Ph.D. Thesis

# Learning Potential Inheritance in Baldwinian Evolution

Jun. 1, 2012

Supervisor Professor Hitoshi Iba

Graduate School of Engineering, the University of Tokyo

Department of Electical Engineering and Information Systems
37-097408

Shu Liu

# Abstract

Inspired by organism evolution, evolutionary algorithms have attracted research interests for several decades, and have been proved effective and efficient to solve optimization problems. For further enhancement, local search are introduced into evolution, imitating the biological fact that organism individuals change themselves to better fit the environment during their lifetime. In the context of evolutionary computation, Baldwinian evolution hybridizes global search through population-based evolution with local search through individual refinements. Refinement influences selection, however, unlike Lamarckian evolution, refined traits are not inherited by the offspring. In Baldwinian evolution, local search guides evolution in an indirect manner, termed "the Baldwin effect". Conventional studies revealed that Baldwinian learning can enhance search, however, there is still considerable work to be done to understand the mechanisms. This thesis investigates Baldwinian evolution's mechanisms in depth. By proposing the method of analyzing individual dynamics, we present some novel views. We find that the substantial inheritance in Baldwinian evolution is the potential to achieve high fitness through learning, that the realization of learning potential is influenced by inheritable and noninheritable factors in evolution, and that learning cost penalties inhibit learning intensity. Our results provide knowledge of Baldwinian evolution's mechanisms, and directions to possible applications.

# Acknowledgment

In writing this thesis, in my research, in my recent 3-year life being a doctoral student, many people helped me. This thesis is a product of not only my efforts, but also theirs. I would like to present my gratitude here.

First of all I would like to thank my parents. They supported my choice of pursuing the doctor's degree in the University of Tokyo. They are not experts in my research field, but are the first tutors in my life. If I am conscientious in my work, if I am positive in my life, that is because I want to be such a person like my parents.

I would like to present my greatest thanks to my advisor, professor Hitoshi Iba. He guides me in how to do research, provides me an environment that I can focus on research, and encourages me to find my own research. He advised me when I was confused by the research problems, and his curiousness and knowledge about science and technology impressed me. Working as his student, I find my view on science and philosophy extended.

I am grateful to all the members of Iba Lab. The whole laboratory is a warm family, they discuss research problems with me, give me advice in daily routines in Japan, and talk together about many relaxing topics. They make the laboratory a comfortable place for research and cultural communications. I thank them all and forgive me for not listing all their names here. Among the members of the laboratory, I would like to specially thank Akio Watanabe, who helped me a lot when I first came to Japan; Dr. Noman Nasimul and Leon Palafox, who advised me much on my studies and papers; also Dr. Toshihiko Yanase, Dr. Makoto Tanji and Dr. Claus Aranha, who gave me many precious suggestions in my student life.

I would also thank some friends out of the laboratory. They read my papers, discussed philosophy with me, and sometimes directed me in programming techniques. Among them I would like to specially mention Dr. Yue Dong, who provided many useful suggestions in my academic activities.

Last but not least, I would like to thank all the professors gave me lectures and advice, all the university staffs helped my campus life, all the researchers I met in conferences and gave me comments, and all my friends supported me. I am not willing to make this acknowledgment a long name list for pages, so please allow me to just express my gratitude to all these people.

# Contents

**Chapter 1 Introduction** **1**

1.1 Lamarckian and Baldwinian Evolution . . . . . . . . . . . . . . . . . . . . . . . . . . 2

1.1.1 Biological Theories of Lamarckism and Baldwinism . . . . . . . . . . . . . 2

1.1.2 Lamarckian and Baldwinian Algorithms . . . . . . . . . . . . . . . . . . . . . 3

1.2 The Baldwin Effect . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

1.2.1 Hinton & Nowlan's Example . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

1.2.2 Two-phase Baldwin Effect . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

1.3 Computational Efficiency . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

1.4 Contents of Thesis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

**Chapter 2 Learning Potential** **12**

2.1 Introduction to the Chapter . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

2.2 Lamarckian and Baldwinian Learning . . . . . . . . . . . . . . . . . . . . . . . . . 14

2.3 Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

2.3.1 NK Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

2.3.2 Plasticity Codes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

2.3.3 Experiment Settings . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

2.4 Experiments and Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19

2.4.1 Improvement Through the Generations . . . . . . . . . . . . . . . . . . . . . 19

2.4.2 Step-by-Step Improvement . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23

2.4.3 Changing the Budget . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25

2.5 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29

2.6 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

**Chapter 3 Realization of Learning Potential** **31**

3.1 Introduction to the Chapter . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

3.2 Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

3.2.1 NK Model with Plasticity . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

3.2.2 Learning Schemes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34

3.2.3 Crossover Operators . . . 37

3.3 Learning Scheme Comparison . . . 38

3.3.1 Low Epistasis Landscape . . . 38

3.3.2 Moderate Epistasis Landscape . . . 43

3.3.3 High Epistasis Landscape . . . 48

3.4 Crossover Comparison . . . 49

3.4.1 Low Epistasis Landscape . . . 52

3.4.2 Moderate Epistasis Landscape . . . 56

3.4.3 High Epistasis Landscape . . . 59

3.5 Discussion . . . 63

3.6 Conclusion . . . 64

**Chapter 4 Learning Intensity and Cost Punishment 65**

4.1 Introduction to the Chapter . . . 66

4.2 Model . . . 69

4.2.1 NK Model with Plasticity and Learning Budget . . . 69

4.2.2 Learning Settings . . . 70

4.3 Components of Selective Pressure . . . 71

4.3.1 Low Epistasis Landscape . . . 73

4.3.2 Moderate Epistasis Landscape . . . 77

4.3.3 High Epistasis Landscape . . . 80

4.4 A Condition for Genetic Assimilation . . . 83

4.5 Discussion . . . 89

4.6 Conclusion . . . 90

**Chapter 5 Discussion 91**

5.1 From Algorithms to Components . . . 92

5.2 From Groups to Individuals . . . 96

**Chapter 6 Conclusion 102**

**Appendix A Continuous Optimization Experiments 107**

**Bibliography 123**

# List of Figures

1.1 Biological explanations of Lamarckian and Baldwinian evolution. . . . . . . . . . 2

1.2 Computational conterparts of Lamarckian and Baldwinian evolution. . . . . . . . 3

1.3 Plastic genotype and learning process, an example given by Hinton & Nowlan . . 5

1.4 Learning's extension smoothes fitness landscape. . . . . . . . . . . . . . . . . . . . 6

1.5 Learning's extension smoothes fitness landscape. . . . . . . . . . . . . . . . . . . . 7

1.6 Two phase process of the Baldwin effect . . . . . . . . . . . . . . . . . . . . . . . 8

1.7 Comparison of evolutionary and memetic search . . . . . . . . . . . . . . . . . . . 8

1.8 From macroscopic to microscopic. . . . . . . . . . . . . . . . . . . . . . . . . . . 11

2.1 Evolutionary search with Lamarckian and Baldwinian learning. . . . . . . . . . . 15

2.2 Fitness calculation of NK model with N=10, K=2. . . . . . . . . . . . . . . . . . 17

2.3 $k$-th trial in learning process. . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

2.4 Genotype with plasticity codes. . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

2.5 Fitness improvement with deterministic learning on the K=2 landscape. . . . . . 20

2.6 Fitness improvement with nondeterministic learning on the K=2 landscape. . . . 21

2.7 Fitness improvement with deterministic learning on the K=5 landscape. . . . . . 22

2.8 Fitness improvement with nondeterministic learning on the K=5 landscape. . . . 23

2.9 Step-by-step learning improvements on the K=2 landscape. . . . . . . . . . . . . 24

2.10 Step-by-step learning improvements on the K=5 landscape. . . . . . . . . . . . . 26

2.11 Periodically changing and constant budgets on the K=2 landscape. . . . . . . . . 27

2.12 Periodically changing and constant budgets on the K=5 landscape. . . . . . . . . 28

3.1 Two types of memetic search. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

3.2 Each partial fitness value is dependent on only a single $K$-neighborhood. . . . . . 33

3.3 Learning uncertainties and effect on children. . . . . . . . . . . . . . . . . . . . . 34

3.4 $k$-th trial in learning process. . . . . . . . . . . . . . . . . . . . . . . . . . . . 35

3.5 Genotype with plasticity codes. . . . . . . . . . . . . . . . . . . . . . . . . . . . 36

3.6 Change in inheritance caused by crossover and effect on children. . . . . . . . . . 37

3.7 Three types of crossovers. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39

3.8 Search capabilities on $K = 2$ landscape with the same population. . . . . . . . . 40
3.9 Learning similarities on $K = 2$ landscape when budget is 20. . . . . . . . . . . . 41
3.10 Learning improvement on $K = 2$ landscape when $b_{pre} = 20$. . . . . . . . . . . . 42
3.11 Search performance on $K = 2$ landscape when budget is 20. . . . . . . . . . . . . 43
3.12 Search capabilities on $K = 5$ landscape with the same population. . . . . . . . . 45
3.13 Learning similarities on $K = 5$ landscape when budget is 20. . . . . . . . . . . . 45
3.14 Learning improvement on $K = 5$ landscape when $b_{pre} = 20$. . . . . . . . . . . . 46
3.15 Search performance on $K = 5$ landscape when budget is 20. . . . . . . . . . . . . 47
3.16 Search capabilities on $K = 15$ landscape with the same population. . . . . . . . . 48
3.17 Learning similarities on $K = 15$ landscape when budget is 20. . . . . . . . . . . . 49
3.18 Learning improvement on $K = 15$ landscape when $b_{pre} = 20$. . . . . . . . . . . 50
3.19 Search performance on $K = 15$ landscape when budget is 20. . . . . . . . . . . . 51
3.20 Search without learning on $K = 2$ landscape. . . . . . . . . . . . . . . . . . . . . . 52
3.21 Learning similarities on $K = 2$ landscape when budget is 20. . . . . . . . . . . . 53
3.22 Learning improvement on $K = 2$ landscape when $b_{pre} = 20$. . . . . . . . . . . . 54
3.23 Search performance on $K = 2$ landscape when budget is 20. . . . . . . . . . . . . 55
3.24 Search without learning on $K = 5$ landscape. . . . . . . . . . . . . . . . . . . . . . 56
3.25 Learning similarities on $K = 5$ landscape when budget is 20. . . . . . . . . . . . 57
3.26 Learning improvement on $K = 5$ landscape when $b_{pre} = 20$. . . . . . . . . . . . 57
3.27 Search performance on $K = 5$ landscape when budget is 20. . . . . . . . . . . . . 58
3.28 Search without learning on $K = 15$ landscape. . . . . . . . . . . . . . . . . . . . . 60
3.29 Learning similarities on $K = 15$ landscape when budget is 20 . . . . . . . . . . . 60
3.30 Learning improvement on $K = 15$ landscape when $b_{pre} = 20$. . . . . . . . . . . 61
3.31 Search performance on $K = 15$ landscape when budget is 20. . . . . . . . . . . . 62

4.1 Two-phase Baldwin effect. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 66
4.2 Equivalent landscape with smoothing and cost punishment. . . . . . . . . . . . . 68
4.3 Equivalent landscape extended along the cost dimension. . . . . . . . . . . . . . . 69
4.4 Genotype with plasticity. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 70
4.5 $k$-th trial in learning process. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 71
4.6 Tournament selection with size 2. . . . . . . . . . . . . . . . . . . . . . . . . . . . 72
4.7 Search performance on $K = 2$ landscape with 2-tournament selection. . . . . . . 74
4.8 Selected individual types on $K = 2$ landscape with 2-tournament selection. . . . 75
4.9 Search performance on $K = 5$ landscape with 2-tournament selection. . . . . . . 78
4.10 Selected individual types on $K = 5$ landscape with 2-tournament selection. . . . 79

4.11 Search performance on $K = 15$ landscape with 2-tournament selection. . . . . . . 81
4.12 Selected individual types on $K = 15$ landscape with 2-tournament selection. . . . 82
4.13 Roulette wheel selection. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 84
4.14 Search performance on $K = 2$ landscape with roulette wheel selection. . . . . . . 85
4.15 Search performance $K = 5$ landscape with roulette wheel selection. . . . . . . . . 86
4.16 Search performance on $K = 15$ landscape with roulette wheel selection. . . . . . 87

5.1 Baldwinian evolution components studied in this thesis. . . . . . . . . . . . . . . 94
5.2 The roles of learning on fitness landscape (by Suzuki *et al* [51]). . . . . . . . . . . 94
5.3 Two phase process of the Baldwin effect . . . . . . . . . . . . . . . . . . . . . . . 95
5.4 Step-wise cost-performance analysis. . . . . . . . . . . . . . . . . . . . . . . . . 97
5.5 Compare children's and their parents' learning steps. . . . . . . . . . . . . . . . . 98
5.6 Global simplification and divergence of individuals. . . . . . . . . . . . . . . . . . 99
5.7 Tournament selection with size 2. . . . . . . . . . . . . . . . . . . . . . . . . . . 100

A.1 Fitness improvements with functions $F_1 \sim F_8$ using BLX and CMA-ES. . . . . . 111
A.2 Fitness improvements with functions $F_9 \sim F_{17}$ using BLX and CMA-ES. . . . . . 112
A.3 Fitness improvements with functions $F_{18} \sim F_{25}$ using BLX and CMA-ES. . . . . 113
A.4 Fitness improvements with functions $F_1 \sim F_8$ using BLX and Solis-Wets. . . . . 115
A.5 Fitness improvements with functions $F_9 \sim F_{17}$ using BLX and Solis-Wets. . . . . 116
A.6 Fitness improvements with functions $F_{18} \sim F_{25}$ using BLX and Solis-Wets. . . . 117
A.7 Fitness improvements with functions $F_1 \sim F_8$ using PBLX and CMA-ES. . . . . 119
A.8 Fitness improvements with functions $F_9 \sim F_{17}$ using PBLX and CMA-ES. . . . . 120
A.9 Fitness improvements with functions $F_{18} \sim F_{25}$ using PBLX and CMA-ES. . . . 121

# List of Tables

2.1 Experiment Parameters . . . 18

3.1 Experiment Parameters . . . 33
3.2 Learning improvement on $K = 2$ landscape . . . 42
3.3 Search performance on $K = 2$ landscape . . . 44
3.4 Learning improvement on $K = 5$ landscape . . . 46
3.5 Search performance on $K = 5$ landscape . . . 47
3.6 Learning improvement on $K = 15$ landscape . . . 50
3.7 Search performance on $K = 15$ landscape . . . 51
3.8 Learning improvement on $K = 2$ landscape . . . 54
3.9 Search performance on $K = 2$ landscape . . . 55
3.10 Learning improvement on $K = 5$ landscape . . . 58
3.11 Search performance on $K = 5$ landscape . . . 59
3.12 Learning improvement on $K = 15$ landscape . . . 61
3.13 Search performance on $K = 15$ landscape . . . 62

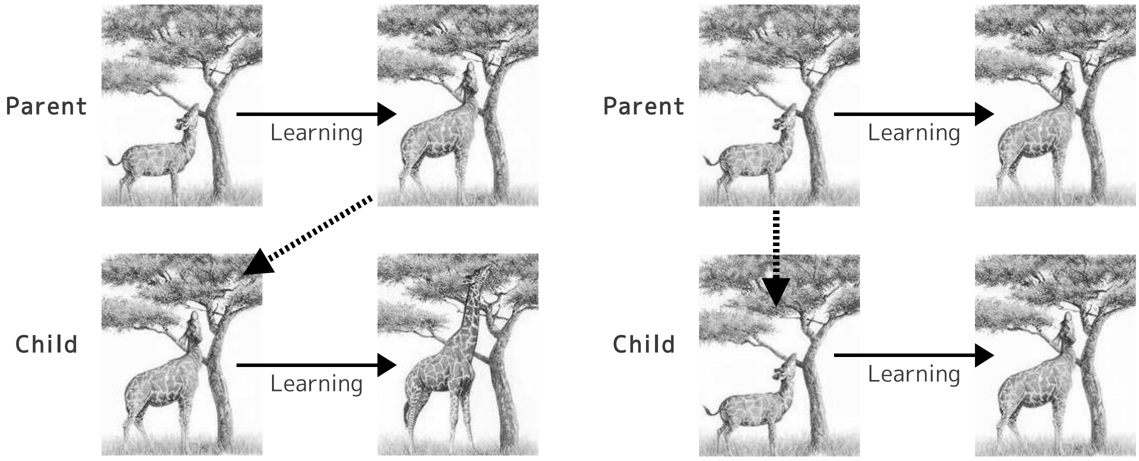
4.1 Experiment Parameters . . . 70
4.2 Search performance on $K = 2$ landscape with 2-tournament selection . . . 73
4.3 Search performance on $K = 5$ landscape with 2-tournament selection . . . 77
4.4 Search performance on $K = 15$ landscape with 2-tournament selection . . . 80
4.5 Search performance on $K = 2$ landscape with roulette wheel selection . . . 85
4.6 Search performance on $K = 5$ landscape with roulette wheel selection . . . 87
4.7 Search performance on $K = 15$ landscape with roulette wheel selection . . . 88

A.1 Functions in IEEE CEC 2005 Test Suite . . . 108
A.2 Fitness and potential scales of the 12th generation using BLX and CMA-ES . . . 110
A.3 Fitness and potential scales of the 12th generation using BLX and Solis-Wets . . . 114
A.4 Fitness and potential scales of the 12th generation using PBLX and CMA-ES . . . 118

# Chapter 1

# Introduction

Evolutionary algorithms imitate organism evolution in the natural world, and are proved effective and efficient to handle optimization problems, especially when the search space is huge and the landscape is irregular. To accelerate evolution, and to overcome even more difficult problems, researchers have been trying to combine evolution with learning techniques in the recent decades. Baldwinian evolution is a hybridization of evolution and learning. It claims that learning throughout the individuals' lifetime can guide evolution to better solutions, without transferring acquired characters back into genotypes. The theme of this thesis is to study Baldwinian evolution's mechanisms. First of all, this chapter introduces the basic conceptions of Baldwinian evolution, the search efficiency issue, and significance of our research.



(a) Lamarckian evolution

(b) Baldwinian evolution

Fig.1.1: Biological explanations of Lamarckian and Baldwinian evolution.

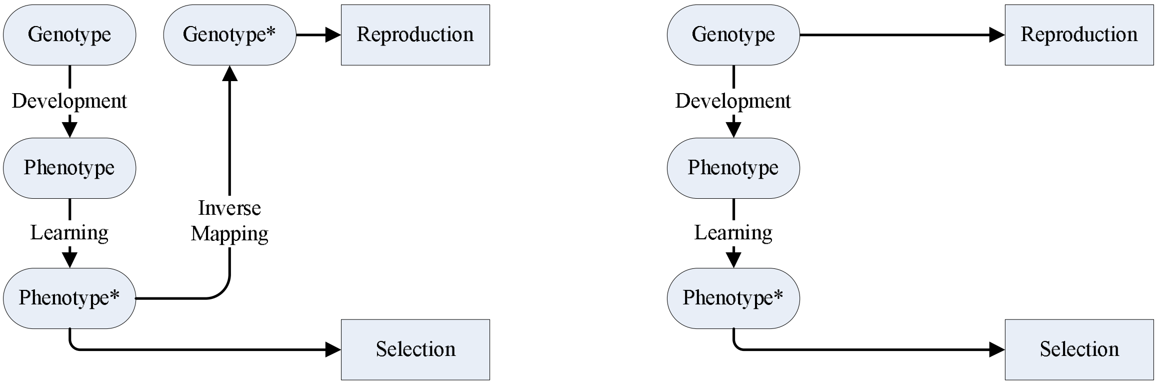
## 1.1 Lamarckian and Baldwinian Evolution

### 1.1.1 Biological Theories of Lamarckism and Baldwinism

As biologists have observed in the natural world, organisms adapt themselves during their lifetime to better fit the environment. They try various possibilities, or are taught by parents, to shift themselves, and this process begins with the characters received inherently. It is apparent that if these adaptations can be passed on to the offsprings, the process of fitting the environment will be evidently accelerated.

The most direct idea is the inheritance of acquired characters, as known as Lamarckism. It can be inferred from Lamarckism that an animal stretching its neck in its life will have children with long necks, or the children of a chess champion will also play chess well. As shown in Fig.1.1(a), a child giraffe is born with the stretched long neck, because its parent stretched the parent's neck. This theory was proposed by biologists in the period when chromosomes and genes were not discovered. With the characters inherited, the species can react immediately to the changes in environment. However, there is a distinction between genes and bodies, and an inverse mapping to genes is implausible. For example, if a person do exercises and build his muscles, he cannot change his DNA according to the muscles and ensure that his children will be strong. Lamarckism is thus refused by current biological science in general.

In 1896, the Baldwin effect was proposed independently by Baldwin [2], Morgan [37] and Orsborn [43], also known as Baldwinian evolution or ontogenic evolution. Later in 1942, Wadding-



(a) Lamarckian algorithm (b) Baldwinian algorithm

Fig.1.2: Computational conterparts of Lamarckian and Baldwinian evolution.

ton [53] reviewed the theory and presented the term ”genetic assimilation” or ”canalization”. In this theory, acquired characters do not transfer information directly back to genotypes. Instead, when an individual learns good characters in its lifetime, it is rewarded in selection, therefore a selective advantage is automatically assessed to the genotypes inherently easier to acquire good characters. It is denoted that the children of a neck-stretching animal receive no long necks but good ability in the stretching, or the children of a chess champion will be excellent in activities asking for logical thinking. As shown in Fig.1.1(b), a child is born with a short neck similar to its parent's, however, it is likely to have good capability in stretching as its parent did. The Baldwin effect has always been controversial in biology, since in the complex natural world it is too difficult to varify.

### 1.1.2 Lamarckian and Baldwinian Algorithms

No matter Lamarckism and Baldwinism are correct or not in biology, they can enhance evolutionary algorithms, and the hybridizations are termed “memetic algorithms”. The flow of a Lamarckian algorithm on an individual is demonstrated in Fig.1.2(a): the individual translates its initial genotype to phenotype/traits, adapt the phenotype through a learning process pursuing better fitness, then encode the adaptated phenotype back into the genotype and pass on the learned traits to the offspring. The flow of a Baldwinian algorithm on an individual is demonstrated in Fig.1.2(b): the individual translates its initial genotype and learns, but the following reproduction is carried out using the initial genotype. The acquired characters are not transferred back to genotype, but the initial genotype receives a bonus or penalty as the start points of good or bad final traits. As concluded in a recent survey [7], Lamarckian learning influences improved solutions and improved fitness, while Baldwinian learning influences initial solutions and improved fitness.

Considering algorithms, the most essential difference between Lamarckism and Baldwinism is the existance of an inverse mapping from phenotype to genotype. In many real world problems there is no such a mapping, or the mapping is too complicated. In this case, Baldwinism is a prior choice than Lamarckism. On the other hand, comparing to Lamarckism, Baldwinism is slower as learned characters are not encoded directly, whereas the slowness brings robustness to environment permutations.

Initially, combinations of evolution and learning are used to solve the topology and synapse strengths of artificial neural networks [27,35,54], however, the methodology is then employed in many other problems. To apply memetic algorithms to real-world problems, it is necessary to have knowledge on how evolution proceeds and which factors influence search performance. So far, the majority of researchers have focused on Lamarckian evolution since it is considered more simple and effective. For example, there are studies on optimizing parameters such as learning frequency and intensity [3,15,29,31,39], and on adapting parameters during a search [30,40–42]. However, the conclusions that have been drawn might not hold for Baldwinian evolution. For example, in a Lamarckian search, it is common to choose a random subset of the population for learning. Conversely, in the Baldwinian scenario a surviving parent is selected for its refined traits after learning, but its children may be judged on only their initial traits before learning. Such initial traits have no guarantee of being beneficial.

Several studies on Baldwinian evolution have also been performed. The guiding effect of Baldwinian learning in evolutionary searches was first verified by Hinton and Nowlan [20]. Since then, there have been works on search process characteristics of Baldwinian evolution [33, 34, 46, 51, 52, 55], on interactions of learning, evolution and development [10–12], on the hiding of Baldwinian learning [44,45], and on combining Baldwinian and Lamarckian evolutions [5, 8, 9, 21]. Attempts have also been made to improve Baldwinian algorithms and to apply them [4, 13]. However, as claimed by Turney [52], Baldwinian evolution is highly complex, and so there is still considerable work to be done.

