Quarterly Review of Biology*, 46(1):3557, 1971.

[100] E. Tuci, C. Ampatzis, F. Vicentini, and M. Dorigo. Evolving homogeneous neurocontrollers for a group of heterogeneous robots: Coordinated motion, cooperation, and acoustic communication. *Artificial Life*, 14(2):157–178, Spring 2008.

[101] J. W. Valentine. Architectures of biological complexity. *Integrative and Comparative Biology*, 43(1):99–103, 2003.

[102] M. J. Wade. A critical review of the models of group selection. *The Quarterly Review of Biology*, 53(2):101–114, 1978.

[103] R. A. Watson. *Compositional Evolution: Interdisciplinary Investigations in Evolvability, Modularity, and Symbiosis*. PhD thesis, Brandeis University, USA, 2002.

[104] R. A. Watson and J. B. Pollack. A computational model of symbiotic composition in evolutionary transitions. *Biosystems*, 69:187–209, 2003.

[105] S. West, A. Griffin, and A. Gardner. Social semantics: Altruism, cooperation, mutualism, strong reciprocity and group selection. *Journal of Evolutionary Biology*, 20:415–432, 2009.

[106] R. P. Wiegand. *An Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, George Mason University, USA, 2003.

[107] Wikipedia. Biological organisation — Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Biological_organization, 2010. [Online; accessed 30-August-2010].

[108] Wikipedia. Classification — Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Classification_%28machine_learning%29, 2010. [Online; accessed 30-August-2010].

[109] Wikipedia. Fixation — Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Fixation_(28population_genetics), 2010. [Online; accessed 30-August-2010].

[110] Wikipedia. Set covering problem — Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Set_cover_problem, 2010. [Online; accessed 30-August-2010].

[111] D. S. Wilson. A theory of group selection. In *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, volume 72, pages 143–146. National Academy of Sciences, 1975.

[112] D. S. Wilson. The group selection controversy: History and current status. *Annu. Rev. Ecol. Syst.*, 14:159–187., 1983.

[113] D. S. Wilson and E. O. Wilson. Survival of the selfless. *New Scientist*, 3:42–46, 2007.

[114] D. S. Wilson and E. O. Wilson. Evolution for the good of the group. *American Scientist*, 96:380–389, 2008.

[115] G. Wilson and M. Heywood. Introducing probabilistic adaptive mapping developmental genetic programming with redundant mappings. *Genetic Programming and Evolvable Machines*, 8(2):187–220, 2007.

[116] D. H. Wolpert, K. R. Wheeler, and K. Tumer. General principles of learning-based multi-agent systems. In O. Etzioni, J. P. Müller, and J. M. Bradshaw, editors, *AGENTS '99: Proceedings of the 3rd Annual Conference on Autonomous Agents*, pages 77–83. ACM, 1999.

[117] S. X. Wu and W. Banzhaf. A hierarchical cooperative evolutionary algorithm. In M. Pelikan and J. Branke, editors, *GECCO '10: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pages 233–240. ACM, 2010. (**Best Paper Award**).

[118] S. X. Wu and W. Banzhaf. The use of computational intelligence in intrusion detection systems: A review. *International Journal of Applied Soft Computing*, 10:1–35, 2010.

[119] S. X. Wu and W. Banzhaf. Evolutionary transition through a new multilevel selection model. In T. Lenaerts, M. Giacobini, and et al., editors, *ECAL 2011:*

Proceedings of the 11th European Conference on Artificial Life, pages 874–881. MIT Press, 2011.

[120] S. X. Wu and W. Banzhaf. Investigations of Wilson's and Traulsen's group selection models in evolutionary computation. In G. Kampis, I. Karsai, and E. Szathmáry, editors, *ECAL 2009: Proceedings of the 10th European Conference on Artificial Life, Part II*, volume 5778/2011 of *LNCS*, pages 1–9. Springer Berlin / Heidelberg, 2011.

[121] S. X. Wu and W. Banzhaf. *Knowledge Mining Using Intelligent Agents*, chapter 2: The Use of Evolutionary Computation in Knowledge Discovery: The Example of Intrusion Detection Systems, pages 27–59. Imperial College Press, 2011.

[122] S. X. Wu and W. Banzhaf. Rethinking multilevel selection in genetic programming. In N. Krasnogor and P. L. Lanzi, editors, *GECCO '11: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, pages 1403–1410. ACM, 2011. (**Best Paper Award**).

[123] V. C. Wynne-Edwards. *Animal Dispersion in Relation to Social Behavior*. Oliver and Boyd, Edinburgh, UK, 1962.

[124] X. Yao. Evolutionary computation: A gentle introduction. In F. S. Hillier, editor, *Evolutionary Optimization*, volume 48 of *International Series in Operations Research & Management Science*, pages 27–53. Springer, 2003.

CENTRE FOR NFLD. STUDIES
JUL 18 1978
MEMORIAL UNIVERSITY
OF NEWFOUNDLAND