## 1.2 The Baldwin Effect

### 1.2.1 Hinton & Nowlan's Example

The term "the Baldwin effect" is widely used in biology and evolutionary computation, however, there is not yet a clear definition and researchers may have different views on the same vague concept. The mechanism of how Baldwinian learning guides evolution presented in this chapter follows the classic article of Hinton & Nowlan [20]. In the article they presented a simple

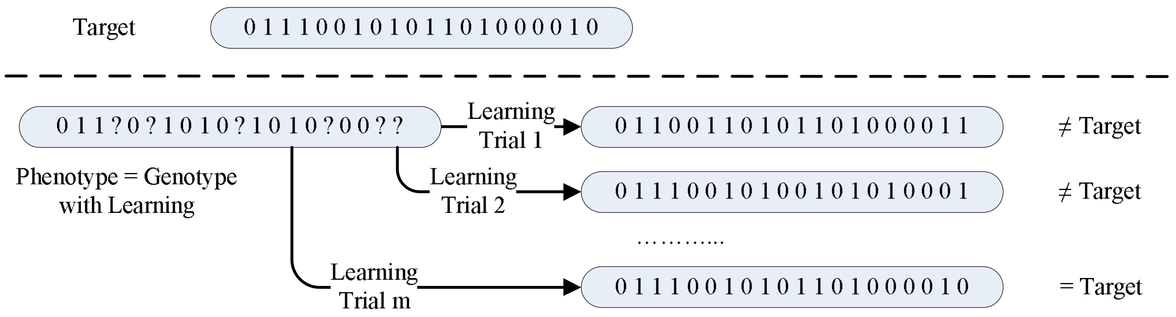


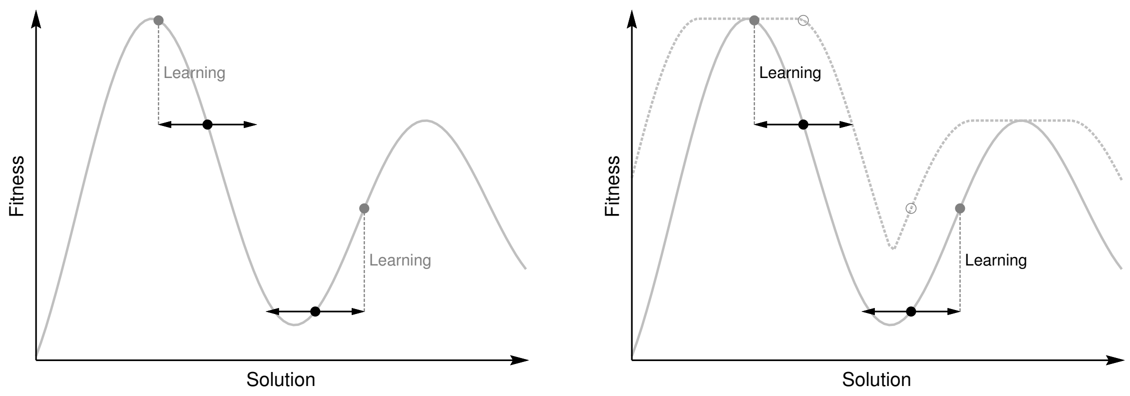
Fig.1.3: Plastic genotype and learning process, an example given by Hinton & Nowlan

and extreme, but very clear example. This example is cited and developed by many researchers in the field of evolutionary computation. We will begin with the revision on the example, too.

Suppose in a search problem, the search target is a random binary string of 20 bits. Only this target string is assigned a positive fitness, and all other strings are set zero fitness. When using 20 bit binary strings as genotypes and corresponding phenotypes, it is extremely difficult to reach the goal. For individuals different from the target, they provide no information about where the target may be. An evolutionary algorithm is thus degraded to a random search, and it reaches the target only when by good luck an individual is generated exactly the same as the target. Even after reaching the goal, crossover and mutation operations may break it.

The situation changes when plasticity is added to the genotypes. Several bits are set to ? instead of 0 or 1, which means the bits are not determined and left to learning after birth. Every individual tries 1000 times to assign 0/1 values to all the ? bits in its initial phenotype, and stops when arriving at the target or the number of trials reaches the limit, as shown in Fig.1.3. At the same time, the fitness is modified to $1 + \frac{19n}{1000}$ for individuals reaching the target in learning, where n refers to the number of remaining trials after reaching the target. As a result, many individuals have the chance to reach the target, and can inform the offsprings that they are close to the target. Furthermore, when one more bit is fixed to genotype, the according individual will need in average half the trials as before to find the target. According to the fitness function, it is ecouraged to have less ? bits through the search, so that the genotypes are driven toward the target after first reached it.

As claimed by Hinton & Nowlan, in Baldwinian algorithms, the learning process acts as a local search in the phenotype space, in a neighborhood around the expression of its genotype. This exploration is similar to generating individuals corresponding to the phenotypes that collects information about the landscape, while with a much lower cost than generating such individuals. When an individual moves to a nearby phenotype, usually it is easy to take the move and check the fitness. The learning process explores the local region efficiently, and discovers directions



(a) Learning explores a neighborhood

(b) Smoothed landscape

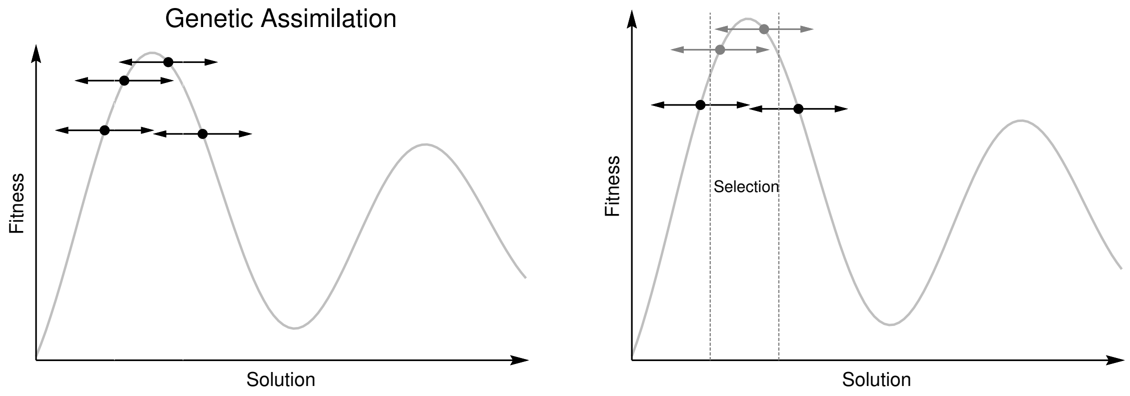
Fig.1.4: Learning's extension smoothes fitness landscape.

towards the target.

### 1.2.2 Two-phase Baldwin Effect

In Hinton & Nowlan's example, the Baldwin effect guides evolution by two mechanisms, known as "smoothing" and "genetic assimilation". In this example, the one-point target is too extreme. However, generally speaking, in many problems the access to a local optimum is like finding a needle in a haystack. Fitness decreases acutely as the solution leaves the local optimum. In this situation, only when an individual falls in the narrow range of the spike, the individual can win in selection and the algorithm can find this local optimum. Therefore, an evolutionary algorithm will not realize there is a local optimum nearby until an individual falls in the range at random. If the individuals have learning ability and can search a neighborhood around its initial position, the way to the local optimum is broadened. The learning process can be implemented by constructing an analytic model of the local area, or by stochastic search methods, or by some random trials as Hinton & Nowlan did. As shown in Fig.1.4(a), not necessarily in the narrow range, if the point is near to the spike, it reaches the spike by learning. It thus receives a fitness bonus and calls attention to the area. With learning. It is like searching in an equivalently smoothed landscape, possibly like the curve in Fig.1.4(b), where peaks become plateaus.

There is a fitness bonus in Hinton & Nowlan's model, to those who learn faster, and lead genotypes to keep less undetermined bits. This bonus comes from the cost of learning, as the cost always exists in the natural world, and is usually significant in evolutionary computation. This is a second driving force after reaching a local optimum, and emphasized by Turney [52] that it is important. Affected by this force, learned characters become fixed into genotype as



(a) All individuals reach the top

(b) Select according to learning cost

Fig.1.5: Learning's extension smoothes fitness landscape.

evolution processes. This mechanism is termed genetic assimilation, or canalization. By chance, some genius individuals may get born with good characters that its ancestors spent time to learn. These individuals have better starting points, thus cost less to learn what their ancestors learned. Under selective pressure, the genius individuals win in selection and the species in allowed to obtain even better initial traits. As shown in Fig.1.5(a), all the individuals can reach the top of the hill by learning. With selective pressure from learning cost, low cost individuals survive, and the population moves closer to the top of hill, as shown in Fig.1.5(b). The Baldwin effect does not transfer acquired characters back to genotype, but it can be observed that the evolution follows the indication of learning, as Waddington [53] claimed the process "by which a phenotypic character, which initially is produced only in response to some environmental influence, becomes, through a process of selection, taken over by the genotype, so that it is formed even in the absence of the environmental influence which had at first been necessary".

In addition, Suzuki [51] claimed there is more than genetic assimilation in Baldwin effect. In some situations, maybe caused by epistasis or other factors, the capability of learning is limited. Individuals can find hills around them by learning, but have not enough capabilities to climb to the top. As the result, there will be a period for climbing. In this period, the genius offsprings will have chance to learn further, and continue to climb the hill their ancestors discovered. This is somehow different from genetic assimilation, and enlightened some of our works.

Affected by these two basic mechanisms, the Baldwin effect acts as a two phase process in the search, as shown in Fig.1.6. In the first phase, learning assisted search dominates. Fitness grows up remarkably, while learning cost keeps high. In the second phase, genetic assimilation dominates. Fitness is stable, while learning cost decreases, even to zero in some situations. This

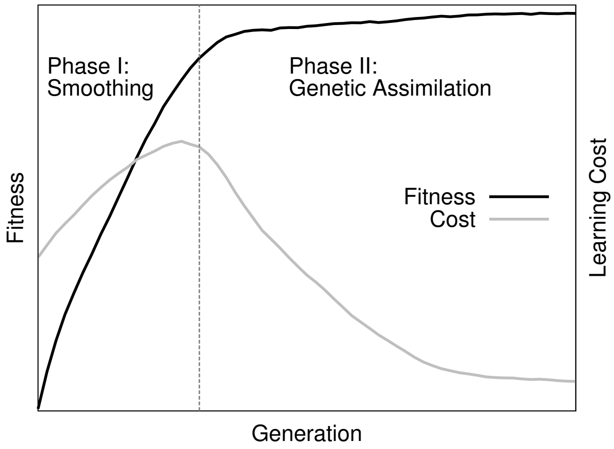
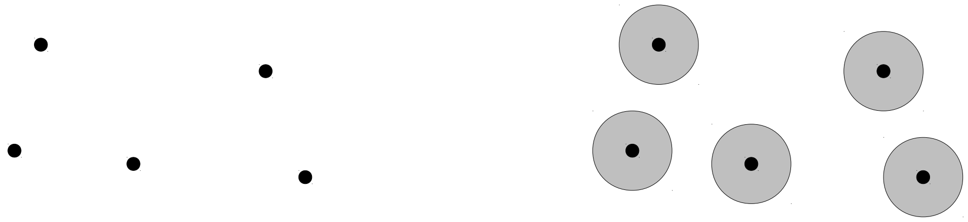


Fig.1.6: Two phase process of the Baldwin effect



(a) Evolutionary Search

(b) Memetic Search

Fig.1.7: Comparison of evolutionary and memetic search

is the basic description of the Baldwin effect. When learning is limited, as claimed by Suzuki, there will be a third phase of hill climbing, between the basic two. Fig.1.6 shows the general trends of fitness and learning cost. This is the conventional theory, and we are to go beyond it.

## 1.3 Computational Efficiency

There has been memetic algorithm applications in problems such as computational finance [1], music composition [24], aircraft design [47], job-shop scheduling [18], and drug design [32]. In applications an essential factor that must be kept in consideration is the search efficiency of the algorithm, in other words the balance of the fitness of the found solution and the cost, usually in term of computation time. To arrive at satisfying solutions within reasonable computation time, there has been investigations into the factors that influence the banlance, and the possible ways to accelerate.

As shown in Fig.1.7, in evolutionary algorithms, the search can be considered to have a set of searching points sampled in the space, and they evolve/update towards better fitness with iterative operations. On the other hand in memetic algorithms, for each searching point it is not limited to the simgle position, but also allowed to search and refine locally in a neighborhood, using either statistic or deterministic methods. This allows the population to cover more area in the space, and the covered area is expected to be relatively good as they are around the good individuals survived in evolutionary selection. On the other hand, however, such local searches call for considerable computation resources. A balance is necessary for applications, especially in Baldwinian evolution where the products of learning are limited.

The goal of optimization algorithms is to find good solutions, and in real world applications it is also in serious concern that how much time has to be paid. In fact many real world problems are reported very expensive in computation. The main reason for this is the high cost to evaluate a candidate solution in complex real world situations. A simgle evaluation call may take minutes to hours, such as in [47]. In memetic algorithms usually the search goes through hundreds or generations, where there are tens or hundreds of individuals in every generation, and each individual may take multiple evaluation calls in its own learning or local refinement. These altogether make the computational cost of memetic algorithms very high when dealing with real world problems.

The intention of memetic algorithms is to improve search performance. However, it is not guaranteed, thus often asks for appropriate combination and fine tuning in practice. The hybridation may attain advantages of both global and local search thus improve, also it is possible that the two ingredients do not fit each other so well and result in lower efficiency. Besides optimistic expectation, it is necessary more solid work on how to understand, measure, control and improve the efficiency in the hybridation algorithms.

So far there has been many reports on the topic, to accelerate memetic search. Basically there are two categories. Some work on the combination of the two components [3, 5, 9, 15, 21, 29–31, 39–42], that is to optimize or adapt the parameters or combination forms to improve the balance. Some work on the extension of accelerating methods already discussed in evolutionary algorithm context [6, 22, 23, 26, 48, 56], and tailor them to suit memetic computation.

However, most of these works focus on Lamarckian algorithms. The difference in inheritance of learned traits distinguishes Lamarckian and Baldwinian evolutions on many aspects, and the roles of learning in both types are quite different. Conclusions of Lamarckian evolution may not hold in Baldwinian evolution. When we tried to apply Baldwinian evolution in real world applications, we find indications in the literature are not adequate. Therefore, we investigate the efficiency characters of Baldwinian evolution, for the ultimate goal of making appropriate applications.

## 1.4 Contents of Thesis

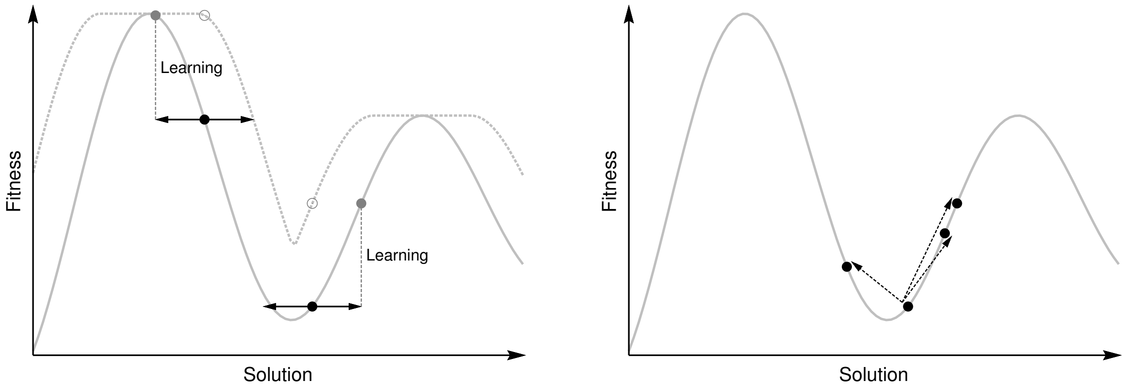
Basically, studies on Baldwinian evolution may have two goals: to provide evidence for biologists, or/and to solve optimization problems. Many researchers focus on the the former topic, and have collected valuable results. As the natural world is too complex and examinations on real creatures are very difficult, these results contribute much to biology. There are also application attempts, but, not many, and some of which are not quite successful. Baldwinian evolution is complex and difficult, and inheriting no learned traits may make its cost performance low. However, it is suitable for dynamic environments, and is likely to be a good choice under some situations.

This thesis contributes to the latter purpose. We study the composing factors of Baldwinian algorithms, and try to find rules concerning search efficiency. Our studies focus on the basic components on static fitness landscapes, however, the ultimate goal is to apply Baldwinian evolution in the real world, in dynamic or/and noisy problems. To build higher towers, we first enhance the foundation.

Our methodology is to present a new view point. The history shows that proposing a new view point is risky, but may bring more knowledge. Nicolaus Copernicus proposed heliocentrism against Ptolemy's geocentrism, and started the astronomy revolution. His key change is the center of the system, not the different-epicycle system [28]. David Hilbert insisted formalism, proposed a program to rebuild the foundations of mathematics, and failed. However, a product of this program, Gödel's incompleteness theorems, brought a revolution to mathematics and philosophy. Human's knowledge is accumulative. A new view point on the foundation enables consequent new ideas and new tools. On the other hand, when we do not start the first step, we can barely see the steps behind. I would like to present my own view point, and some of its consequents.

Our unique method of analyzing Baldwinian evolution is taking the microscopic view. In conventional studies, researchers focus on the global behaviors of the whole population. Of course, this is necessary. Evolutions are on groups or species. However, the group's power comes from diversity. The evolving population can achieve good search because there are different individuals and they form a gene pool. Furthermore, the mechanisms of Baldwinian evolution are very complex. Taking only the global view, we see many phenomena, find some reasons, but cannot figure out how it works in details. We investigate microscopic behaviors from individual to individual, comparing with other individuals of the same generation, or with the parent in the previous generation.

Fig.1.8 shows an example. In conventional studies, Baldwinian learning's enhancement in



(a) Smoothing effect

(b) Individual difference

Fig.1.8: From macroscopic to microscopic.

evolution is described as an equivalent fitness landscape, drawn in the dashed line in Fig.1.8(a). However, this is a simplification that considering all the individuals taking the same deterministic learning process, and a starting point is mapped to a single learning destination. In algorithms, learning usually has randomness, and the destination may vary, thus there is a problem for the child to re-discover its parent's achievement (Fig.1.8(b)). In this case, the equivalent landscape is highly dynamic and noisy, and the metaphor does not provide much convenience. In order to understand the mechanisms, we have to compare children's learning processes to those of their parents. We conducted all our studies in this manner.

For the rest of this thesis, in Chapter 2 we discuss the catching-up behavior in Baldwinian evolution, by comparing children's fitness improvement and that of their parents. Chapter 3 introduces a study on the realization of inherited learning potential, by comparing children's and their parents' learning steps. Chapter 4 presents experiments examining learning cost penalties' effects, by analyzing selection winner types in the population. Chapter 5 presents some discussions, and Chapter 6 is the conclusion. Finally we present an appendix about some experiments on continuous optimization benchmarks, to enhance our conclusions described in the main body of the thesis.

# Chapter 2

# Learning Potential

Baldwinian and Lamarckian evolution are different in inheritance, and this makes their learning processes playing significantly different roles in the whole search. Considering computational resources, the cost-performance also varies. In this chapter, we investigate Baldwinian evolution's mechanism, in term of computational costs and fitness improvements, to find what is produced in Baldwinian learning, and what role Baldwinian learning plays. We found that, on the static landscapes involved, learning cost is paid to maintain a certain level of potential to reach good solutions, rather than to further explore on the landscape. Plasticity codes in genotypes can help in selecting appropriate parts to refine and improve search performance. However, this improvement remains limited because no learned traits are passed on, and does not enable exploration far beyond parents. Some further experiment results in continuous optimization are presented in the appendix.

## 2.1 Introduction to the Chapter

Memetic algorithms are extensions of evolutionary algorithms, inspired by the biology's hold that individual organisms often tailor themselves within their lifetimes to fit the environment. Memetic algorithms combine population-based evolution and individual refinements, a process also called "learning", have attracted increasing attention. Local refinements can, in the context of individual learning, reportedly improve search efficiency. Hybrid algorithms may also, however, perform worse. The individual learning process is often expensive in computation. A single evaluation call may take minutes to hours [47]. Memetic search often continues through hundreds of generations, with tens or hundreds of individuals in every generation, each making multiple evaluation calls in its own learning or local refinement. Altogether these may make the computational cost of memetic algorithms very high when dealing with real-world problems. Only when learning processes are more beneficial in improving fitness than in computational cost can they accelerate the search.

Individual learning often takes multiple evaluation calls, and learning generally takes much more computational effort than evolution, so work has gone into limiting the number of evaluation calls in learning, or preventing some individuals from learning [3, 15, 29, 31, 39]. In some further works, memetic algorithms have been designed self-adaptive [30,40–42]. The parameters such as learning intensity and frequency are tuned online, to attain more flexibility.

Work mainly has been focused on Lamarckian learning, the most widely used pattern encoding learned traits back to genotypes and passing them on to the offspring. need exists, however, for the other pattern of individual learning, Baldwinian learning, suggested by James Baldwin a century ago [2]. Baldwinian learning selects individuals based on refined fitness while using initial genotypes in reproduction. In computation scenarios, Baldwinian evolution is often regarded as a necessary substitution for Lamarckian learning, if the inverse mapping from learned phenotypes to genotypes does not exist, or the environment is dynamic, either of which may make Lamarckian evolution fail. Knowledge pertaining to Lamarckism may thus not necessarily be appropriate in Baldwinian evolution. Specially for Baldwinian evolution, there has been studies on the basic mechanisms [10–12,20,45,51,52], and on search efficiency [5,9,21].

The knowledge accumulated is still not enough to indicate a way to make Baldwinian algorithm designs efficient. In this chapter, we analyze computational cost and fitness improvement in Baldwinian learning, to find some fundamental views. We assign individuals the same learning computation cost, called "budget" hereafter, and compare the refined fitness of them and their parents. We also examine genotypes with and without bit plasticity codes, to find the role plasticity codes play in the scenario.

Note here that we are interested only in the speed to find satisfactory solutions, so genetic assimilation, the second phase of the Baldwin effect, is not considered. In genetic assimilation, individuals rarely achieve even higher fitness and traits become fixed into genotypes. Learning cost falls and the best fitness found changes little, having much less effect on efficiency than the first phase. Learning cost is not considered in selection in experiments, because it is considered trivial in the first phase, and also in experiments, individuals invariably cost the same.

Having various learning budgets and cost punishment in fitness function, even if considered trivial, can bring in the power of selection, and balance cost and performance. However, each possible cost punishment term has its unique way of balancing, based on the form and scale: linear or quadric, big or small. The balance changes search target substantially, from "pursuing high fitness" to "pursuing balanced cost and fitness," and may lead to a unique behavior of convergence. To find a good cost punishment term is thus another question and, in this chapter, we decide not to include its effect in the current step and, instead, focus on lost learned trait information in Baldwinian evolution.

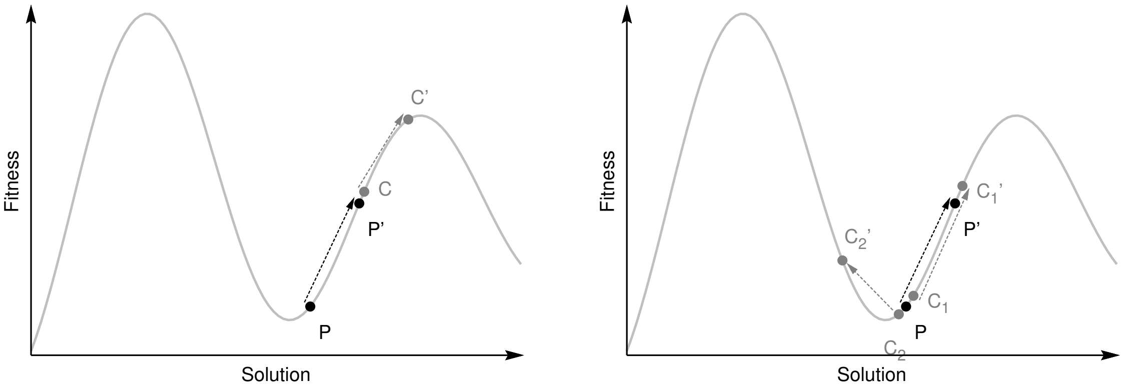
Section 2.2 presents a simple analysis of the mechanisms of Baldwinian and Lamarckian evolution. Section 2.3 introduces experiment settings of Baldwinian/Lamarckian evolution and plasticity codes. Section 2.4 details experiment results, which section 2.5 discusses further. Section 2.6 lists conclusions and future work. In addition, the appendix presents some experiments in continuous optimization.

## 2.2 Lamarckian and Baldwinian Learning

Both Baldwinian and Lamarckian evolution apply individual learning to improve fitness and advantageously affect selection. The difference is whether traits attained in learning are inherited.

Lamarckian theory claims that learned traits can be inherited by the offspring. Based on current knowledge, it substantially asks for the inverse mapping from phenotypes to genotypes not occurring in organisms, and thus not correct in biology. In evolutionary computation, however, such mapping is often easy to attain, and phenotypes may even be the same as genotypes. Lamarckism is simple to implement and brings refinement information directly to evolution. As shown in Fig.2.1(a), individual $P$ undergoes learning and reaches $P'$ with higher fitness. Child $C$ is then generated around final position $P'$, receiving all productions of parental learning, starting with higher fitness and learning to explore beyond.

Baldwinian learning is relatively complex, as shown in Fig.2.1(b). Individual $P$ pays works to learn, refines fitness to some extent, and receives a corresponding bonus in selection. By the



(a) Lamarckian learning

(b) Baldwinian learning

Fig.2.1: Evolutionary search with Lamarckian and Baldwinian learning.

simple fact of survival, all that individual $P$ can tell its offspring is its potential to reach high fitness, not how to learn or where. Its offspring are born similar to $P$ and worse than $P'$, having to work to first catch up to their parents, before searching further. They learn similar to their parents similar to child $C_1$, or possibly take another way similar to child $C_2$, either better or worse than their parents' way.

Taking a different approach, Baldwinian evolution is usually recognized as eliminating inverse mapping from Lamarckian evolution. It can also be considered a traditional evolutionary process with a different selection standard. What provides advantages in survival is the potential to achieve such fitness through the certain learning process, not having high fitness traits directly encoded in genotypes. Offspring also receive potential. Realizing this potential, however, requires nonstop computational effort, even if learning is often similar to that of an ancestor.

Search efficiency is an important factor in many applications, and the principle of evaluating whether an individual learning process enhances the whole search, should be the improvement between sequential generations, not fitness improvement within a generation, Lamarckian evolution does not need to distinguish between the two ways because learning is usually beyond that of the parents. Baldwinian evolution, however, children must catch up to parents, which makes the two ways quite different. This implies a possible reason for Baldwinian learning inefficiency: most computational cost in learning is paid in following in the footsteps of ancestors and maintaining potential. Fitness arises through generations with margins similar to simple evolution, although computation cost is much greater. This in turn, implies a further problem: the need for the learning budget to be constant throughout generations. A varying budget may otherwise become insufficient for maintaining potential, or inefficient in searching further.

A further question arises about plasticity encoding. In conventional Baldwinian evolution studies, genotypes can also include codes referring to which bits or parts are "plastic," in addition to encoding for initial phenotypes. Only plastic parts can be modified in a lifetime of learning. Plasticity encoding can limit the local search range and prevent the search from being inefficient, because the codes also face selective pressure. Plasticity codes are not seen in Lamarckian evolutions because learning goes beyond the previous, and there is no guarantee whatsoever that previous "good" local search directions will remain so. In Baldwinian evolution, the fact that children follow parents brings to bear the significance of plasticity encoding, because children start learning in situations similar to those of parents. Plasticity codes, as a consequence, are expected to bring advantages in a search, but not likely to break the need to follow before pushing the learning of children beyond that of the parents.

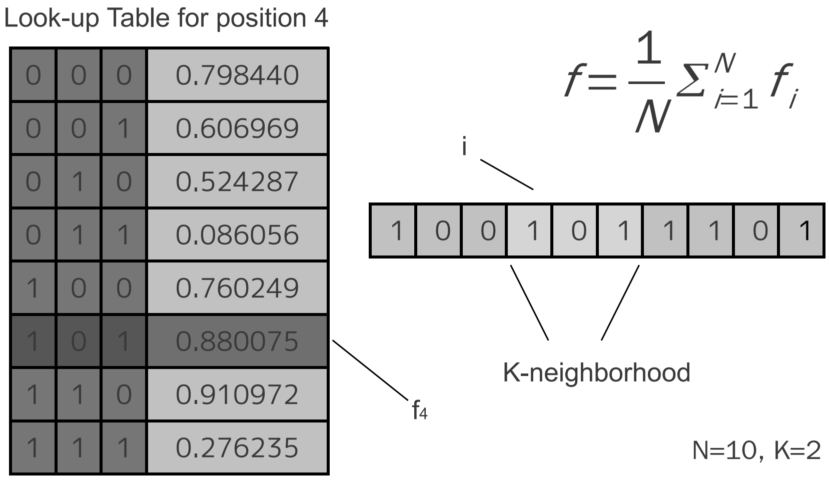
## 2.3 Model

### 2.3.1 NK Model

To test our proposal, we conducted experiments with an artificial NK model, broadly used to simulate general discrete optimization problems proposed by Kauffman [25]. Some researchers use this model to test general rules in evolution [34].

With the NK model, solutions are expressed as binary strings. A landscape is defined by $N$ lookup tables, each of which corresponds to a position of the binary strings. Each lookup table holds information on a position's $(K+1)$-bit neighborhood in the solution string, and lists partial fitness values $f_i$ assigned to $2^{K+1}$ possible binary instances of the $(K+1)$ substring. Then, to calculate the fitness $f$ of a solution, the associated partial fitness values are acquired from the $N$ lookup tables and the average is calculated: $f = \frac{1}{N}\sum_{i=1}^{N} f_i$. $N$ therefore determines the scale of search space in the model, and $K$ adjusts the epistasis/complexity level. Fig. 2.2 shows an example NK model, where $N = 10$ and $K = 2$. Thus, the landscape has 10 lookup tables, the 4th of which is shown here. This table determines the 4th partial fitness, and is indexed by the binary values of the 4th to 6th bits in the solution string. Here, the three bits of concern are 101, and the partial fitness $f_4 = 0.880075$. In our experiments, values in the lookup tables are generated randomly according to a uniform distribution on $[0, 1]$ in order to produce relatively "general" conditions for our comparisons.

To study the effect of uncertainties, we designed two learning schemes: deterministic and non-deterministic. Deterministic scheme modifies bit one by one from most to least significant, and non-deterministic scheme modifies random bits. These schemes are all bitwise climbing trials that start from solution $S_0 = G_I$ and produce solutions $S_1, S_2, \cdots$. An example of the



| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0.798440 |
| 0 | 0 | 1 | 0.606969 |
| 0 | 1 | 0 | 0.524287 |
| 0 | 1 | 1 | 0.086056 |
| 1 | 0 | 0 | 0.760249 |
| 1 | 0 | 1 | 0.880075 |
| 1 | 1 | 0 | 0.910972 |
| 1 | 1 | 1 | 0.276235 |

$$f = \frac{1}{N} \Sigma_{i=1}^{N} f_i$$

Fig.2.2: Fitness calculation of NK model with N=10, K=2.

$k$-th trial in a learning process is shown in Fig. 2.3. On the basis of previous result $S_k$, the position $i = 4$ is selected and the $i$th bit in the solution is modified to produce $S'_k$. $S'_k$ is then compared with $S_k$ and the fitter of the two is set as $S_{k+1}$. Such trials are repeated for different modifying positions until a given number of trials have been conducted. The final solution is then the learning destination and its fitness is set as the fitness of the individual.

### 2.3.2 Plasticity Codes

The above explanations apply to Baldwinian and Lamarckian evolution without plasticity codes, where in all bits are considered plastic. For comparison we include a version of Baldwinian search that uses plasticity codes to show how plasticity inheritance works in the search.

The extended genotype with plasticity is shown in Fig.2.4. $G_I$ records the learning start point, as explained. Additional binary string $G_P$ points out which bits in $G_I$ can be modified in lifetime learning. In the learning process, only the plastic bits, those having 1 in $G_P$) of $G_I$, are modified to try new strings and pursue higher fitness. The only difference in learning is that when determining the swapping bit, only candidate bits, not all bits are plastic according to $G_P$.

### 2.3.3 Experiment Settings

Experiments use two-tournament selection, a simple two-point crossover and a one-bit mutation. Parameters lists Table 2.1.

Three groups of experiments are conducted separately with the model on two landscapes. Group 1 compares Baldwinian and Lamarckian learning, i.e., how the learning budget affects

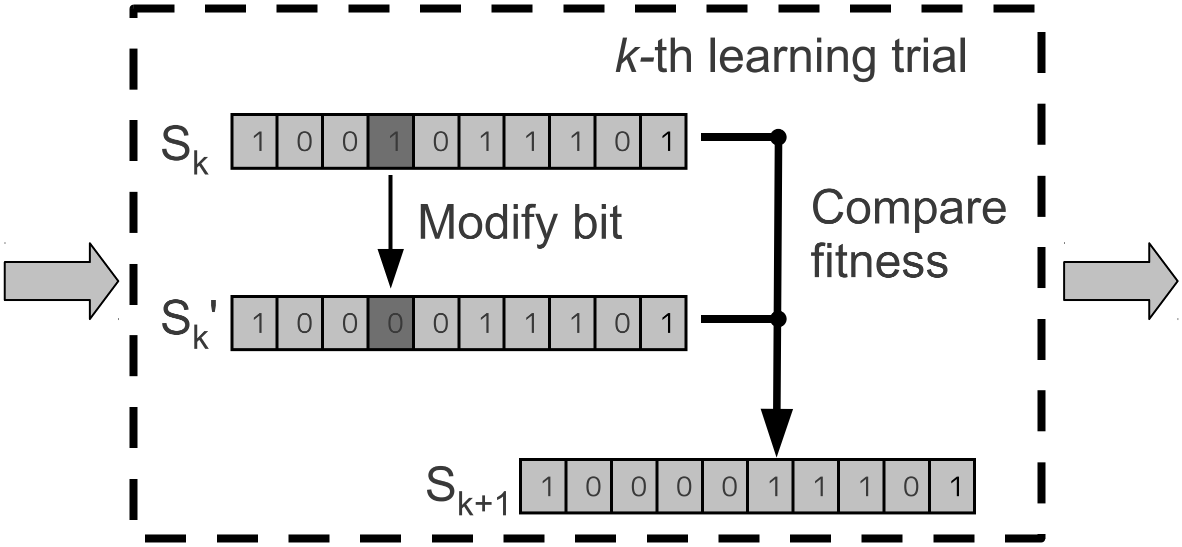


Fig.2.3: $k$-th trial in learning process.

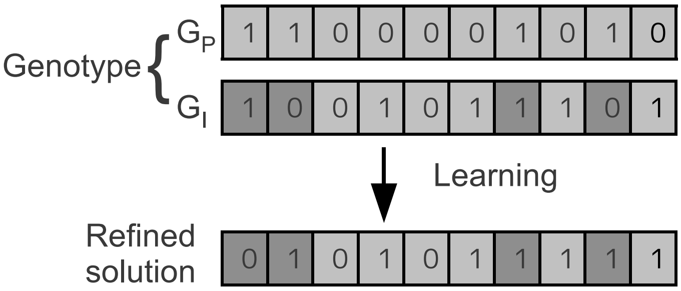


Fig.2.4: Genotype with plasticity codes.

Table2.1: Experiment Parameters

| | |
|---|---|
| N | 20 |
| K | 2 / 5 |
| Population Size | 500 |
| Crossover Rate | 0.50 |
| Mutation Rate | 0.05 |
| Iteration Runs | 50 |

the scale of improvement between generations. A test of how much improvement each step brings in Baldwinian learning is then made. A Baldwinian search process with a periodically changing learning budget then shows that budget changes may cause additional inefficiency. Both versions, with and without plasticity, are examined for all Baldwinian evolution runs.

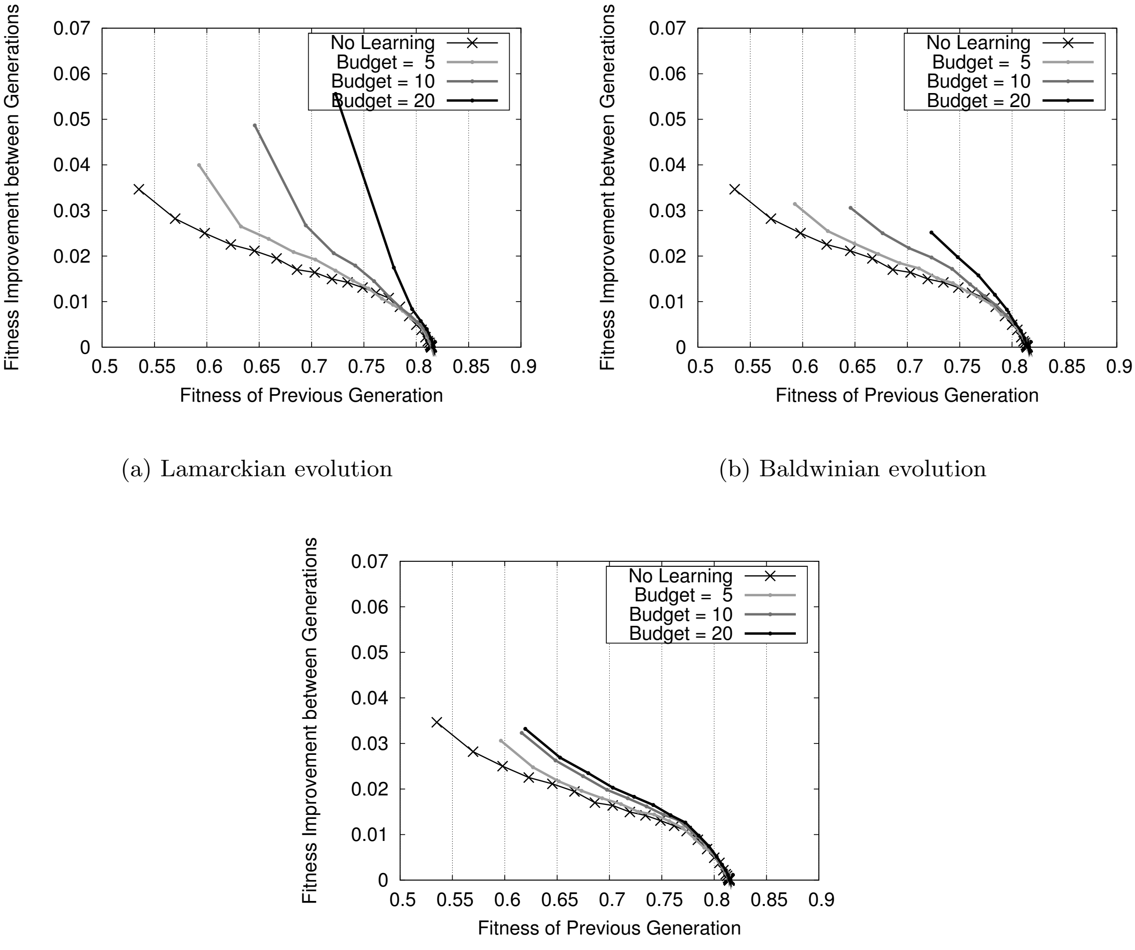
In this chapter we use only two landscapes with $K = 2$ and $K = 5$. The $K = 15$ landscape, which will be used in latter chapters, is not included here. On the $K = 15$ landscape, things are not changed, but differences become less distinct because learning is very difficult to follow. On the other hand, we already have comparisons of two learning schemes and three values of learning budgets. It is possible that an additional landscape and additional 50% data makes the explanations more complicated rather than clearer. We decide to present results of only two landscapes here.

## 2.4 Experiments and Results

In an early generation of a memetic search, individual learning usually does not lead directly to the global optimum, We are often not even sure if there is a better solution undiscovered. Given this, the benefit of learning is improvement expected in later generations rather than a rise in yhe current generation, so we choose average fitness instead of best fitness as the score. In Lamarckian evolution, any benefit from learning is apparently fully inherited by the offspring. Benefits passed on to the offspring are thus almost the same as in fitness improvement during current learning. In Baldwinian evolution, however, inheritance is not direct benefit. It is thus worth determining to what extent learning achievements are passed on or, in the sense of computational efficiency, to what degree Baldwinian learning pushes the whole search into going further. Experiments all deal with this topic, and some fundamental propositions can be inferred from results.

### 2.4.1 Improvement Through the Generations

In Baldwinian evolution, the learning process can improve fitness through the individual's lifetime, but what the offspring actually receive is doubtful. In these experiments, exactly the same learning processes are applied in Baldwinian and Lamarckian evolution for comparison. The learning budget, i.e. the number of learning steps for each individual, is fixed through each run, and various budget scales are tested. What is recorded and compared is the difference between the average final fitness of the current generation and that of the previous generation. We are, in fact, measuring what is brought into the current generation rather than what is passed on to the offspring. In Baldwinian evolution with plasticity, learning schemes do, in fact,



(a) Lamarckian evolution

(b) Baldwinian evolution

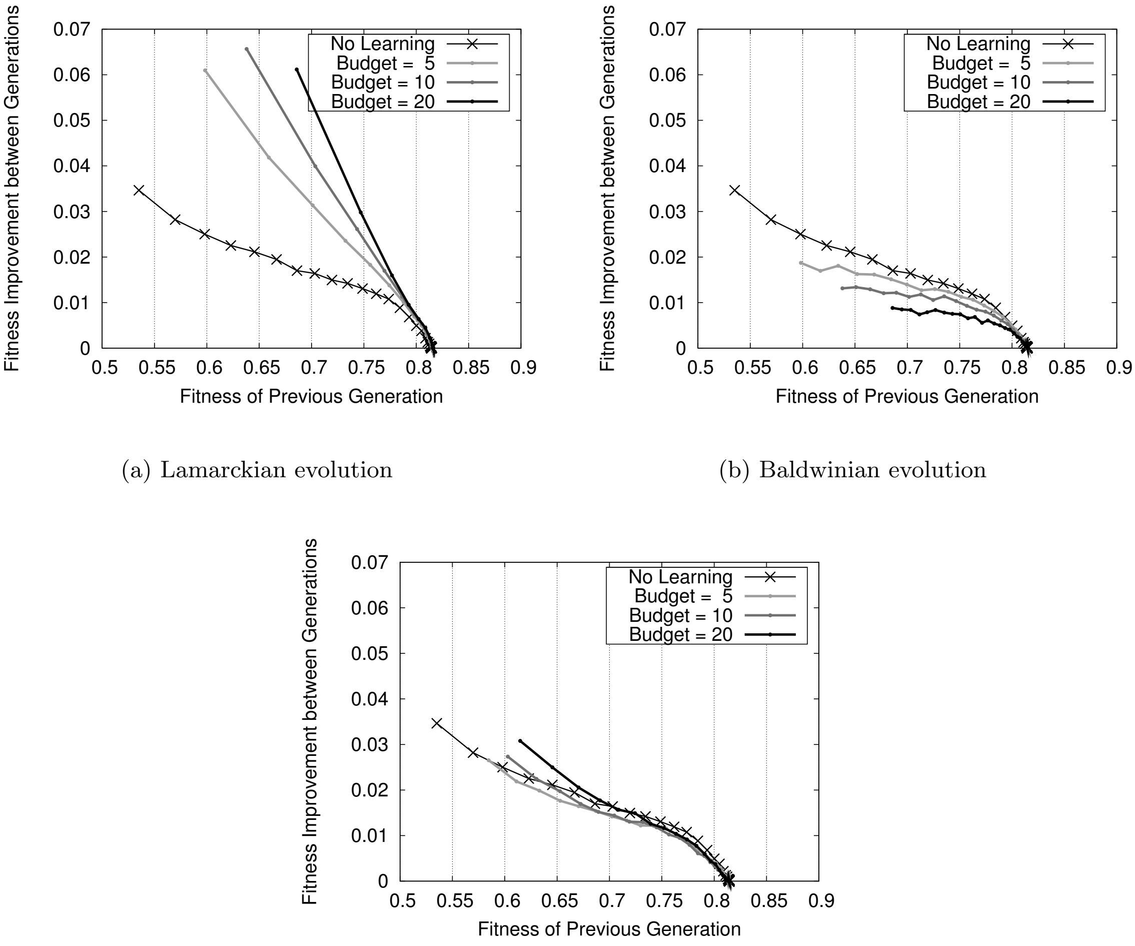
(c) Baldwinian evolution with plasticity

Fig.2.5: Fitness improvement with deterministic learning on the K=2 landscape.

different due to the change in candidate bit sets. This is, however, the very way to compare and judge the effect of plasticity codes.

Relatively low fitness is naturally easier to improve further, so different evolution versions should be compared for similar previous fitness levels. We use the average fitness of the last generation as index x, and the average fitness differences of two generations as index y. Data shown is for averages of 50 random runs.

Fig.2.5 shows the result of using a deterministic local search, on the K=2 landscape, where epistasis is low and optimization is relatively simple. Results show that improvement between generations in Baldwinian evolution is much lower than that in Lamarckian evolution. The little advantage of no learning can be observed, and as the learning budget increases, learning passes on a bit more benefit to the offspring. Adding plasticity codes keeps fitness improvements



(a) Lamarckian evolution

(b) Baldwinian evolution

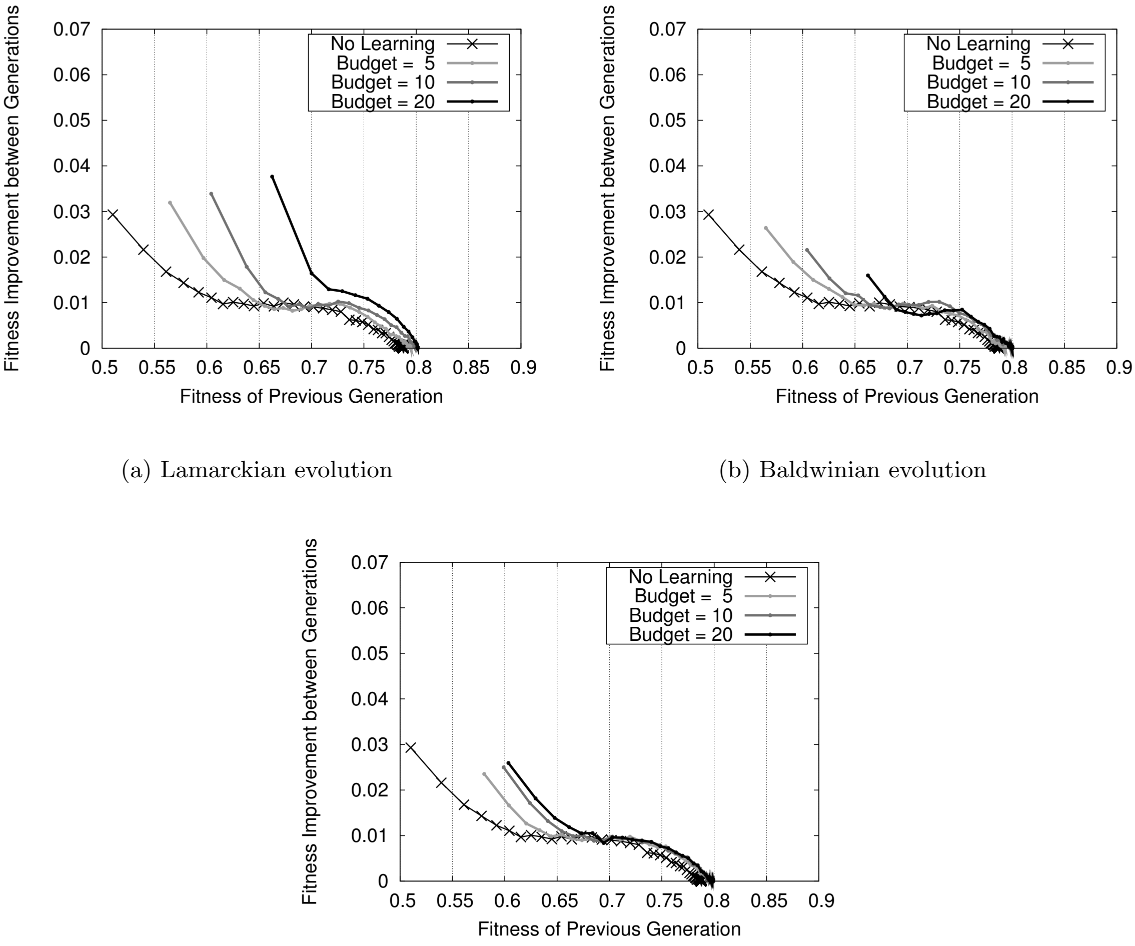
(c) Baldwinian evolution with plasticity

Fig.2.6: Fitness improvement with nondeterministic learning on the K=2 landscape.

almost the same, or even a little lower.

Fig.2.6 shows results of nondeterministic local search, on the K=2 landscape. Note that with Lamarckian learning, the improvement between generations is even higher than with deterministic learning. Uncertainties appear to offer more chances and Lamarckian learning has managed to catch them. In contrast, Baldwinian evolution performs even worse than no learning. As the learning budget increases, it becomes more and more difficult to realize and the computational cost grows even as improvement through generations falls slightly. With plasticity codes, the Baldwinian evolution becomes better than no learning whatsoever, but still only slightly.

Figs.2.7 and 2.8 show results on the K=5 landscape, a landscape that is more complex, and has higher epistasis. With deterministic learning scheme, Lamarckian evolution improves most, and both Baldwinian versions improve just a little more than no learning evolution. This is similar



(a) Lamarckian evolution

(b) Baldwinian evolution

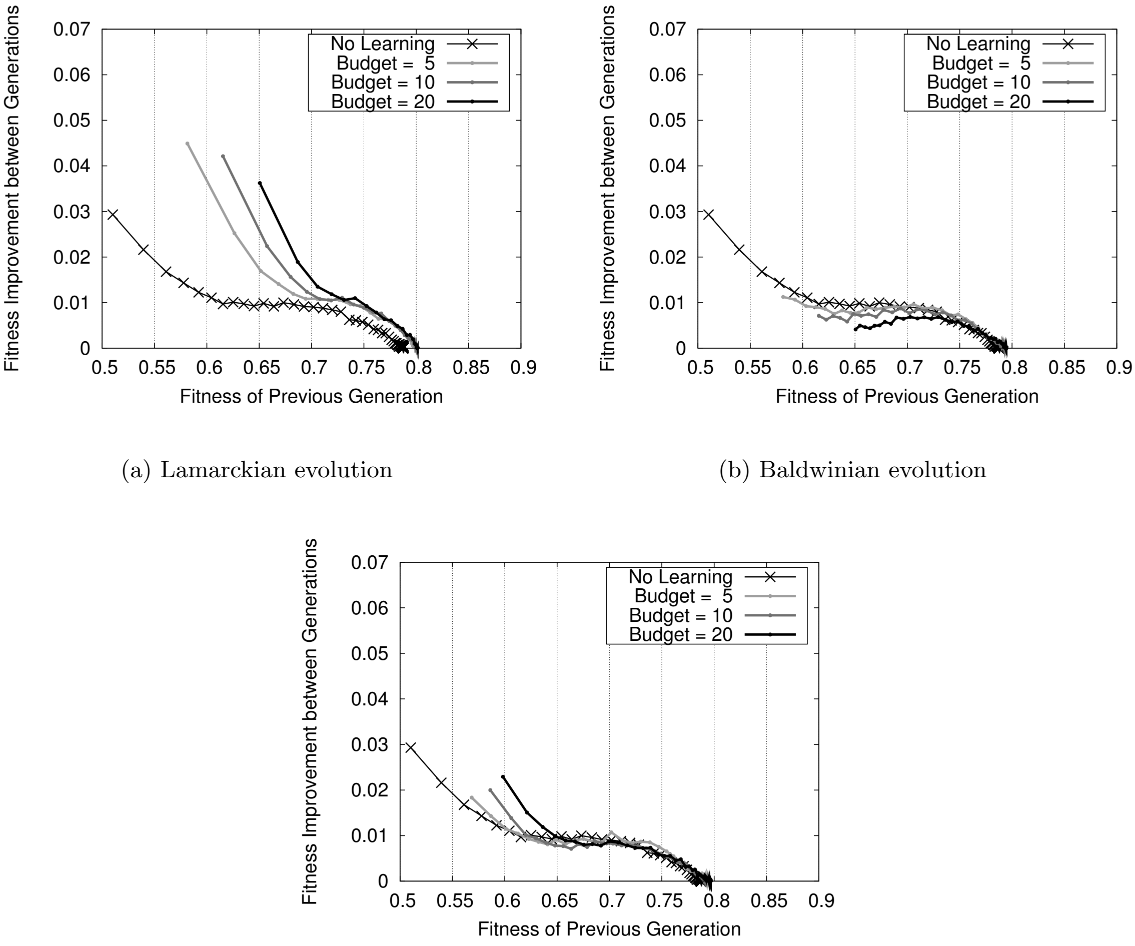
(c) Baldwinian evolution with plasticity

Fig.2.7: Fitness improvement with deterministic learning on the K=5 landscape.

to the situation for the K=2 landscape, but has smaller differences between the algorithms. With the nondeterministic learning scheme, things are similar to that on the K=2 landscape. Another thing common to this landscape is that after fitness reaches a certain level 0.7 here, all the algorithms perform similarly in fitness improvements. This is related to landscape features.

In the experiments above, note that, the Baldwinian learning effort brings little if any advantage in fitness through generations, compared to no learning evolution. Plasticity improves performance, but not essentially. With the same budget as that of the ancestors, the offspring in learning keep just the potential they inherited. Baldwinian learning may bring higher diversity, however, does not seem to accelerate the increase in fitness.

The most important point here is search speed, in fitness improvement through generations. The learning schemes themself are not powerful to improve fitness, leading to fitness converging



(a) Lamarckian evolution

(b) Baldwinian evolution

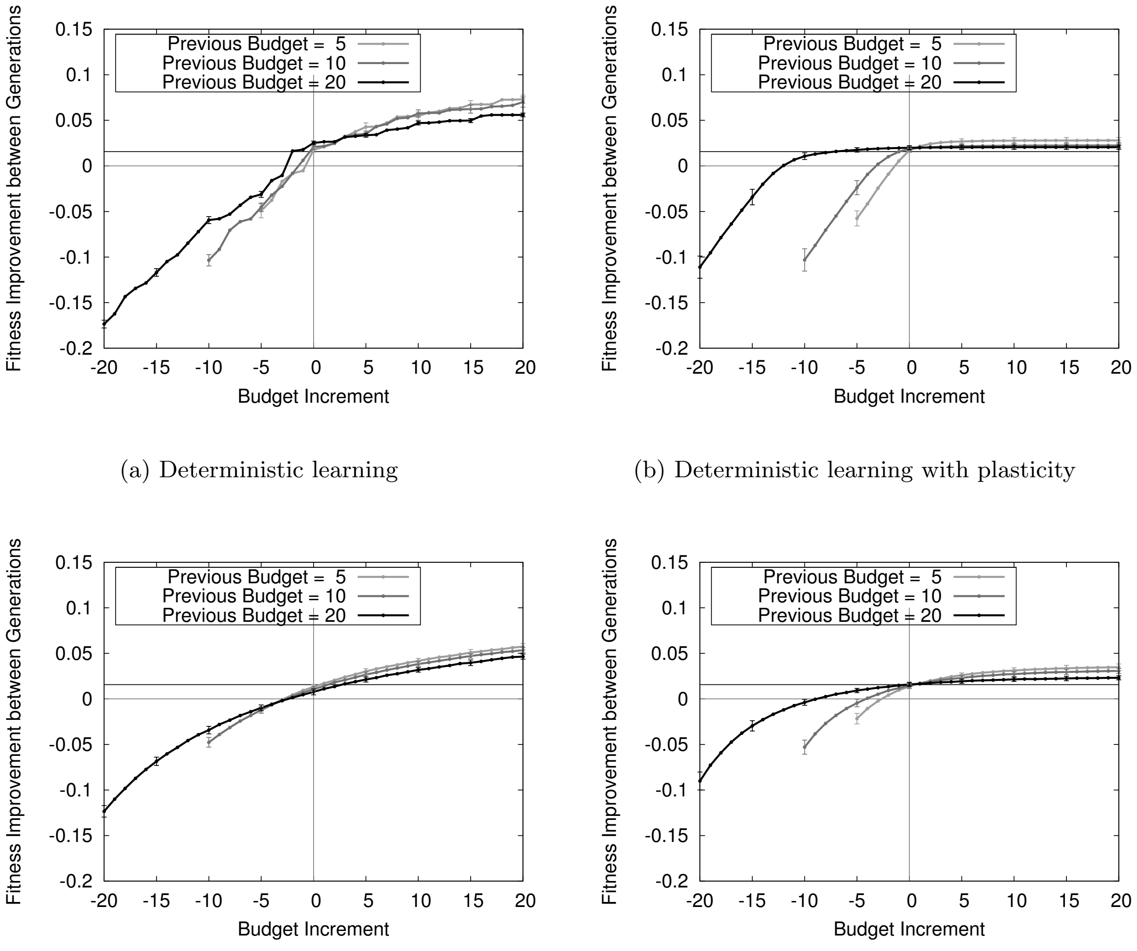
(c) Baldwinian evolution with plasticity

Fig.2.8: Fitness improvement with nondeterministic learning on the K=5 landscape.

above 0.8 with small differences. The difference between Lamarckian and Baldwinian evolution, and between the two learning schemes is significant, however, it would interesting to check more powerful learning schemes, however, that may accelerate the search phase and converge fast, leaving us less data to compare differences in the search phase.

### 2.4.2 Step-by-Step Improvement

With a fixed budget, Baldwinian learning performs to maintain learning potential, rather than pushing the population into going further. The question thus arises: what happens when the learning budget changes through the search process. In these experiments, we first let the Baldwinian evolution run with a fixed learning budget, until the average fitness of the population reaches a certain level set here at 0.70. After that learning is taken trial by trial, to measure



(a) Deterministic learning

(b) Deterministic learning with plasticity

(c) Nondeterministic learning

(d) Nondeterministic learning with plasticity

Fig.2.9: Step-by-step learning improvements on the K=2 landscape.

how much it improves in each learning step, i.e. we determine what fitness level the current generation can achieve.

To compare this to parents and see how inherited potential is realized, we expressed the learning budget differences from the previous budget, not to absolute values. For the same reason, improvement is expressed the differences in final fitness between the two generations, not in the fitness increment during learning. Data is averages of 50 random runs.

Results for the K=2 landscape are shown in Fig.2.9. The abscissa shows the current learning budget comparing to the previous one, and the ordinate shows fitness improvement since the last generation. The horizontal line at y=0.0156449 shows on the average how much it improves if no learning evolution is applied. Deviations in fitness improvement at some points are shown by error bars.

First it is clear that, with either deterministic or nondeterministic learning, with or without plasticity, only when the current learning budget approaches the previous budget does the current generation catch up with the previous generation in fitness. When the same budget as used before is taken, the fitness increment of the current generation is very close to that of no learning evolution. This is again evidence that Baldwinian learning's main job is to maintain potential and to catch up with parents. It can be further inferred that when the learning budget drops, the population loses some of the potential inherited, causing the generation to stop or even back up in fitness.

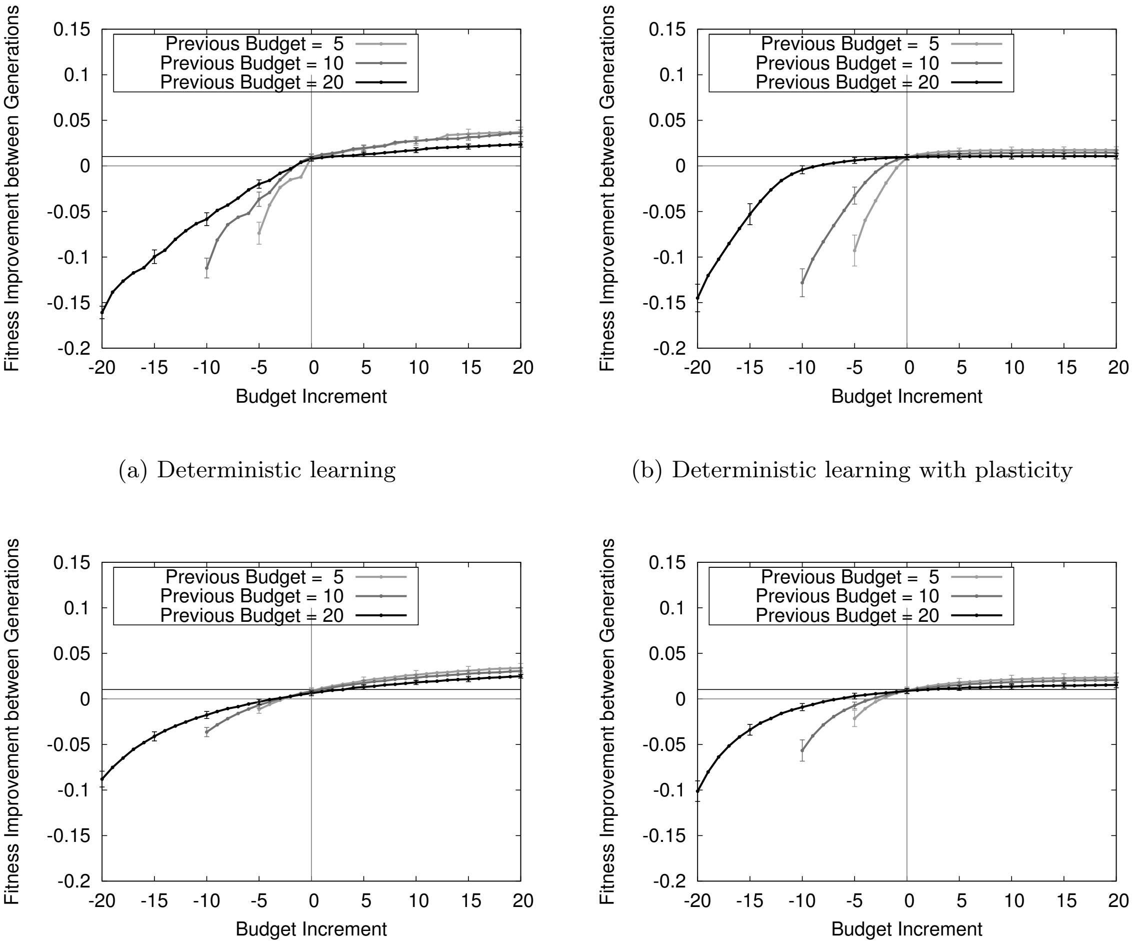
Inherited potential, on the other hand, appears to be only the potential to reach high fitness in the fixed budget, with learning steps beyond realizing inherited potential becoming inefficient. Note in Figs.2.9(b) and 2.9(d), for Baldwinian learning with plasticity, that after taking the same budget (increment=0), further steps make hardly any progress. In Figs.2.9(a) and 2.9(c), the non-plasticity versions, this phenomenon becomes less obvious, but the difference in fitness increasing speed is apparent, comparing periods before and after reaching the previous budget. Plasticity codes thus appear to make the budget limit stricter.

Results on the K=5 landscape are shown in Fig.2.10. The horizontal line at y=0.0102983 shows on the average how much it improves if no-learning evolution is applied. Similar to the situation for K=2, the same computational cost is paid to achieve similar improvements as in no learning evolution. As landscape complexity increases, it becomes more difficult to maintain potential and explore unknown areas.

These results verified again learning cost is paid to keep inherited potential, implying that adaptating learning budget may bring inefficiency into a search. When the budget drops, some inherited potential is not realized and is hence lost, rendering some effort useless. Even when the budget rises again afterward, a less efficient local search will be made to rediscover potential, slowing the whole search even more.

### 2.4.3 Changing the Budget

In the search of a memetic algorithm, learning intensity/budget is an important performance parameter. Some studies on this in Lamarckian evolution may lead to claims that techniques used in Lamarckian evolution can also be applied to Baldwinian evolution [40]. These experiments argue that this is not easy, because what is inherited is indirect potential corresponding to the learning process, not direct fitness. The difference in inheritance lets Baldwinian learning affect even the performance of later generations, and it is inappropriate to adapt a budget from outside of evolution. Such adaptations break inherited potential and prevent children from realizing what the parents achieved, further adversely influence search performance.



(a) Deterministic learning

(b) Deterministic learning with plasticity

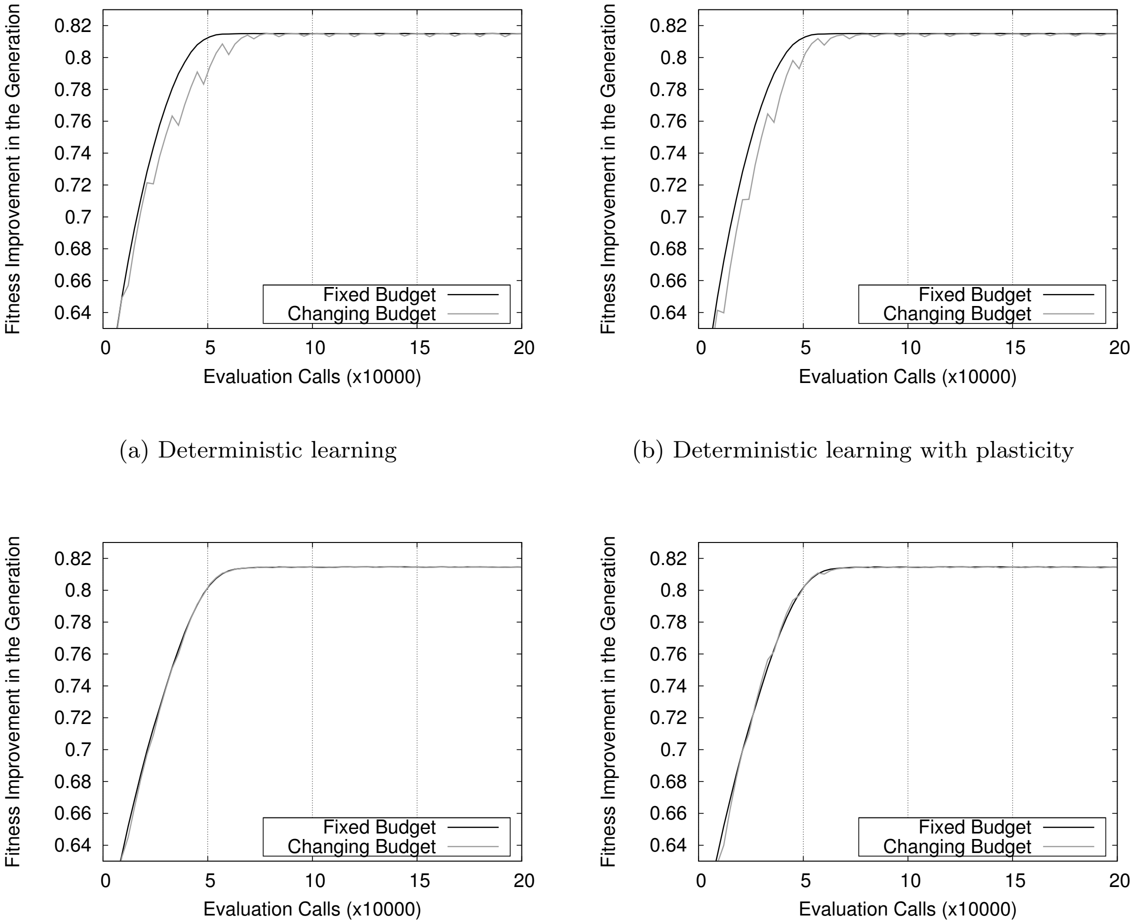
(c) Nondeterministic learning

(d) Nondeterministic learning with plasticity

Fig.2.10: Step-by-step learning improvements on the K=5 landscape.

Generally speaking, a certain learning scheme makes the inheritance a certain level of potential, to which it is related to but not always in accord with other levels of potentials. The potential to reach a good solution after 10 learning steps does not guarantee that a point achieved by only 5 steps is a good one, or that after another 10 steps a prior solution can be found comparing to other individuals taking 20 steps. The individuals are selected for 10-step learning.

From this, changes in a learning budget may break potential and slow the search in a Baldwinian scenario. In experiments, we set a scenario to verify this effect. Experiments test two budget assignment cases: one using a fixed learning budget of 5 steps, and the other using 1 step and 9 steps alternatively, changing every 4 generations. This means that over a long period, the average generation computational cost of the two methods is exactly the same. Baldwinian



(a) Deterministic learning

(b) Deterministic learning with plasticity

(c) Nondeterministic learning

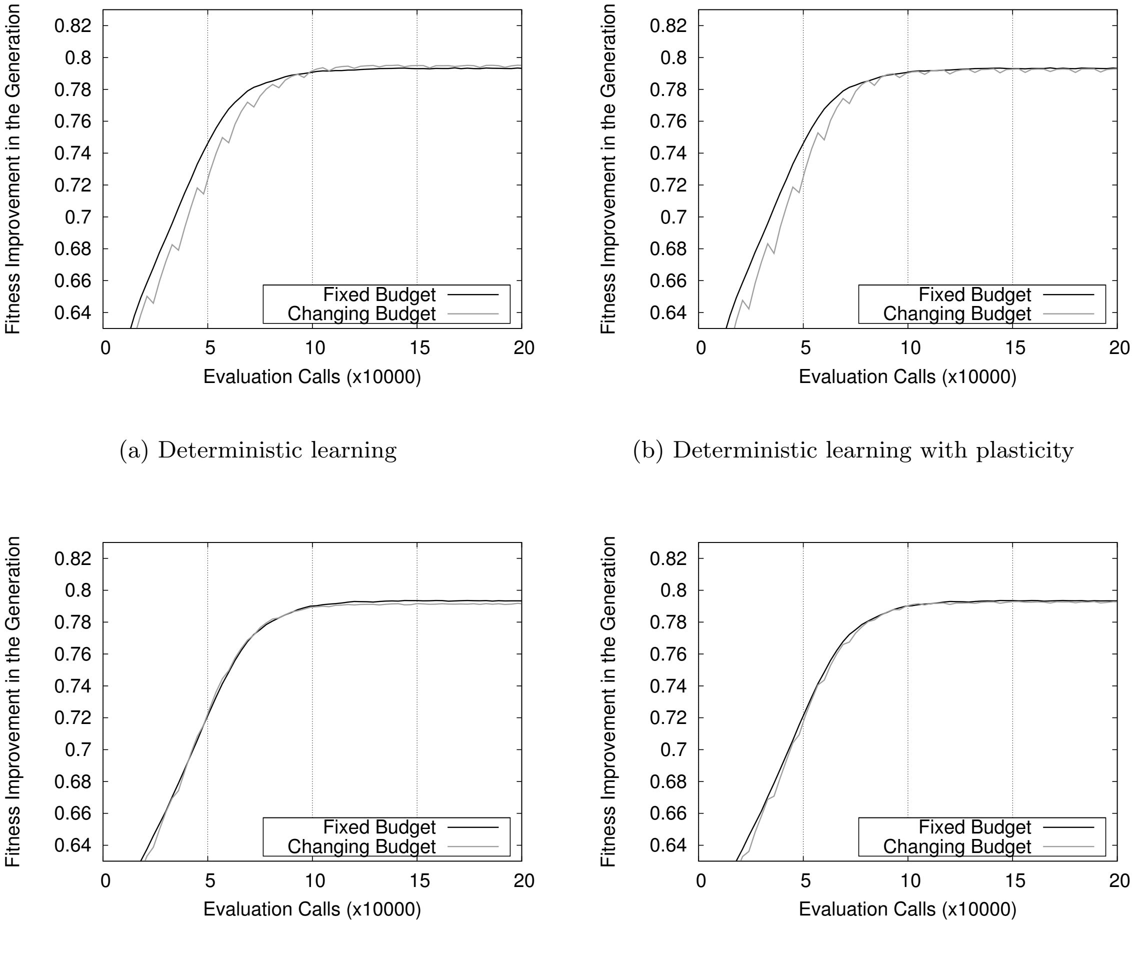
(d) Nondeterministic learning with plasticity

Fig.2.11: Periodically changing and constant budgets on the K=2 landscape.

versions both with and without plasticity codes are included.

We simply compared the search speed of the two methods. Data is averages of 50 random runs. Fig.2.11 shows results on the K=2 landscape. Figs.2.11(a) and 2.11(b) show that with deterministic learning, search performance is adversely affected by the changing learning budget: a budget decrease causes fitness to drop in the next generation. In later generations it pursues the fitness of the fixed budget version, but can hardly catch up with it, even in next 4 steps of a high learning budget. Taking averages of 50 runs and 500 individuals per run, the trend is smoothed, but the difference in changing and constant budgets is distinct. With deterministic learning scheme, regardless of whether plasticity is included, the entire search becomes slower if the budget is dynamic.

This does not, however, always happen. As shown in Figs.2.11(a) and 2.11(b), the results of



(a) Deterministic learning

(b) Deterministic learning with plasticity

(c) Nondeterministic learning

(d) Nondeterministic learning with plasticity

Fig.2.12: Periodically changing and constant budgets on the K=5 landscape.

nondeterministic learning do not imply such a phenomenon. Average lines show that changing and constant budgets yield almost the same performance, probably because learning is random and offspring rely less on their ancestors.

The case is similar for K=5, as shown in Fig.2.12. With deterministic learning, fitness with a changing budget drops and rises regularly, finally converging at a level lower than that of the constant budget version. With nondeterministic learning, changing and constant budget versions perform quite similarly, but here the changing budget converges at a slightly lower fitness level.

Experiments thus verify that in Baldwinian evolution, the level of inherited potential is fixed by the allowed budget. Online adaptation of the budget may bring additional inefficiency and weaken the search. It is not a problem in this search that with deterministic learning, the changing budget search converges only a few generations later than the fixed budget search. It is possible, however, that problems may arise when other learning schemes are applied, perhaps with more procedure information or heuristics. Cost is also likely to rise, if the landscape is more complex and the search period much longer, such as tens or hundreds of generations, or the learning budget variation becomes more complex. This makes it necessary to consider this inefficiency when applying adaptations to learning intensity in Baldwinian evolution, or to avoid designing such online adaptations from the very beginning.

## 2.5 Discussion

In this chapter, we revealed a novel view of Baldwinian learning in the stage before convergence: Learning effort is paid to maintain a certain level of potential to achieve high fitness through learning, rather than to spread the search and pursue higher fitness. This presents further understanding of the Baldwinian evolution performance in the search phase.

We also examined the use of plasticity encoding, which is widely used in Baldwinian evolution research, and considered an enhancement. It is reasonable to set some general rules or limitations on genotypes, but plasticity itself becomes practical because children and parents are starting from similar positions and searching almost the same areas. This cannot change the basic fact that learned traits are not known to the offspring, or the result that learning cost is mostly paid to maintain potential. Our experiments also verified this.

This work presents the view that Baldwinian evolution may work this way, rather than to argue that it is invariable in all possible situations. Some work already showed different Baldwinian evolution performances on different landscapes [46, 51, 55]. Baldwinian evolution showd its strengths when the problem is deceptive or the environment is dynamic. In this work we included fewer factors and analyzed performance on very simple static landscapes. The view attained here is not directly useful to design efficient Baldwinian algorithms, nor does it even imply that in such situations, Baldwinian evolution is an undesirable choice for such practical requests. By analyzing the basic facts, however, we can attain an alternative viewpoint that may help in the study of more complex situations.

There are many possible extensions for future works. First of all is the question of what Baldwinian learning achieves. From the examination we found it difficult to exploit fitness after the first generation. It is, however, said to be robust. What the robustness is against, and how the trade-off between the two aspects can be measured, calls for further investigation.

Another question arises about the end condition of learning. In our experiments the end condition is the limit on the learning budget. Another end condition frequently used is the arrival at a local optimum or a threshold for low improvement. We wonder how the cost-benefit relationship in individual learning is to be determined, and how the budget assignment and end condition are to be used to achieve an efficient search.

This then leads to the question about the effect of learning cost in selection. This is conventionally considered to be the condition necessary to enable genetic assimilation. It can be expected that having various learning levels in one generation and putting the cost into selection may help with computational efficiency, as individuals having smaller potential ask their offspring to make less effort in realizing their inheritance. This is, in fact, the most essential part in learning cost that we are concerned with before genetic assimilation. As mentioned in the introduction, questions exist on how to fix the form and scale of this balance, and how the performance will be changed.

Another point important in acceleration is the comparison of efficiency in evolution and learning. Our examination mainly implies that their products are not independent, and that the total effect is not simply the sum of both. In Baldwinian learning the search history should be considered. Chanllenges thus remain in how to make an efficient estimation, how to assign a set of computational resources, and how to optimize search efficiency.

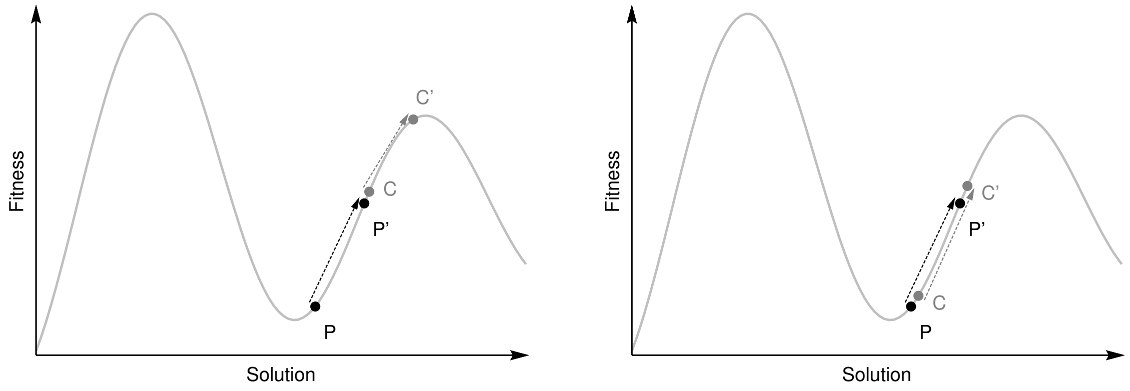
## 2.6 Conclusion

This chapter has, from the view of computational cost and fitness improvement, examined the basic mechanism in Baldwinian evolution in which no learned traits are passed on to the offspring. It is revealed that, on the static landscapes involved, learning cost is paid to maintain a certain level of potential to reach good fitness, rather than to further explore on the landscape. If the level of potential is compromised, for example, by learning intensity adaptation, some inheritance may be lost and the search becomes worse. We also experimented with plasticity codes, finding that plasticity encoding may improve the performance as good areas are selected, but does not change the fact that Baldwinian learning effort is paid to realize inherited potential. There are some further verifications in continuous optimization in the appendix of this thesis, and the results also support the conclusions found in this chapter. These results provide an alternative viewpoint on the basic Baldwinian mechanism, and may be a helpful reference in complex situations.

# Chapter 3

# Realization of Learning Potential

In Baldwinian evolution, refined fitness influences selection. However, unlike Lamarckian evolution, refined traits are not passed on to the offspring. This loss of refined trait information implies that the guiding effect of learning comes from the inheritance of learning potential. Offspring do not directly inherit beneficial traits, but instead inherit genes that lead to beneficial traits after their lifetime learning. In the previous chapter we verified the existence of such potential, and in this chapter we study how it works. The realization of learning potential, namely, how learning behavior compares to the previous generation's learning, are studied, in addition to how much learning improves the current generation's initial phenotypes. Since children are affected by learning schemes and genetic operators, they may prefer either to follow and repeat their parents' learning, or to explore a new direction. In this chapter, we investigate how learning schemes and genetic operators affect the realization of learning potential, and, in turn, how this influences search performance. Some further experiments in continuous optimization are presented in the appendix. The results show that learning schemes and genetic operators have different impacts: uncertainties in learning schemes slow down speeds and lower fitness, whereas genetic operators balance exploitation and exploration. The guiding effect of Baldwinian learning is thus implied to originate from only what children can inherit or follow.



(a) Lamarckian evolution

(b) Baldwinian evolution

Fig.3.1: Two types of memetic search.

## 3.1 Introduction to the Chapter

The biological concept of Baldwinian evolution is especially complicated, as well as sometimes being quite vague [52]. In this chapter, we consider a simplified situation by using randomly generated static landscapes and an evolution process that includes only the most basic rule: "learned traits are not inherited." We then propose a method to explain that Baldwinian evolution's properties may have impact on performance and how this occurs. In particular, we make a comparison between children's and parents' learning processes. This approach differs from many conventional studies that examine learning performance only in the current generation.

The present work is inspired by Suzuki *et al.* [51], who have reported climbing behavior in Baldwinian evolution and the effect of keeping track of the optimum. Here, we study the outcome of an offspring's decision to follow (or not) their parents' learning trajectories or directions, which can be affected by the learning starting point and its process.

Fig. 3.1 shows examples of Lamarckian and Baldwinian evolution. In Lamarckian evolution, a child $C$ completely inherits the refined traits of its parent $P'$. Then, through the child's learning process $C \to C'$, it goes beyond the learning destination of its parent. In Baldwinian evolution, $C$ is born close to $P$, the starting point of its parent. $C$ must therefore first catch up with the final position of its parent $P'$ before it can progress even further.

During this catch-up process in Baldwinian learning, individuals may choose to follow their parents' footprints or a different pathway that explores a new area of the landscape. In this chapter, we study this preference, how it is influenced by algorithm parameters and how, in turn, it influences search performance. Our results reveal that learning schemes and genetic

Table3.1: Experiment Parameters

| | |
|---|---|
| $N$ | 20 |
| $K$ | 2/5/15 |
| Population Size | 500 |
| Learning Budget | 5/10/20 |
| Crossover Rate | 0.50/0.75 |
| Mutation Rate | 0.05 |

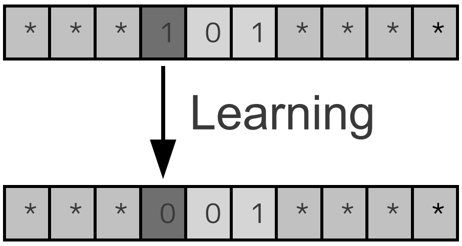


Fig.3.2: Each partial fitness value is dependent on only a single $K$-neighborhood.

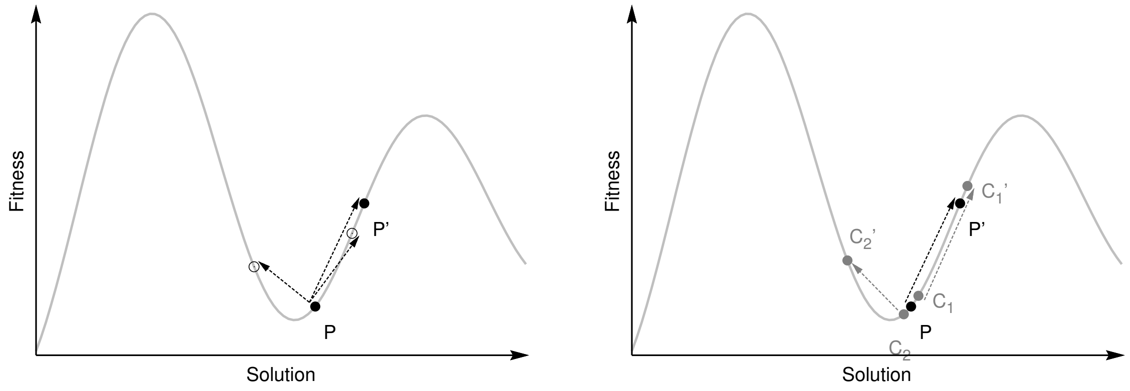
operators have different impacts. Uncertainties in learning schemes degrade speed and fitness, while variation in genetic operators appears as a trade-off between speed and diversity. The chapter proceeds as follows. In Section 3.2, we introduce the NK model as a benchmark, and present the learning schemes and crossover operators involved in our comparison. In Sections 3.3 and 3.4, we then outline the experiment for comparing the learning schemes and crossover operators, respectively. A discussion on the experimental results and possible future work is given in Section 3.5. Finally, in Section 3.6, the conclusions of this study are presented. In addition, some experiments with continuous optimization benchmarks are presented in the appendix of this thesis.

## 3.2 Model

### 3.2.1 NK Model with Plasticity

We use NK model with plasticity codes for benchmark, as introduced in the previous chapter. Conventional 2-tournament selection and one-bit mutation is employed. Learning schemes and crossover operators are introduced later. Some experiment parameters are included in Table 3.1. Experiments are taken on three fitness landscapes, to include various levels of epistasis and complexity. For each landscape the N look-up tables are generated randomly once, and on this same landscape various tests and their iterations take place.

In the experiments, we compare children's and parents' learning processes to investigate



(a) Learning uncertainties

(b) Children's learning

Fig.3.3: Learning uncertainties and effect on children.

whether learning is “similar”. Here, similarity is an imprecise concept. Affected by genetic operators such as crossovers and mutations, children differ from their parents even though they inherit their parents' genes. Learning potential is realized not by children having exactly the same initial points and learning processes as their parents, but by inheriting functional gene pieces that enable the children to perform the same local refinements. Fortunately, the NK model facilitates quantitative measurement of this local similarity. A solution's fitness is the average of $N$ partial fitness values, each of which is dependent on only a single $K$-neighborhood. If, at time $t$, a change occurs in a $K$-neighborhood of a learning individual, and the same change in the same $K$-neighborhood previously occurred for one of its parents, then in the scope of this $K$-neighborhood and learning change, the child's learning is similar to that of its parent. Fig. 3.2 shows an example of this: if the 4th bit is changed from 1 to 0, and the 5th and 6th bits are 0 and 1, respectively then the change in $f_4$ is determined regardless of the other bits in the solution string. Learning similarity in our experiments is measured as the number of such learning changes repeated locally.

The learning schemes and crossover operators for testing are introduced in the next section. Several experiment parameters are listed in Table 3.1, and, 2-tournament selection and one-bit mutation are employed. Experiments are performed on three fitness landscapes to examine various levels of epistasis and complexity.

### 3.2.2 Learning Schemes

When comparing learning processes between parents and children, the most essential factor is the employed learning scheme. If uncertainties exist in the learning scheme, the trajectory

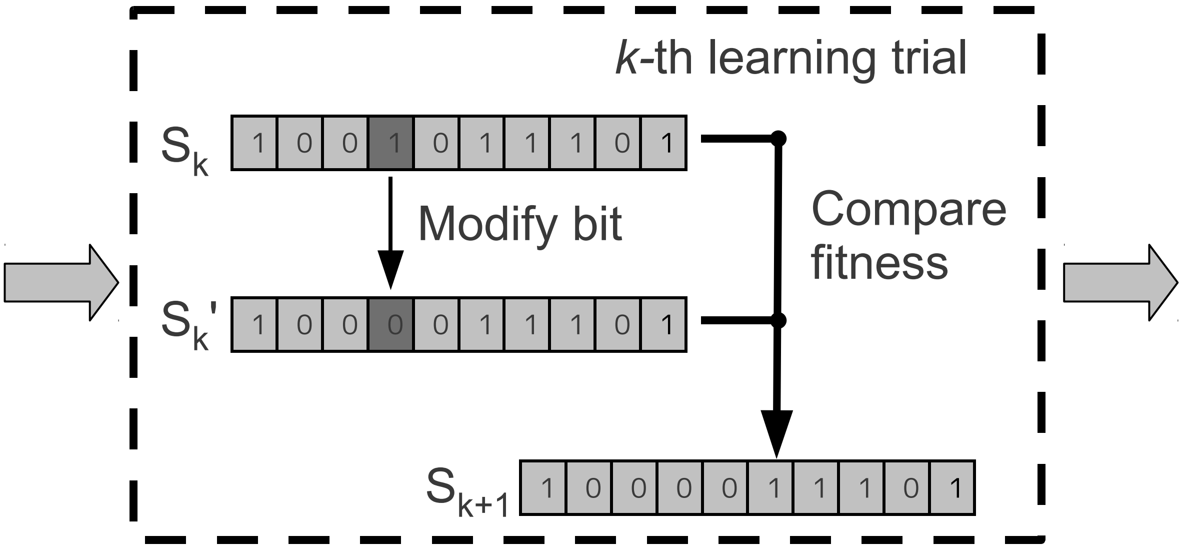


Fig.3.4: $k$-th trial in learning process.

and results of learning may vary even when starting with exactly the same solution. As shown in Fig. 3.3(a), if the starting point is $P$, the destination could be $P'$. However, uncertainties can shift the destination to a point in the same direction as $P'$ but closer to $P$, or even to a point in a different direction. Hence, although children inherit their parents' genes, they are not guaranteed to rediscover their parents' learning destinations. A child may ($C_1 \to C_1'$) or may not ($C_2 \to C_2'$) follow their parent's path (Fig. 3.3(b)). Under different levels of uncertainties, a child preference for following its parents changes. Consequently, an individual's survival is affected by the nature of learning uncertainties, and search performance varies accordingly.

The trade-off between a child following its parents or exploring elsewhere can be thought of as exploitation versus exploration. However, the mechanism here is not exactly that topic. Without handing down the learning routes/destinations to offspring, neither exploitation nor exploration makes sense. We thus expect that under a learning scheme with a high level of uncertainty, the achievements of ancestors are difficult to repeat, and the search may be inefficient. The search target in such a case can be considered as changing from "looking for peaks" to "looking for plateaus", because surviving individuals must achieve high fitness in most of their possible learning directions.

To study the effect of uncertainties, we designed three learning schemes. These schemes are all bitwise climbing trials that start from solution $S_0 = G_I$ and produce solutions $S_1, S_2, \cdots$. An example of the $k$-th trial in a learning process is shown in Fig. 3.4. On the basis of previous result $S_k$, the position $i = 4$ is selected and the $i$th bit in the solution is modified to produce $S_k'$.

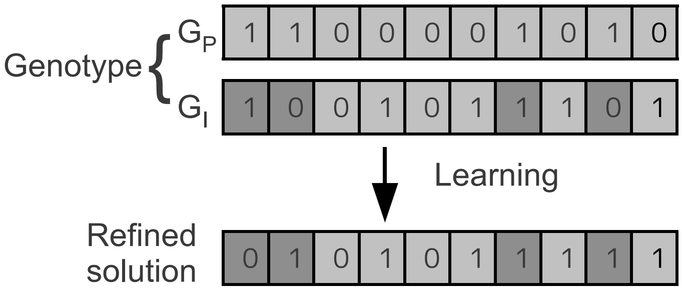


Fig.3.5: Genotype with plasticity codes.

$S'_k$ is then compared with $S_k$ and the fitter of the two is set as $S_{k+1}$. Such trials are repeated for different modifying positions until a given number of trials have been conducted. The final solution is then the learning destination and its fitness is set as the fitness of the individual.

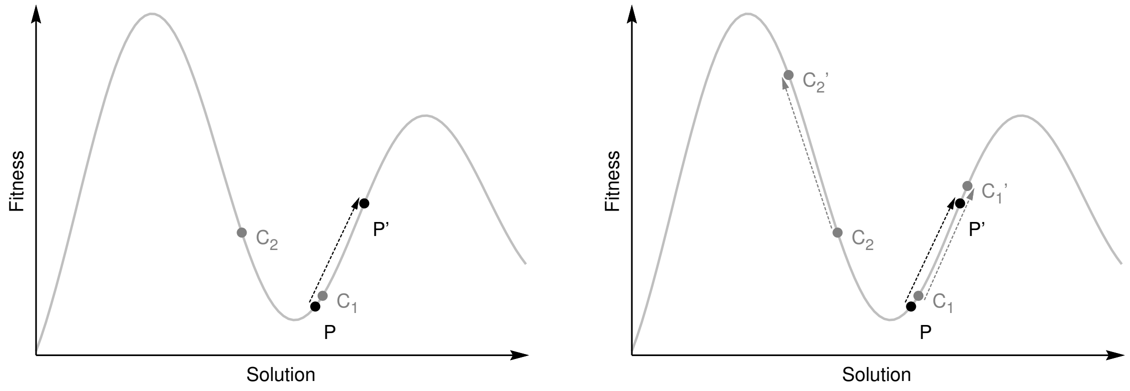
Our three designed learning schemes have similar procedures and search capabilities, but are different in terms of uncertainty level since they determine the modifying position from trial to trial by different methods. Note that the plastic positions of an individual, from most to least significant, are $i_1, i_2, \cdots, i_m, 1 \leq i_j \leq N$. For example, in Fig. 3.5, the plastic positions are 1, 2, 7 and 9. Determination of the modifying positions for the three schemes, in ascending order of uncertainty level, are as follows.

1) "Sequential" scheme: modify the plastic bits one by one in sequence order $i_1, i_2, \cdots, i_m, i_1, i_2, \cdots$. Referring to Fig. 3.5, the sequence is 1, 2, 7, 9, 1, 2, 7, 9,$\cdots$. This scheme has the fewest uncertainties among the three, and the same individual always has the same learning performance under the scheme.

2) "Permuted" scheme: modify the plastic bits one by one in a randomly permutated order. If we denote a permutation of $1, 2, \cdots, m$ as $p(1), p(2), \cdots, p(m)$, then the corresponding modified position sequence is $i_{p(1)}, i_{p(2)}, \cdots, i_{p(m)}, i_{p(1)}, i_{p(2)}, \cdots$. Starting with the same initial solution, the permutation can change, resulting in a different learning result. Referring to Fig. 3.5, the sequence could be, for instance, 1, 7, 9, 2, 1, 7, 9, 2,$\cdots$ or 9, 2, 1, 7, 9, 2, 1, 7,$\cdots$. This scheme has limited uncertainties.

3) "Random" scheme: choose a random plastic bit in each trial. Sequences are of the form $i_{r_1}, i_{r_2}, i_{r_3}, \cdots$, where $r_1, r_2, r_3, \cdots$ are randomly selected from $1, 2, \cdots, m$. Referring to Fig. 3.5, the sequence could be, for instance, 1, 2, 9, 2, 9, 7, 1,$\cdots$ or 7, 1, 1, 9, 2, 7, 1,$\cdots$. Because this scheme is random, an individual with multiple plastic bits often proceeds in a new learning direction.

Here, we limit the number of learning trial iterations to a prescribed value, which we call the "budget" hereinafter. This terminating condition for learning is easy to control and measure,



(a) Crossover breaks building blocks

(b) Children's learning

Fig.3.6: Change in inheritance caused by crossover and effect on children.

and makes comparison between generations more convenient. Learning is also commonly terminated when little to no further learning improvement occurs. While this second terminating condition is reasonable, having various learning iterations in the population is complicated. Furthermore, the fitness function in our experiments does not include a learning cost, since we are interested in only the search phase of Baldwinian evolution. During this phase, enhancement of the local search is claimed to dominate [9, 20], and the learning cost penalty is considered trivial. Studying the effect of the learning cost penalty on the search phase is beyond the scope of this chapter. Instead, we use cost-free Baldwinian learning.

### 3.2.3 Crossover Operators

The starting point of learning is another important factor in the learning process. Crossover and mutation operations add diversity during evolution and enhance search. However, these operations also break building blocks in genotypes at random, and therefore children can have initial solutions that their parents might never have experienced. In evolutionary algorithms without learning, these building blocks are directly related to beneficial traits. In Baldwinian evolution, they are related to good learning potential, and breaking such blocks prevents children from replicating their parents' learning achievements. If a child inherits genes from a parent at $P$ (Fig. 3.6), the child's initial position could be close to that of the parent (e.g., $C_1$), such that it can easily follow its parent's path. However, if genetic variation produces a considerable change, such as moving the child's position to $C_2$, the child must explore a new path.

Unlike random learning schemes that merely waste parents' efforts, crossovers and mutations add variations to the gene pool, and a beneficial variation can be inherited by later generations unless further changes occur. Genetic operators that break few building blocks enable children to realize their inherited learning potential and to accelerate search. In contrast, genotype diversity through the breaking of inherited building blocks can be advantageous, and may prevent premature convergence. The balance between preserving and breaking building blocks is an exploitation and exploration trade-off, similar to that in simple evolution. Inherited sequences are no longer representations of beneficial traits in the current case, however. Instead, inheritance maintains traits through learning, and building blocks provide promising starting points for learning.

To verify this idea and study the influence of breaking building blocks, the three crossover operators in Fig. 3.7 are compared in the experiments alongside a fixed 5% one-point mutation operation. In increasing order of breaking building blocks, the crossover operators are as follows.

1) One-bit crossover: exchange only one random position. Although this operator is uncommon, we use it to represent an extreme case. Since genotypes barely change, the population has insufficient diversity.

2) Two-point crossover: select two random positions and exchange the bit sequences between them. This operator is commonly used, and retains a certain amount of the building blocks.

3) Uniform crossover: each bit in the genotype has a 50% chance of being exchanged. This crossover operation is also popular, and breaks building blocks considerably in the binary strings.

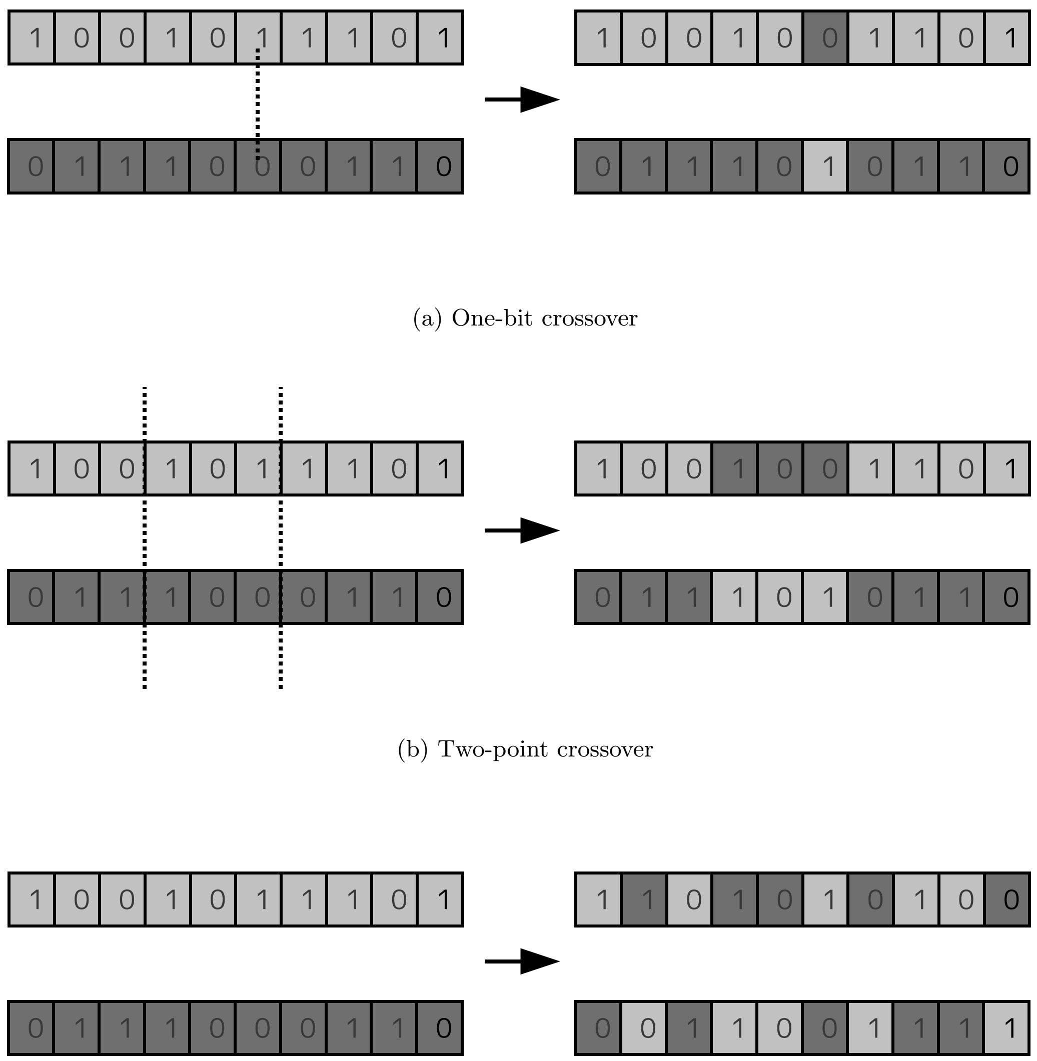
## 3.3 Learning Scheme Comparison

The following subsections introduce experiments on three landscapes in which we compare the effects of the three learning schemes. We first clarify that the schemes have comparable search capabilities, and then examine the similarity between search generations for each scheme and the inheritance of learning potential. Finally, we compare their search performance. In this section, all experiments use two-point crossover and 50% crossover rate. The results are averages of 50 runs.

### 3.3.1 Low Epistasis Landscape

First, we present the experimental results for a $K = 2$ fitness landscape with low-level epistasis. This landscape is the simplest in our experiments; genotype bits have few connections to each other, and optimization is relatively straightforward.

The three learning schemes involve similar procedures. Furthermore, the schemes are verified as being somewhat equal in terms of search capabilities when applied to the same starting population that evolved without the bias of any scheme. We evolve this population without learning



(a) One-bit crossover

(b) Two-point crossover

(c) Uniform crossover

Fig.3.7: Three types of crossovers.

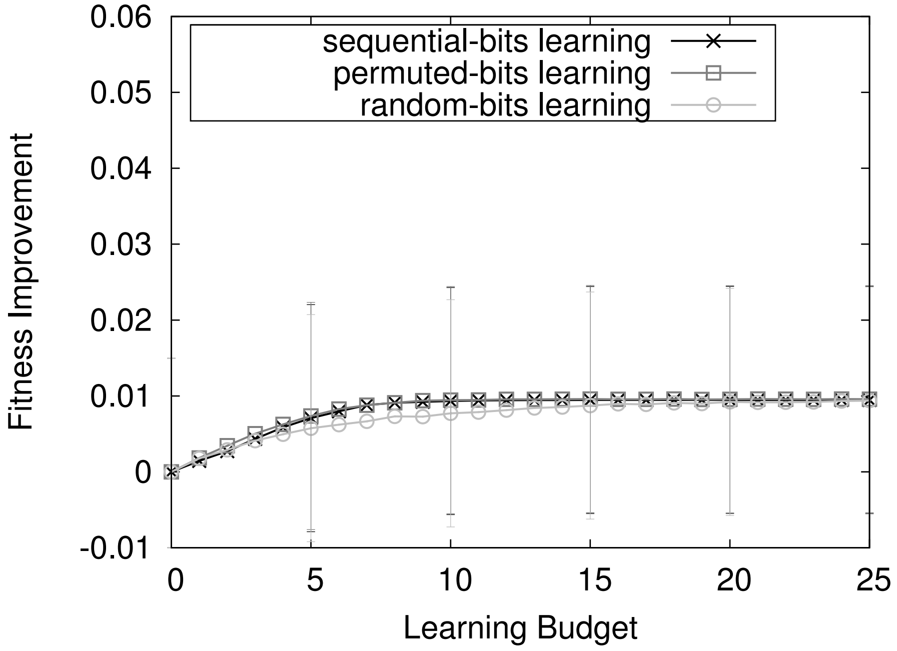


Fig.3.8: Search capabilities on $K = 2$ landscape with the same population.

until the average fitness exceeds 0.75, and then ascertain how much the fitness is improved by the learning schemes. The exact optimum is 0.817048, as found through enumerations, and therefore convergence has not occurred at a fitness of 0.75. In Fig. 3.8, the $x$-axis represents the budget, namely, the number of learning trials, and the $y$-axis represents the average fitness improvement achieved by learning. The sequential and permuted schemes have approximately the same performance, whereas that of the random scheme is slightly worse. Thus, differences between these learning schemes are caused by their uncertainties, not their search capabilities.

We also measured learning similarity between generations during each search when the learning budget is 20. The results are given in Fig. 3.9, which shows the preferences under each scheme of following parents' paths. The learning similarities of the sequential and permuted schemes increase rapidly, whereas that of the random scheme is slower.

Consequently, the preference for following parents influences the selective pressure—individuals are selected as being suitable in the learning scheme, and the potential inheritance of learning is shaped accordingly. To show the differences in learning potential, we compare the fitness values of the current generation during learning with the final value of the previous generation. To this end, we first evolve a population with a fixed learning budget of $b_{pre} = 20$ until the learning-improved fitness reaches 0.75, which we denote as $f_{pre}$. Then, during learning in the next generation, we record the improved fitness after each additional learning trial has been conducted. We denote fitness after each learning step as $f(b)$, corresponding to the budget $b$

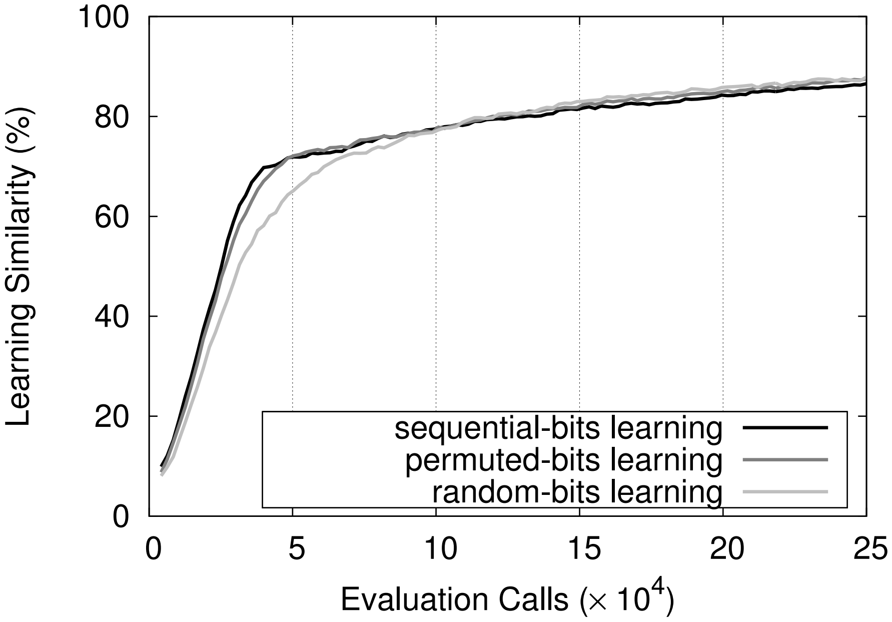


Fig.3.9: Learning similarities on $K = 2$ landscape when budget is 20.

(= 0,1,2,...). The results are shown in Fig. 3.10, where the $x$-axis represents the difference in learning budget between the current and previous generations $b - b_{pre}$, and the $y$-axis represents the difference in fitness between the current generation's learning achievements and the previous generation's final fitness $f(b) - f_{pre}$. For example, after all individuals in the current generation have taken five learning trials, the average fitness of the population is $f(5)$, which is plotted as the point $(-15, f(5) - f_{pre})$ in the figure. Table 3.2 lists further results for the three schemes, including some statistics, when $b_{pre} = 5$, 10 and 20.

The results show that fitness considerably improves during learning, but the current generation's final fitness is only a little higher than that of the previous generation, regardless of the employed learning scheme. The majority of the current generation's learning effort is hence used to catch up with the previous generation, and the fitness improvement during the current generation's learning can be considered as a measure of their learning potential. Comparing the potentials across the schemes, the sequential scheme provides greater potential than the permuted one, and the random scheme's potential is substantially lower. With fewer uncertainties, children are able to track their parents' achievements even from positions located farther away. According to the fitness improvement data in Table 3.2, differences in fitness improvement among the schemes are also seen when comparing with the previous generation. These differences may lead to different search speeds for the schemes.

Finally, we compare the search performance of the three schemes. Fig. 3.11 shows the search

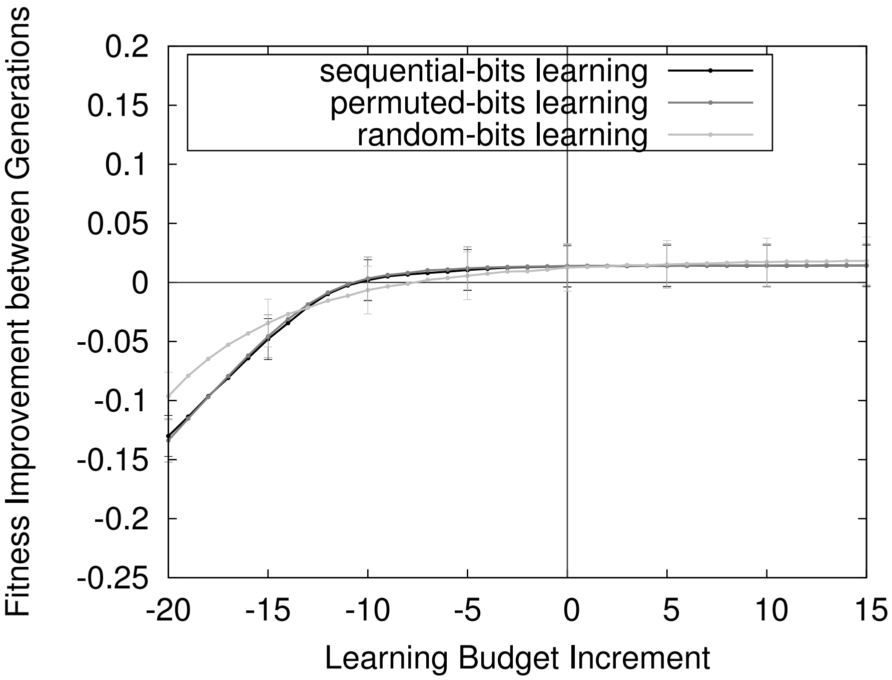


Fig.3.10: Learning improvement on $K = 2$ landscape when $b_{pre} = 20$.

Table3.2: Learning improvement on $K = 2$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Compared with Before Learning $(f(b) - f(0))$ | Sequential | 0.081556<br>(±0.023206) | 0.131644<br>(±0.031032) | 0.153381<br>(±0.030198) |
| | Permuted | 0.065039<br>(±0.026127) | 0.122322<br>(±0.028635) | 0.150823<br>(±0.031647) |
| | Random | 0.023136<br>(±0.008275) | 0.067394<br>(±0.020891) | 0.110723<br>(±0.020286) |
| Compared with Previous Generation $(f(b) - f_{pre})$ | Sequential | 0.011700<br>(±0.003829) | 0.013574<br>(±0.004168) | 0.012827<br>(±0.003877) |
| | Permuted | 0.010531<br>(±0.004331) | 0.012240<br>(±0.004364) | 0.012765<br>(±0.003866) |
| | Random | 0.008906<br>(±0.004989) | 0.008101<br>(±0.005209) | 0.009782<br>(±0.005140) |

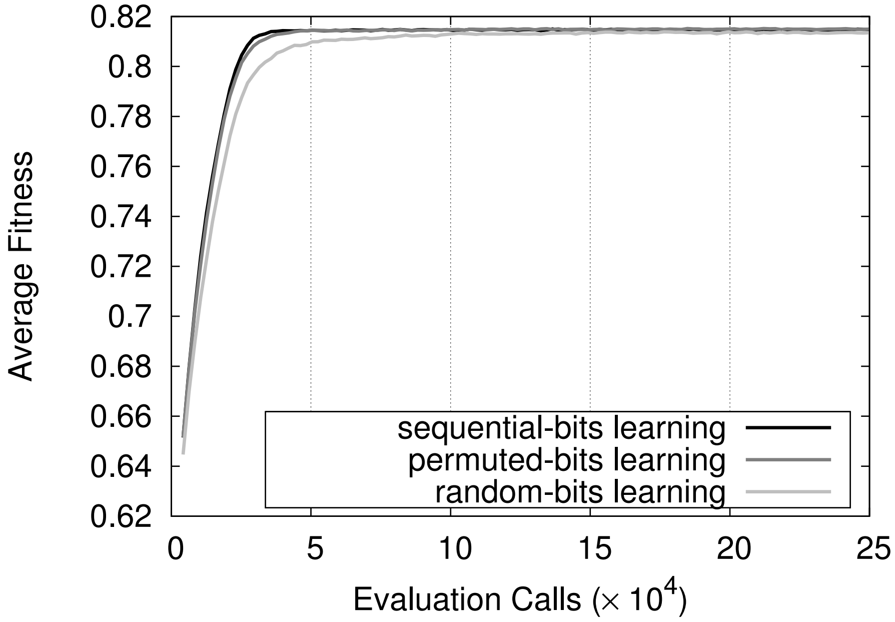


Fig.3.11: Search performance on $K = 2$ landscape when budget is 20.

performance with a fixed learning budget of 20. Table 3.3 then lists the convergence times and fitness values for the schemes when the budget is 5, 10 and 20. Since this landscape is simple, Baldwinian search under all three learning schemes leads to fairly good solutions, while their search performance is indistinct. The solutions of the sequential and permuted schemes are approximately equal in terms of high accuracy, although the sequential scheme is slightly faster. The random scheme is slowest, and converges at lower fitness values.

Thus far, the results have shown that with a high level of uncertainties, it is difficult for children to follow their parents. As the learning similarity between generations decreases, learning potential also decreases such that the search becomes slow and the final fitness value is low. However, differences in performance due to uncertainties are not distinct on this simple landscape.

### 3.3.2 Moderate Epistasis Landscape

We next examine the results for a $K = 5$ fitness landscape with higher epistasis and complexity than the previous one.

For this fitness landscape, we evolve a population without learning until the average fitness exceeds 0.70, and then determine fitness improvements. This change in the fitness threshold compared with the previous experiments is to guarantee the sampled period during the search phase. Since the current landscape has greater complexity, and the exact optimum is 0.803160,

Table3.3: Search performance on $K = 2$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Converging Fitness | Sequential | 0.814074<br>(±0.004357) | 0.814694<br>(±0.003487) | 0.814832<br>(±0.003193) |
| | Permuted | 0.813532<br>(±0.004826) | 0.815032<br>(±0.002607) | 0.815043<br>(±0.002244) |
| | Random | 0.810283<br>(±0.009025) | 0.811861<br>(±0.005343) | 0.813789<br>(±0.003234) |
| Eval. Calls to Reach 99% Fitness | Sequential | 9972<br>(±1805) | 15598<br>(±3003) | 28602<br>(±7273) |
| | Permuted | 10968<br>(±1939) | 16566<br>(±3307) | 29736<br>(±8252) |
| | Random | 13896<br>(±5498) | 23716<br>(±9082) | 37632<br>(±8003) |

the threshold is reduced. From Fig. 3.12, the sequential and permuted schemes realize similar fitness improvements, whereas that of the random scheme is slightly worse. Examining the standard deviations drawn in the figure, we see that all three schemes still have approximately the same search capability for unbiased populations.

However, differences brought about by uncertainties become distinct on this landscape. Fig. 3.13 shows the learning similarities between generations when the learning budget is 20. The sequential scheme provides a higher learning similarity than the permuted one, while the permuted scheme enables a greater number of individuals to follow parents than the random scheme does. Following parents thus becomes difficult on this landscape such that the performance of the three learning schemes have apparent distinctions.

As a result, the characteristics of surviving individuals have greater diversity among the learning schemes. Fig. 3.14 shows the fitness improvements when $b_{pre} = 20$, and Table 3.4 lists further fitness improvement data. The sequential scheme provides greater learning potential than the permuted scheme does, especially when $b_{pre} = 5$ or 10. Furthermore, the learning potential of the random scheme is considerably smaller than that of the permuted scheme. Although the improvements of the three learning schemes between generations are still similar, applying the sequential scheme is advantageous.

Fig. 3.15 shows the search performance of the schemes with a fixed budget of 20. Table 3.5 then lists convergence times and fitness values when the budget is 5, 10 and 20. Since this landscape has some complexity, the three learning schemes attain different fitness levels. The solutions of the sequential and permuted schemes are approximately equally in terms of accuracy, and are better than those of the random scheme. Conversely, the sequential scheme

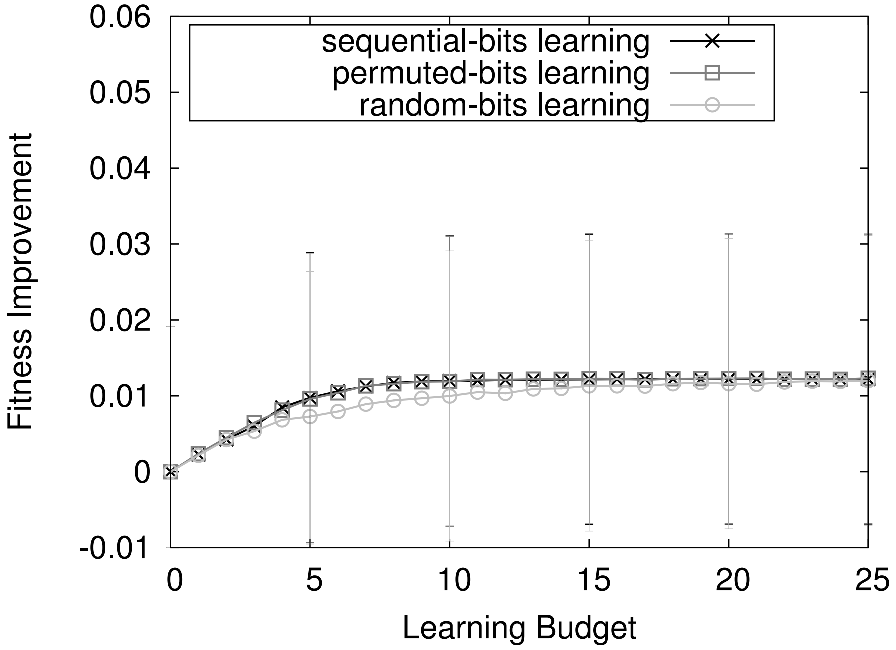


Fig.3.12: Search capabilities on $K = 5$ landscape with the same population.

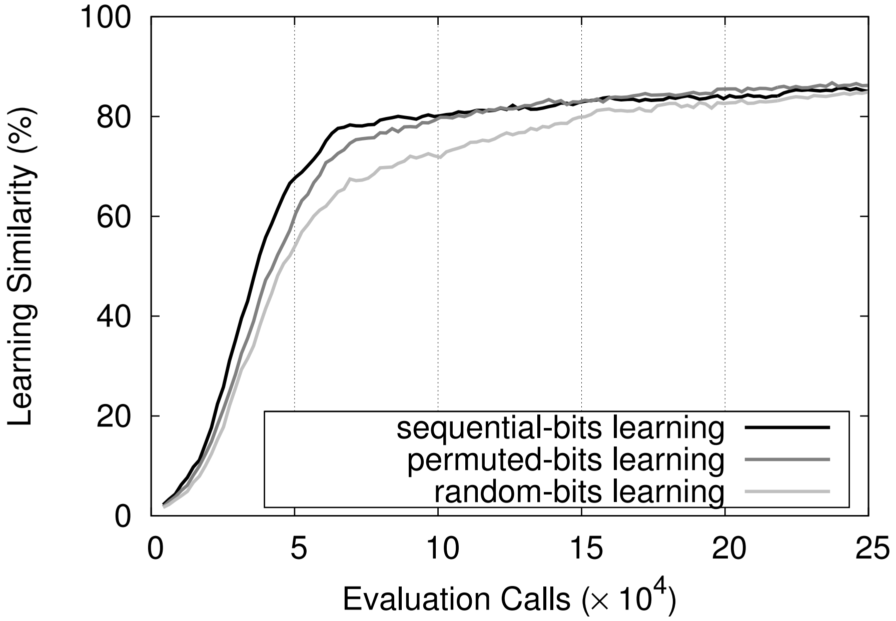


Fig.3.13: Learning similarities on $K = 5$ landscape when budget is 20.

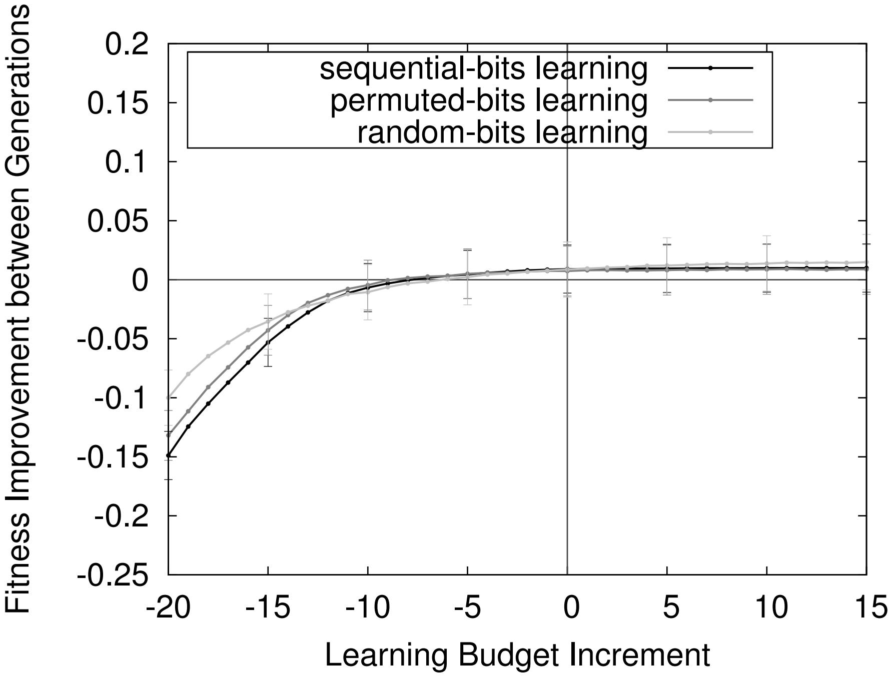


Fig.3.14: Learning improvement on $K = 5$ landscape when $b_{pre} = 20$.

Table3.4: Learning improvement on $K = 5$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Compared with Before Learning $(f(b) - f(0))$ | Sequential | 0.107202<br>(±0.031928) | 0.133819<br>(±0.025349) | 0.150970<br>(±0.025699) |
| | Permuted | 0.076220<br>(±0.036756) | 0.114828<br>(±0.026211) | 0.143798<br>(±0.028833) |
| | Random | 0.027006<br>(±0.012308) | 0.062833<br>(±0.023066) | 0.107060<br>(±0.028150) |
| Compared with Previous Generation $(f(b) - f_{pre})$ | Sequential | 0.007008<br>(±0.005064) | 0.008571<br>(±0.007307) | 0.009232<br>(±0.005002) |
| | Permuted | 0.008071<br>(±0.006426) | 0.008298<br>(±0.006023) | 0.008034<br>(±0.006350) |
| | Random | 0.007872<br>(±0.006299) | 0.007061<br>(±0.006347) | 0.008646<br>(±0.005188) |

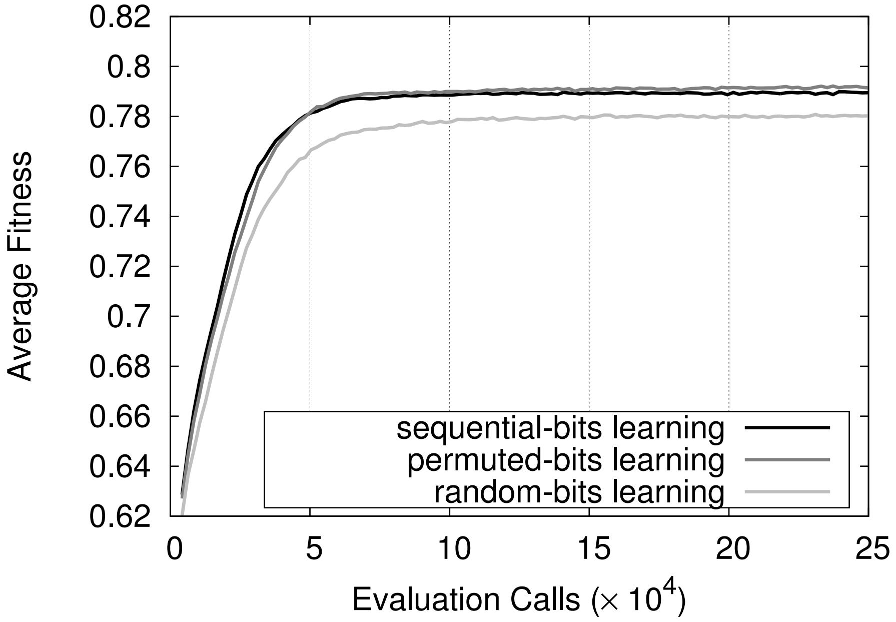


Fig.3.15: Search performance on $K = 5$ landscape when budget is 20.

Table3.5: Search performance on $K = 5$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Converging Fitness | Sequential | 0.777821<br>(±0.020729) | 0.786345<br>(±0.014021) | 0.788744<br>(±0.012398) |
| | Permuted | 0.781688<br>(±0.014764) | 0.786077<br>(±0.011939) | 0.789492<br>(±0.011760) |
| | Random | 0.769680<br>(±0.023172) | 0.776895<br>(±0.019193) | 0.782463<br>(±0.014602) |
| Eval. Calls to Reach 99% Fitness | Sequential | 14964<br>(±6475) | 29348<br>(±20068) | 54894<br>(±46130) |
| | Permuted | 20100<br>(±13975) | 52272<br>(±81908) | 63462<br>(±45098) |
| | Random | 27708<br>(±35874) | 38742<br>(±17893) | 62244<br>(±23109) |

converges more rapidly than the other two schemes.

The results on this landscape also show the difficulty for children to follow their parents under uncertainties. The differences in learning similarity, learning potential, and search performance are more distinct for a more complex landscape, especially when comparing the sequential and random schemes. The effect of uncertainties is thus similar to that for the $K = 2$ landscape.

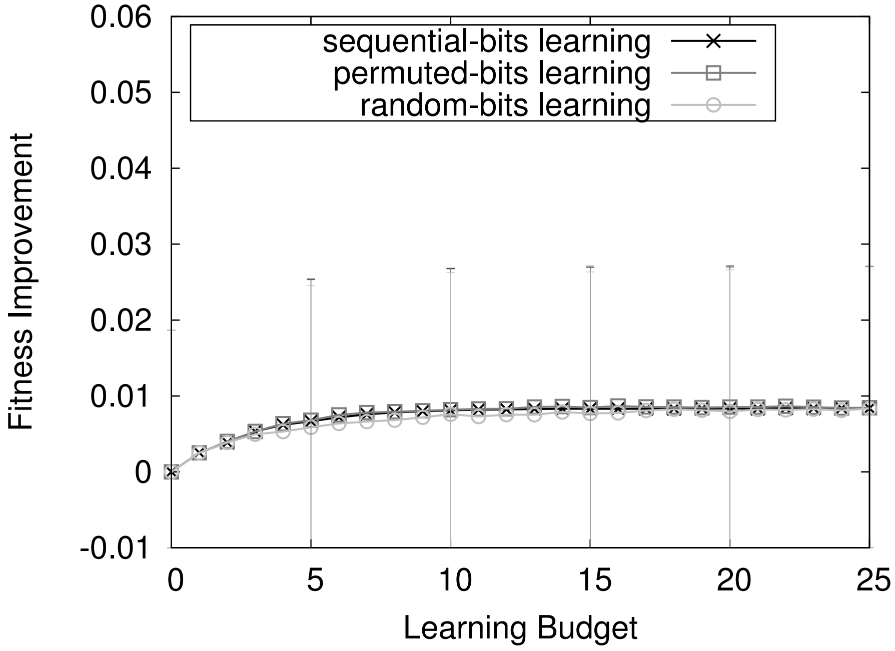


Fig.3.16: Search capabilities on $K = 15$ landscape with the same population.

### 3.3.3 High Epistasis Landscape

We finally examine a highly complex $K = 15$ fitness landscape. The epistasis of this landscape is thus very high. Any crossover or mutation in such a case is likely to break building blocks and prevent children from following their parents.

For this fitness landscape, the sampling threshold during the search phase is set to 0.65, because under some learning schemes/budgets, the average fitness may not even reach 0.70 (the exact optimum is 0.801496). The three schemes still have similar search capabilities on this landscape (Fig. 3.16).

The difference in learning similarity is highly evident on this landscape where following is always difficult. Fig. 3.17 shows the similarities when the learning budget is 20. The separation between the plots of the three schemes is obvious compared with those on the other two landscapes. Even after convergence, high uncertainties result in lower learning similarities.

Differences in learning potential are also clear on this landscape. Fig. 3.18 shows the fitness improvements when $b_{pre} = 20$, and Table 3.6 lists further data. The sequential scheme provides greater learning potential than the permuted scheme does, when $b_{pre} = 5$, 10 or 20. The random scheme still provides the least learning potential. From Table 3.6, improvements between generations are small under all three learning schemes. With parents' learning hard to replicate on this landscape, the random scheme may show greater improvement than the sequential scheme

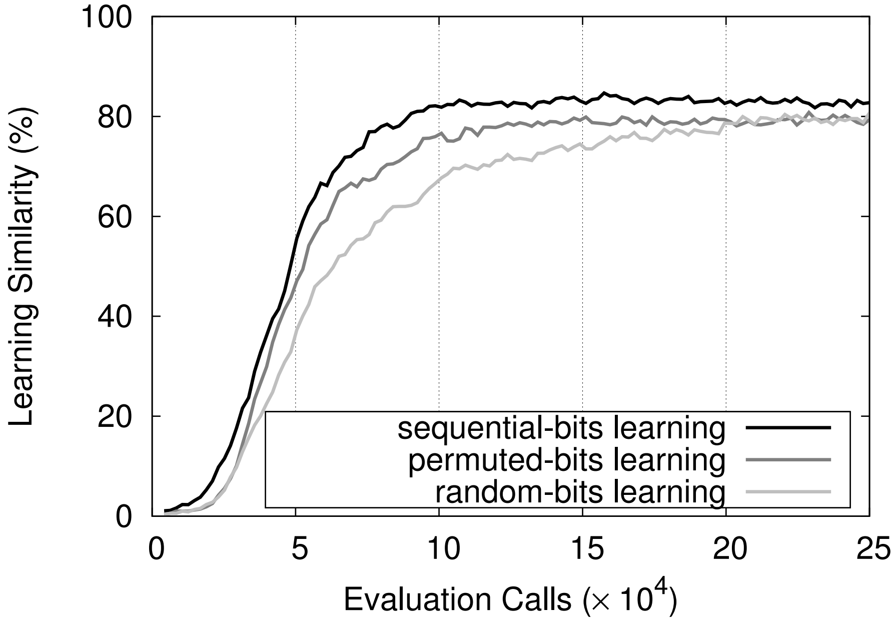


Fig.3.17: Learning similarities on $K = 15$ landscape when budget is 20.

when the learning budget is high, since the random scheme enables individuals to traverse a greater number of possible solutions through learning.

Fig. 3.19 shows the fitness growth of the schemes with a fixed learning budget of 20. Table 3.7 then lists the convergence times and fitness values when the budget is 5, 10 and 20. Since this landscape is highly complex, the solutions of all three learning schemes are far from the real optimum. Substantial differences are found among the schemes. Baldwinian search under the sequential scheme finds solutions with the highest accuracy, and has the fastest convergence. The random scheme shows greater improvements during search, but improvements are not maintained throughout the evolution. Thus the search becomes slow and solutions are unfit.

All three schemes are unable to find the global optimum on this landscape. Moreover, their ability to determine a relatively good solution is different. With the least uncertainties, the sequential scheme retains a high learning similarity and high learning potential. Consequently, applying this scheme leads to faster convergence and fitter solutions.

## 3.4 Crossover Comparison

The following subsections introduce experiments on the same three landscapes as in Section 3.3, in which we compare the effects of the three crossover operators. We first examine the performance of the crossovers under evolution without learning, and then measure their search

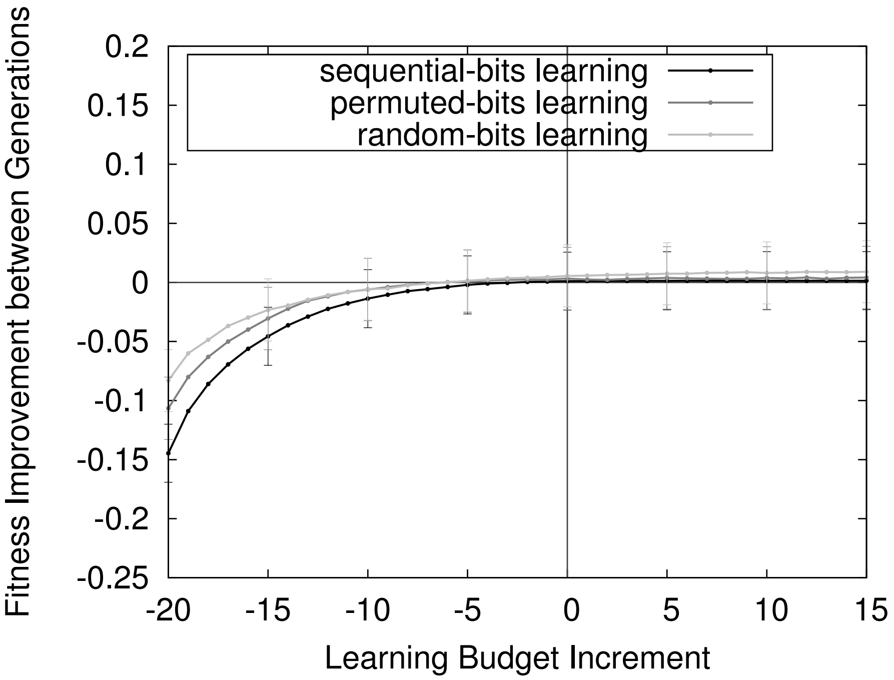


Fig.3.18: Learning improvement on $K = 15$ landscape when $b_{pre} = 20$.

Table3.6: Learning improvement on $K = 15$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Compared with<br>Before Learning<br>$(f(b) - f(0))$ | Sequential | 0.140207<br>(±0.049448) | 0.142670<br>(±0.026759) | 0.151701<br>(±0.025760) |
| | Permuted | 0.068164<br>(±0.042362) | 0.096168<br>(±0.033593) | 0.109059<br>(±0.025264) |
| | Random | 0.027944<br>(±0.027808) | 0.060741<br>(±0.039543) | 0.081730<br>(±0.029312) |
| Compared with<br>Previous Generation<br>$(f(b) - f_{pre})$ | Sequential | 0.006983<br>(±0.009712) | 0.005321<br>(±0.008959) | 0.000936<br>(±0.006853) |
| | Permuted | 0.006880<br>(±0.008140) | 0.005577<br>(±0.008986) | 0.003944<br>(±0.006582) |
| | Random | 0.006249<br>(±0.009153) | 0.006954<br>(±0.008443) | 0.005423<br>(±0.007555) |

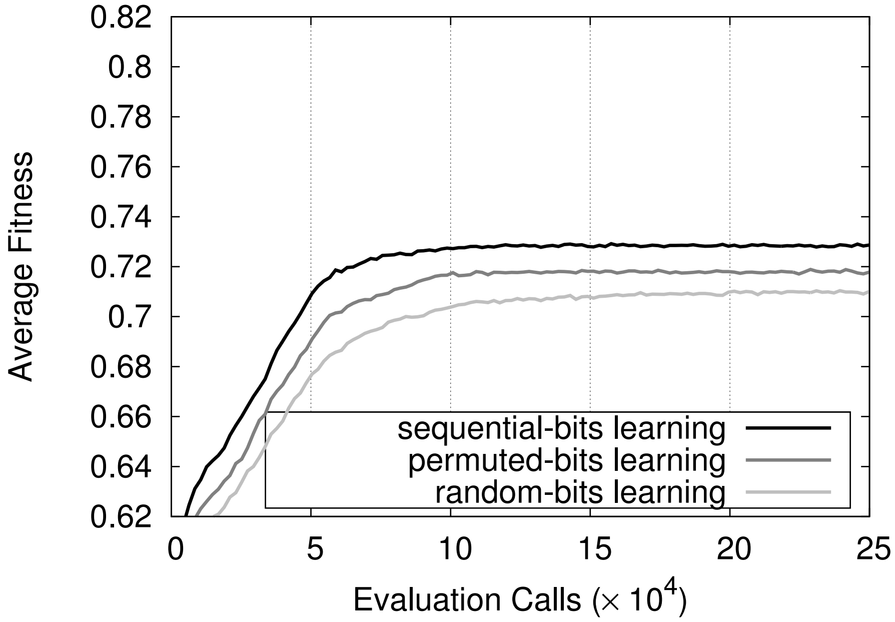


Fig.3.19: Search performance on $K = 15$ landscape when budget is 20.

Table3.7: Search performance on $K = 15$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Converging Fitness | Sequential | 0.713321<br>(±0.022128) | 0.726721<br>(±0.017311) | 0.730913<br>(±0.018842) |
| | Permuted | 0.704109<br>(±0.024239) | 0.708605<br>(±0.016018) | 0.716028<br>(±0.021400) |
| | Random | 0.696998<br>(±0.026925) | 0.704753<br>(±0.021346) | 0.709458<br>(±0.020702) |
| Eval. Calls to Reach 99% Fitness and | Sequential | 30984<br>(±66048) | 32274<br>(±24799) | 54348<br>(±14932) |
| | Permuted | 30384<br>(±39435) | 33594<br>(±13879) | 79926<br>(±71295) |
| | Random | 37584<br>(±75975) | 49676<br>(±51267) | 66948<br>(±23132) |

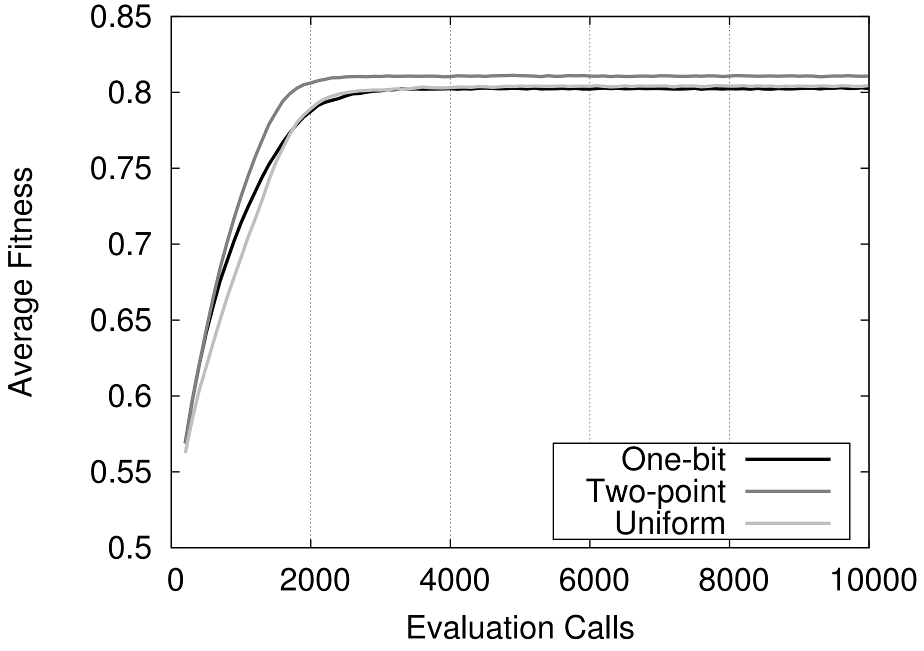


Fig.3.20: Search without learning on $K = 2$ landscape.

similarities and learning potentials. Finally, we compare their search performance. In this section all experiments use the sequential learning scheme, and crossover rate is set to 75% such that the results of each operation are distinct. All results are averages of 50 runs.

### 3.4.1 Low Epistasis Landscape

Crossover operators provide different trade-offs between diversity improvement and learning potential degradation. We first apply the operators under the condition of learning-free evolution in order to compare their performance with that under Baldwinian evolution conditions. From Fig. 3.20, two-point crossover provides the fastest convergence and fittest solutions, whereas one-bit and uniform crossover converge at lower fitness values. The lack of diversity in genomes and the high level of building blocks breaking produced by one-bit and uniform crossovers limit their performance, respectively.

We expect from the operator designs that one-bit and uniform crossovers break building blocks the least and most, respectively. This supposition is verified in Fig. 3.21, which shows the learning similarities of the three operations when the learning budget is 20. One-bit crossover provides the highest learning similarity over the entire search, attaining a value of nearly 90% when terminated. Two-point crossover achieves a higher learning similarity than uniform crossover does, but this similarity is considerably lower than that of one-bit crossover.

Although the three crossovers provide different learning similarities, their learning potentials

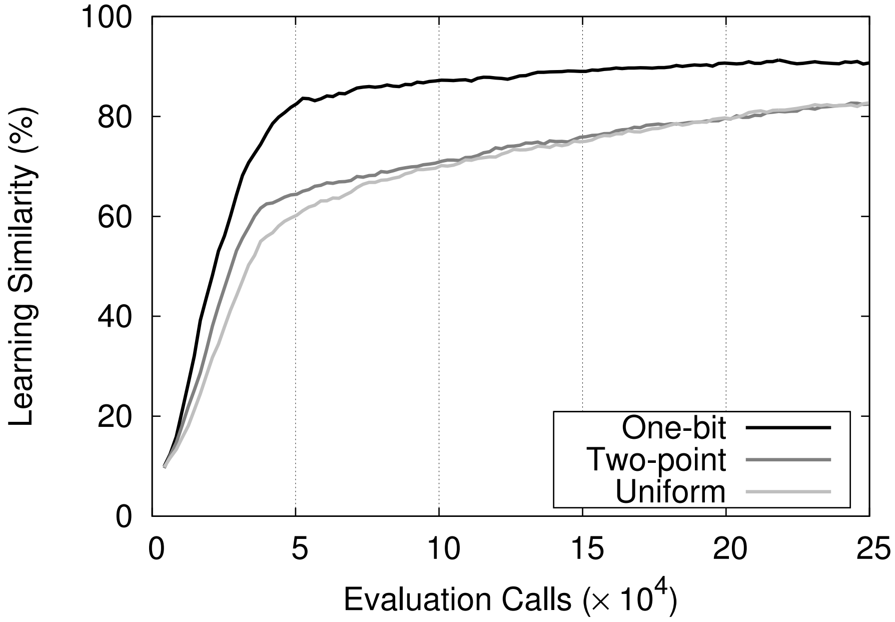


Fig.3.21: Learning similarities on $K = 2$ landscape when budget is 20.

are almost equal. Fig. 3.22 shows the fitness improvements provided by the crossovers when $b_{pre} = 20$. Further fitness improvements results are included in Table 3.8. The highest learning potential is attained with one-bit crossover, while two-point crossover provides higher learning potential than uniform crossover does. However, the difference between the potentials is small. Changes in learning resulting from the genetic operators are inheritable, and may explain why the learning potential is the same across the crossovers.

Fig. 3.23 shows search performance with the crossovers when the learning budget is 20. Table 3.9 lists convergence time and fitness values when the budget is 5, 10 ad 20. With one-bit crossover, the search initially rises at a similar rate to that with two-point crossover, but converges at a lower fitness value. Conversely, the search is slower with uniform crossover, but converges at a value equal to that with two-point crossover. The performance of all three Baldwinian evolutions is similar to that without learning, except that uniform and two-point crossovers provide almost the same solutions here.

Applying the three crossover operators on this landscape, the learning similarity varies according to the level of breaking building blocks. However, little difference is found in terms of learning potential. The crossovers influence search in similar manners without learning and with Baldwinian learning.

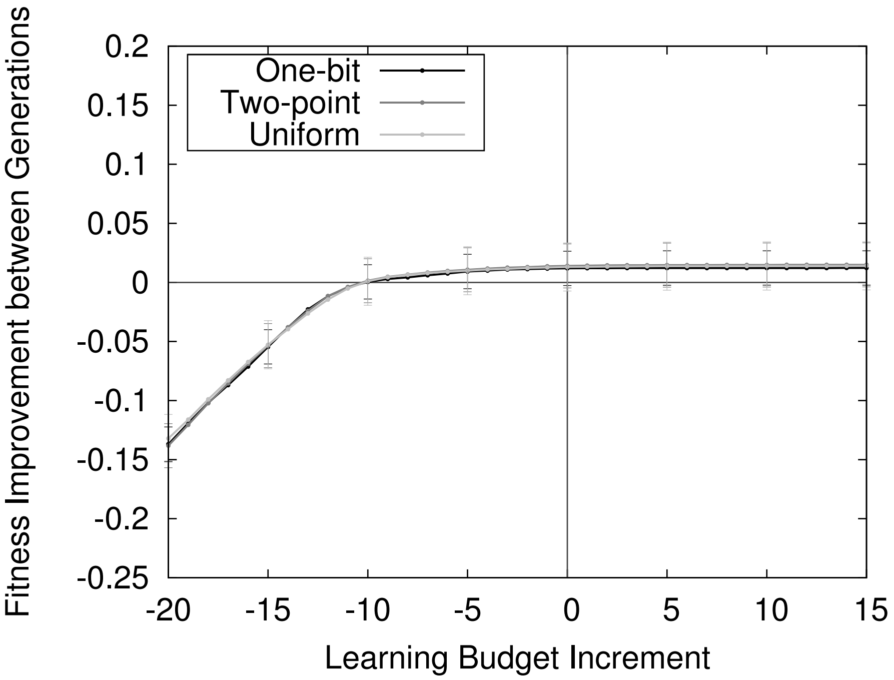


Fig.3.22: Learning improvement on $K = 2$ landscape when $b_{pre} = 20$.

Table3.8: Learning improvement on $K = 2$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Compared with Before Learning $(f(b) - f(0))$ | One-bit | 0.094481<br>(±0.033120) | 0.139956<br>(±0.038957) | 0.151684<br>(±0.038570) |
| | Two-point | 0.079049<br>(±0.018962) | 0.131793<br>(±0.028551) | 0.145607<br>(±0.027656) |
| | Uniform | 0.069697<br>(±0.014039) | 0.124134<br>(±0.020257) | 0.144526<br>(±0.021511) |
| Compared with Previous Generation $(f(b) - f_{pre})$ | One-bit | 0.010360<br>(±0.004386) | 0.011148<br>(±0.003973) | 0.012465<br>(±0.004322) |
| | Two-point | 0.013248<br>(±0.003827) | 0.014986<br>(±0.003204) | 0.014910<br>(±0.003013) |
| | Uniform | 0.010752<br>(±0.004505) | 0.010962<br>(±0.004712) | 0.012998<br>(±0.004502) |

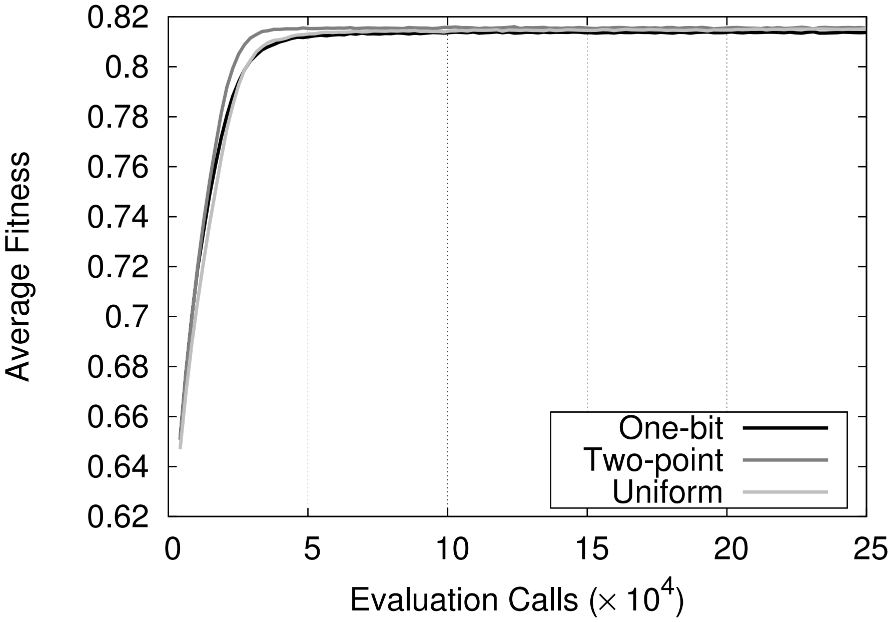


Fig.3.23: Search performance on $K = 2$ landscape when budget is 20.

Table3.9: Search performance on $K = 2$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Converging Fitness | One-bit | 0.808148<br>(±0.009504) | 0.812554<br>(±0.006096) | 0.812834<br>(±0.005742) |
| | Two-point | 0.814171<br>(±0.003854) | 0.815382<br>(±0.002252) | 0.815449<br>(±0.001064) |
| | Uniform | 0.812643<br>(±0.005507) | 0.814180<br>(±0.004206) | 0.815660<br>(±0.000879) |
| Eval. Calls to Reach 99% Fitness | One-bit | 18132<br>(±51235) | 17182<br>(±4073) | 30912<br>(±7150) |
| | Two-point | 9600<br>(±1494) | 15004<br>(±2264) | 26922<br>(±3858) |
| | Uniform | 14616<br>(±9552) | 19074<br>(±4301) | 33810<br>(±13970) |

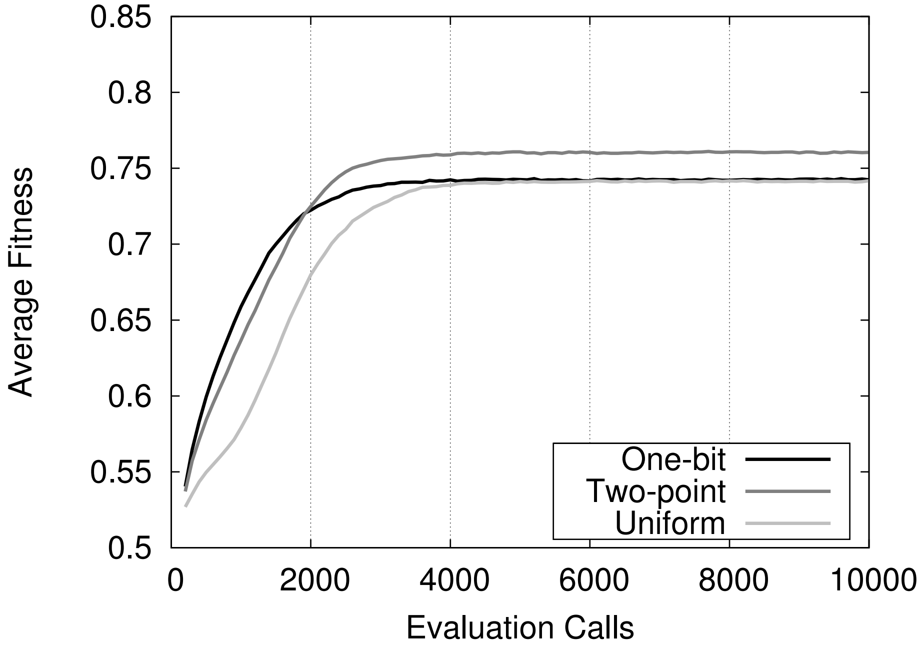


Fig.3.24: Search without learning on $K = 5$ landscape.

### 3.4.2 Moderate Epistasis Landscape

Fig. 3.24 shows the search performance with the crossover operators under the condition of no-learning evolution. Search is fastest with one-bit crossover, but attains the lowest fitness value at convergence. With two-point crossover, the search is slightly slower, but the highest fitness is achieved.

Fig. 3.13 shows the learning similarities on the landscape when the learning budget is 20. Considerable differences are still seen in learning similarities. In particular, the separation between the plots associated with each crossover is greater on this landscape.

Fig. 3.26 shows the fitness improvements provided by the crossovers when $b_{pre} = 20$. Further results are included in Table 3.10. The difference between the potentials is even smaller than on the $K = 2$ landscape.

Fig. 3.27 shows the search performance with the crossovers when the learning budget is 20. Table 3.11 lists the convergence times and fitness values when the budget is 5, 10, and 20. Search performance is also similar to that without learning. With one-bit crossover, the fitness rises quickly but converges too early. With two-point crossover, the search achieves the highest fitness, and converges only slightly later than the search with one-bit crossover.

On this landscape, crossover operator effects without learning evolution and with Baldwinian evolution are still equal. Learning similarities vary considerably according to the crossover,

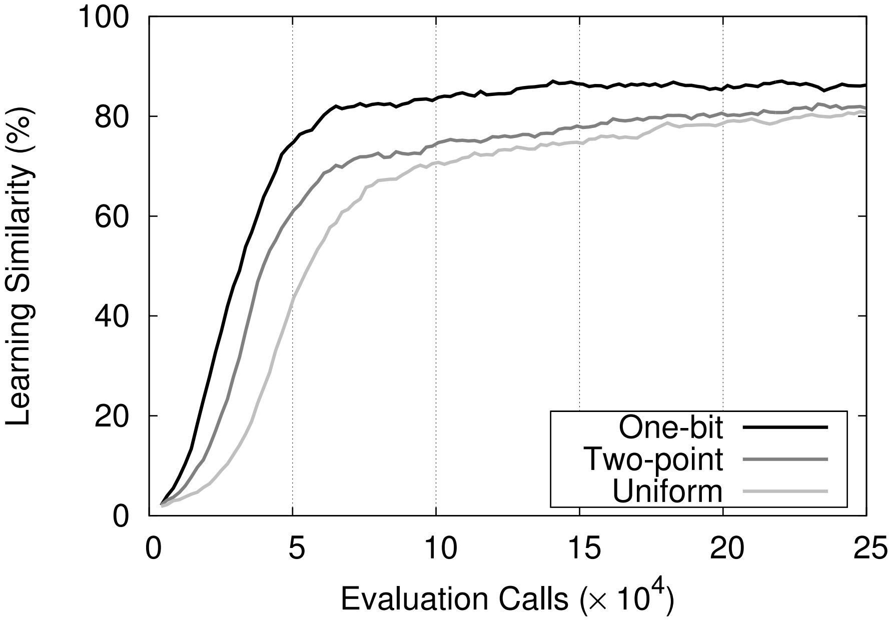


Fig.3.25: Learning similarities on $K = 5$ landscape when budget is 20.

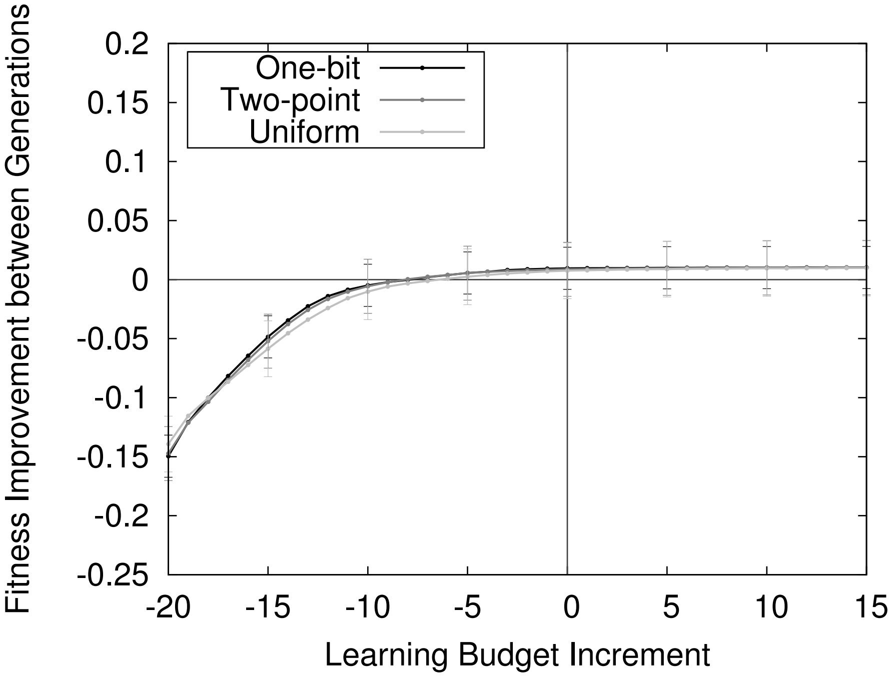


Fig.3.26: Learning improvement on $K = 5$ landscape when $b_{pre} = 20$.

Table3.10: Learning improvement on $K = 5$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Compared with Before Learning $(f(b) - f(0))$ | One-bit | 0.106776<br>(±0.040169) | 0.141225<br>(±0.029885) | 0.155552<br>(±0.026686) |
| | Two-point | 0.093782<br>(±0.028046) | 0.134509<br>(±0.023538) | 0.149825<br>(±0.026349) |
| | Uniform | 0.090604<br>(±0.023266) | 0.129036<br>(±0.023969) | 0.154065<br>(±0.017483) |
| Compared with Previous Generation $(f(b) - f_{pre})$ | One-bit | 0.007807<br>(±0.005599) | 0.008925<br>(±0.004710) | 0.008783<br>(±0.004817) |
| | Two-point | 0.009067<br>(±0.007128) | 0.011150<br>(±0.005115) | 0.008801<br>(±0.005956) |
| | Uniform | 0.009676<br>(±0.006719) | 0.007965<br>(±0.006587) | 0.007605<br>(±0.005735) |

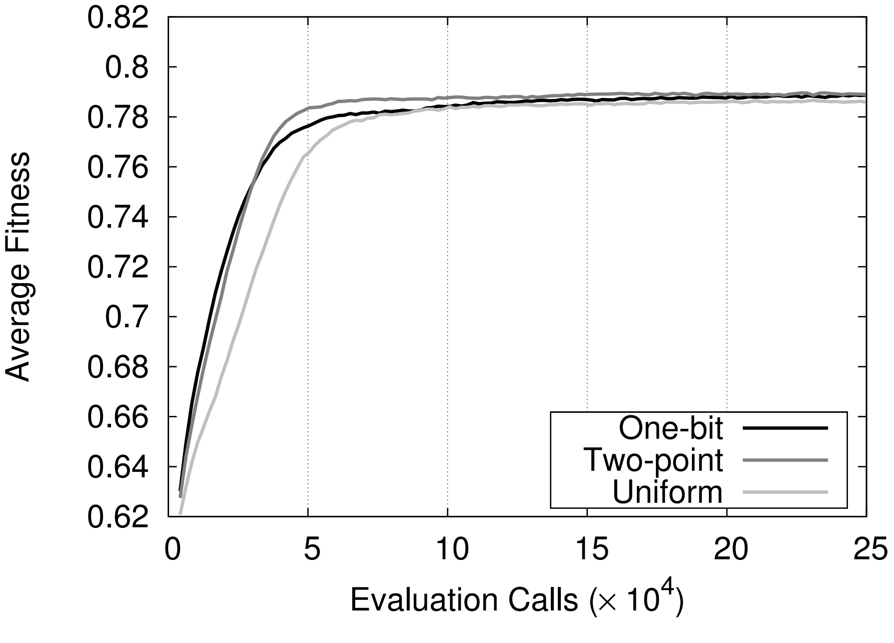


Fig.3.27: Search performance on $K = 5$ landscape when budget is 20.

Table3.11: Search performance on $K = 5$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Converging Fitness | One-bit | 0.766761<br>(±0.024777) | 0.784450<br>(±0.018082) | 0.783249<br>(±0.016570) |
| | Two-point | 0.781207<br>(±0.018430) | 0.787092<br>(±0.013144) | 0.792753<br>(±0.009547) |
| | Uniform | 0.776598<br>(±0.016397) | 0.784388<br>(±0.013313) | 0.790300<br>(±0.015055) |
| Eval. Calls to Reach 99% Fitness | One-bit | 39300<br>(±81156) | 40656<br>(±46077) | 58338<br>(±71022) |
| | Two-point | 19524<br>(±20130) | 27434<br>(±9242) | 61026<br>(±40715) |
| | Uniform | 24456<br>(±26856) | 47366<br>(±56264) | 101220<br>(±77513) |

whereas learning potentials vary only slightly.

### 3.4.3 High Epistasis Landscape

Fig. 3.28 shows the search performance under the condition of learning-free evolution. The search performance on this fitness landscape, which has the highest complexity, is quite different from that on the other landscapes. Search is the fastest with one-bit crossover, and the highest fitness value is attained at convergence. Search is the slowest with uniform crossover, while the lowest fitness is achieved. Uniform crossover breaks inherited blocks too many times on this landscape, and retaining positive traits becomes beneficial.

As expected, the differences between learning similarities increases even more on this landscape, as shown in Fig. 3.17 when the learning budget is 20. In the early stages of a search with any of the three crossovers, very few children repeat their parents' learning. However, this period finishes earlier with a crossover that does not break building blocks greatly.

Fig. 3.30 shows the fitness improvements when $b_{pre} = 20$. Further results are included in Table 3.12. As on the other two landscapes, the learning potentials are almost equal in this case.

Fig. 3.31 shows the search performance when the learning budget 20. Table 3.13 then lists the convergence times and fitness values when the budget is 5, 10 and 20. Similar to the case of evolution without learning, the Baldwinian search becomes slower when the crossover breaks many building blocks, and converges at a lower fitness value.

As on the other two landscapes, crossover operator effects on search performance are similar without learning evolution and with Baldwinian evolution. Crossover operators change the

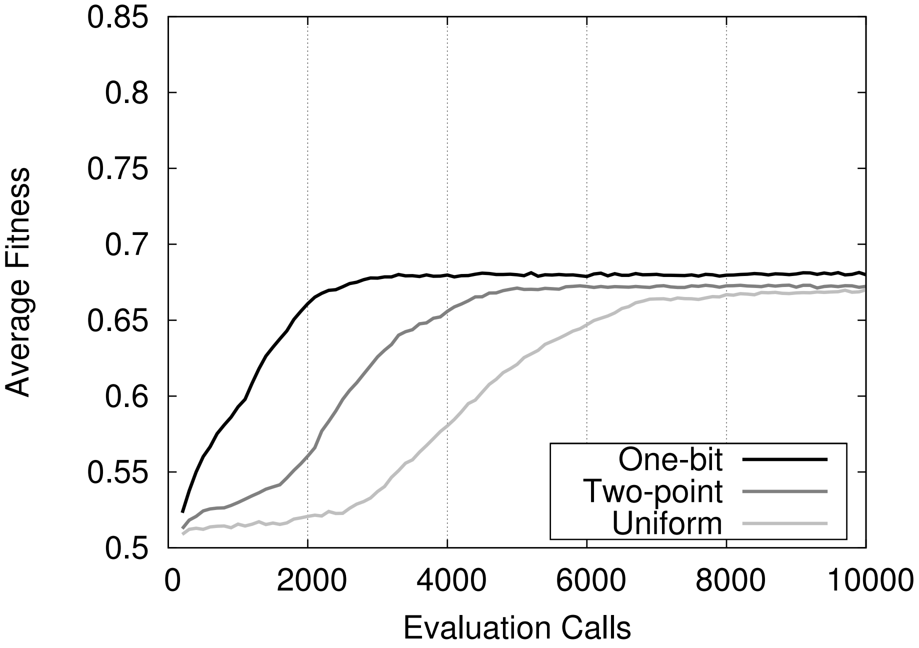


Fig.3.28: Search without learning on $K = 15$ landscape.

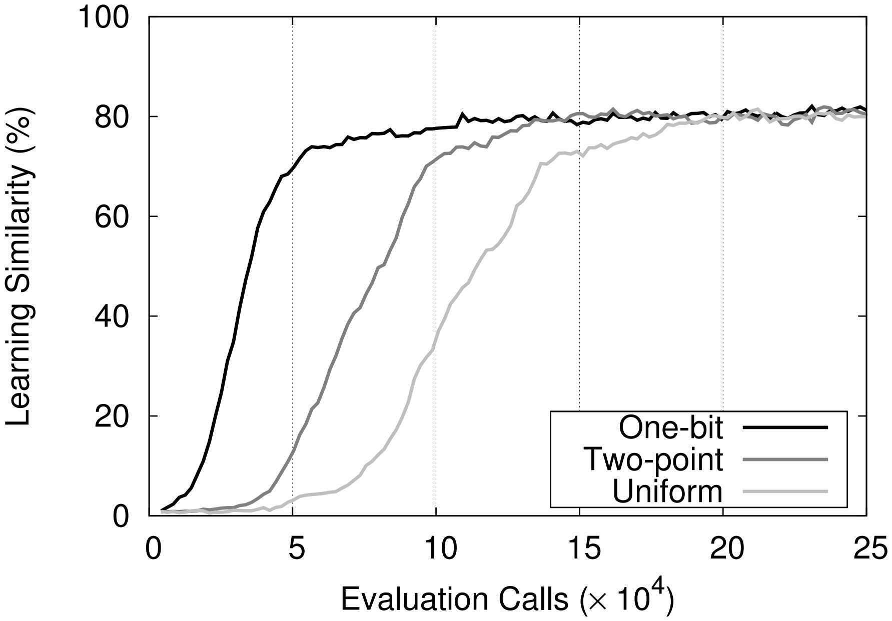


Fig.3.29: Learning similarities on $K = 15$ landscape when budget is 20

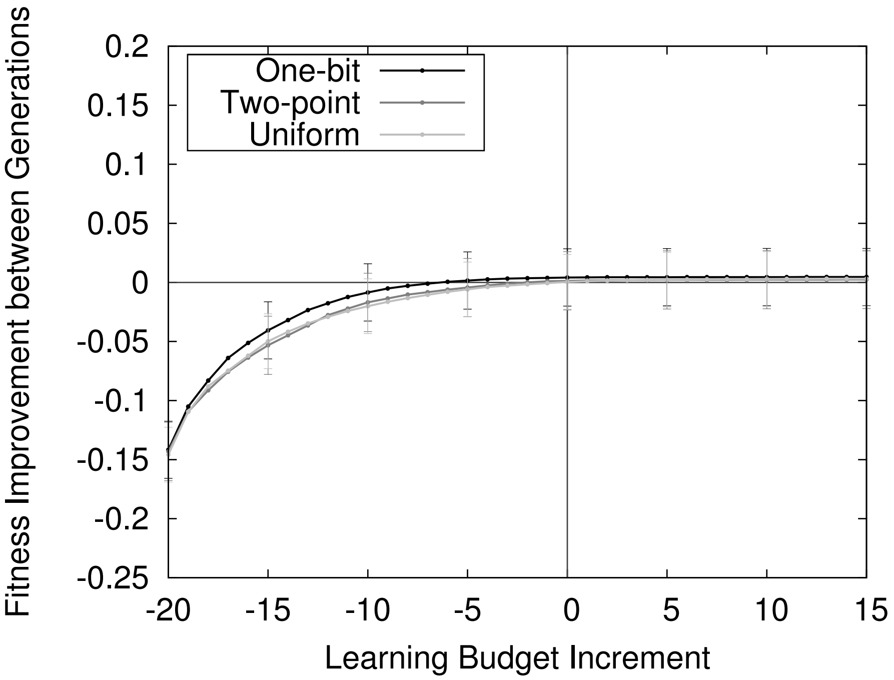


Fig.3.30: Learning improvement on $K = 15$ landscape when $b_{pre} = 20$.

Table3.12: Learning improvement on $K = 15$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Compared with Before Learning $(f(b) - f(0))$ | One-bit | 0.131926<br>(±0.033166) | 0.139718<br>(±0.027106) | 0.148496<br>(±0.023469) |
| | Two-point | 0.109040<br>(±0.046359) | 0.139111<br>(±0.029782) | 0.146858<br>(±0.014729) |
| | Uniform | 0.108928<br>(±0.032270) | 0.125843<br>(±0.026315) | 0.138671<br>(±0.017363) |
| Compared with Previous Generation $(f(b) - f_{pre})$ | One-bit | 0.004638<br>(±0.008298) | 0.004029<br>(±0.007335) | 0.003473<br>(±0.005131) |
| | Two-point | 0.007676<br>(±0.009682) | 0.005871<br>(±0.007483) | 0.002583<br>(±0.007728) |
| | Uniform | 0.010552<br>(±0.007672) | 0.005524<br>(±0.007909) | 0.000325<br>(±0.006829) |

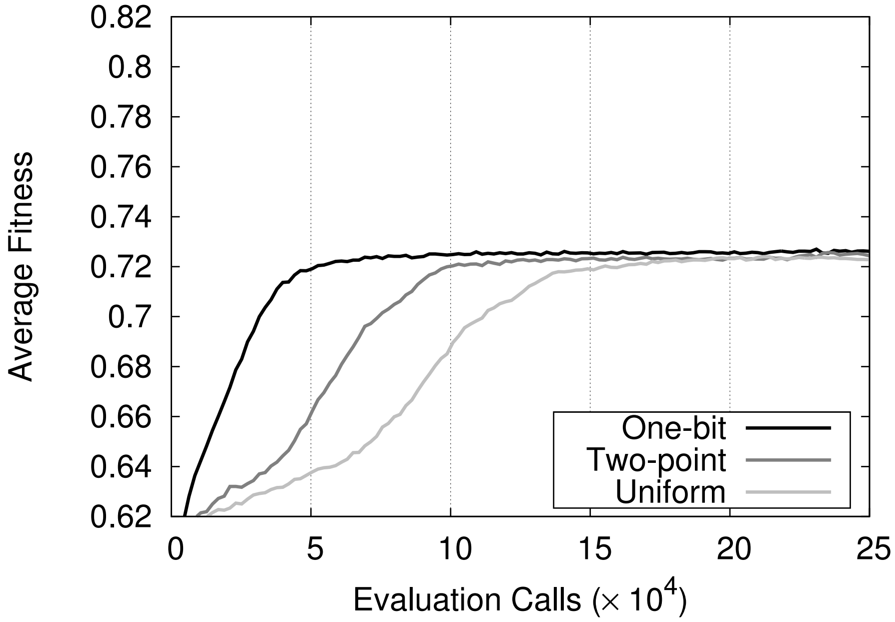


Fig.3.31: Search performance on $K = 15$ landscape when budget is 20.

Table3.13: Search performance on $K = 15$ landscape

| | Budget | 5 | 10 | 20 |
|---|---|---|---|---|
| Converging Fitness | One-bit | 0.714271<br>(±0.020064) | 0.723905<br>(±0.024034) | 0.726737<br>(±0.022697) |
| | Two-point | 0.704900<br>(±0.022111) | 0.723319<br>(±0.017530) | 0.726353<br>(±0.019928) |
| | Uniform | 0.706928<br>(±0.019591) | 0.718666<br>(±0.018622) | 0.720197<br>(±0.017455) |
| Eval. Calls to Reach 99% Fitness | One-bit | 12828<br>(±8854) | 47784<br>(±85079) | 50316<br>(±41511) |
| | Two-point | 24288<br>(±8302) | 44330<br>(±17673) | 88704<br>(±51589) |
| | Uniform | 37032<br>(±22477) | 78474<br>(±52445) | 143136<br>(±77618) |

learning similarities in searches, but learning potentials are almost the same with all operators.

## 3.5 Discussion

In the experiments, the effects of learning scheme uncertainties and crossover's building block breaking were examined. Both of these properties influence the preference of children to follow their parents' learning paths. However, the effects of each property are completely different. Under different learning schemes, learning potentials vary substantially, although their improvement capabilities are approximately equal when applied to an unbiased starting population. Search performance varies such that when the uncertainty level rises, the converging speed and final fitness are both degraded. With different crossover operators, the learning similarity considerably changes, but the learning potential hardly varies. Search performance alters according to the well-known scenario of exploitation/exploration trade-off.

Performance differences occur due to differences in the underlying mechanisms. Learning scheme uncertainties vary the learning process, whereas crossover's block breaking varies the learning start points. Randomness in learning schemes is a part of learning, in addition to evolution, and changes resulting from randomness are not guaranteed to be inherited or repeated. Block breaking during crossover is a part of reproduction, and changes take the form of genotype diversity, which can be inherited if no further variation occurs. The above is the difference between noninheritable and inheritable characteristics, which determine whether a parents' learning efforts are "wasted".

Following the pioneering work by Hinton and Nowlan [20], conventional statements of Baldwinian evolution describe the search phase as "smoothing the fitness landscape". In such a case, learning can be considered a necessary part of the fitness function. However, learning is not a simple, deterministic mapping from a "needle in a haystack" landscape to a "smoothed" landscape. Uncertainties are often found in learning schemes, and an initial solution may reach several different refined solutions after learning, while a single refined solution may result from different initial solutions. A smoothed landscape is analogous to a noisy landscape, and a smoothing effect is analogous to a noisy landscape transformation. Thus, we do not employ the classic view of landscape transformation here, but instead focus on detailed dynamics—specifically, comparison of learning behavior between generations—to explore unanswered questions on the smoothing effect.

Our viewpoint is necessary for understanding Baldwinian evolution. In Lamarckian evolution, children's achievements through learning are always beyond those of their parents. But in Baldwinian evolution, realization of inherited learning potential calls for learning effort, and successful realization is not guaranteed. Our findings may imply that, in Baldwinian evolution, what learning provides to guide evolution is what can be inherited or followed by children. These "inherited or followed" factors might be gene building blocks or beneficial local search steps. These factors must be realized during a child's learning process; parents' achievements guide their children, and Baldwinian learning guides evolution.

This chapter has contributed to the understanding of Baldwinian evolution. The following behavior of children's learning, and the two main factors that affect this, have been studied. The conclusions that we have attained about learning scheme and genetic operator characteristics may provide directions toward possible applications.

For future work, the questions of what and how to inherit must be explored. In this chapter, we concluded that children following parents' learning may be beneficial. However, if they should just follow their parents, why do we not use Lamarckian evolution? How to properly apply Baldwinian evolution, and how to combine it with Lamarckian evolution, is still a problem for consideration. Designing genotypes to control the learning process beyond the boundaries of plasticity is also possible. Conversely, under a full Baldwinian scenario, learning costs affect learning potential and its realization. Further studies are thus required.

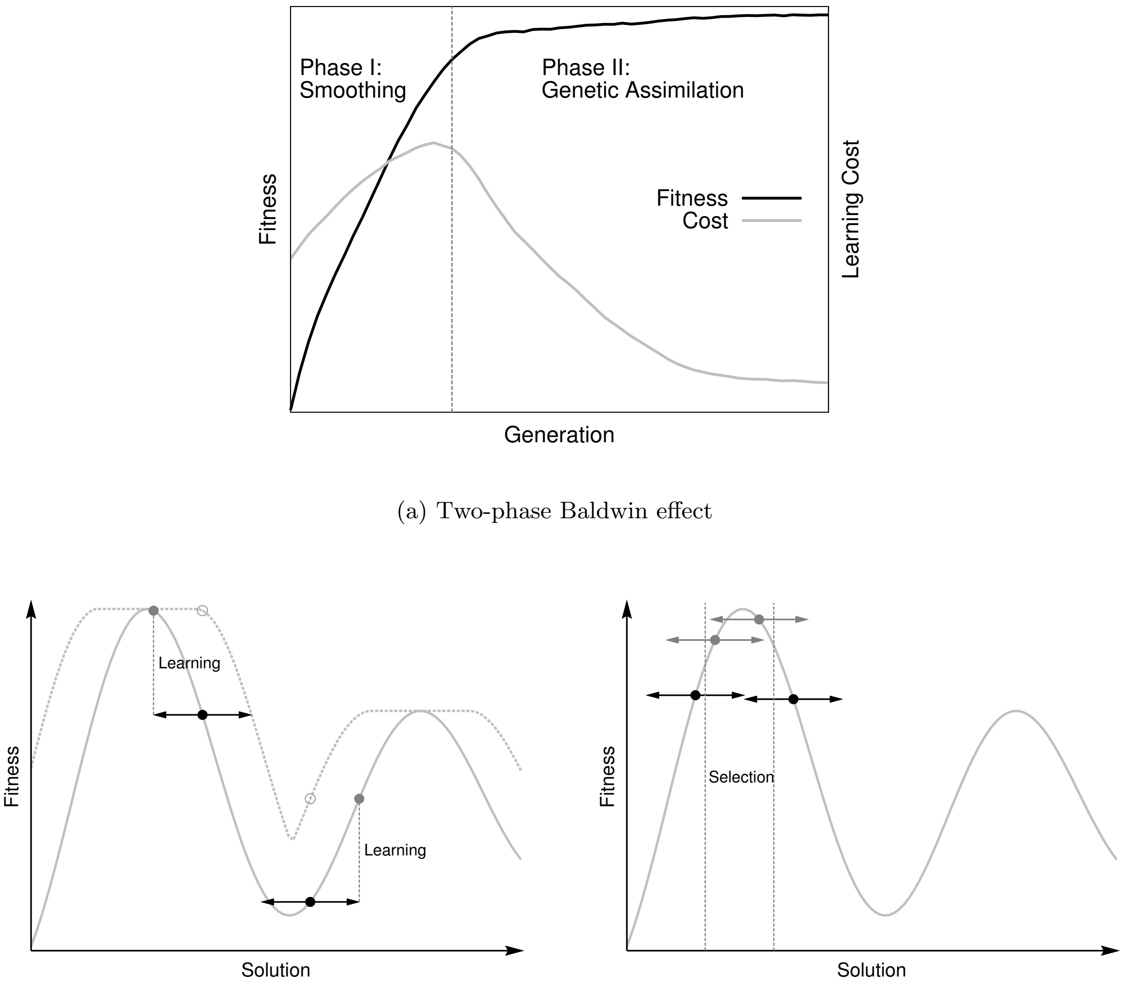
## 3.6 Conclusion

This chapter has presented a study on how individuals realize learning potential in Baldwinian evolution. The preference of children to follow their parents' learning is affected by two factors: uncertainties in learning scheme affect the learning process, and breaking of building blocks by genetic operators affect initial solutions. These two factors were shown to have different impacts. Uncertainties in a learning scheme provide noninheritable variations and substantially change the search target. As a result, learning potential is limited, searches are slow, and solutions are unfit. Genetic operators break building blocks in genotypes and provide inheritable variations. As a result, learning similarity is influenced, and exploitation is balanced with exploration. The guiding effect of Baldwinian learning on evolution is therefore implied to be that which can be inherited or followed by children. There are some further verifications in continuous optimization in the appendix of this thesis, and the results also support the conclusions found in this chapter. Our results help to further understand Baldwinian evolution, and can be a reference point for possible applications.

# Chapter 4

# Learning Intensity and Cost Punishment

In the previous chapters learning cost punishment is not included, however, it is an essential factor in Baldwinian evolution. In Baldwinian algorithms, a selective penalty on individuals with high learning costs is usually implemented by the addition of a negative term to the fitness function. This penalty is necessary for genetic assimilation, and influences the search by limiting the increase of learning intensity. Although a cost penalty can accelerate search, it sometimes inhibits the discovery of better solutions. Appropriate design of a penalty is important for building effective algorithms; however, the exact effect of penalties and how best to design them is still not fully understood. This chapter investigates the mechanisms of cost penalties in depth, focusing on their influence on learning intensity during the search phase. Our results show that genetic assimilation may start long before search convergence, and cost penalty effects may vary considerably dependent on the employed selection scheme. A selection scheme has to provide competitions of individuals learning the same traits, to guarantee the occurrence of genetic assimilation. High punishments transform the landscape significantly, and assign individuals learning the same traits different fitness. It is a substitution of converged individuals' structural competitions, but has the side-effect of inhibiting the "smoothing" effect.



(a) Two-phase Baldwin effect

(b) Smoothing effect

(c) Genetic assimilation

Fig.4.1: Two-phase Baldwin effect.

## 4.1 Introduction to the Chapter

As Hinton and Nowlan claimed [20], the Baldwin Effect consists of two phases. Fig. 1.6 shows the two phases of the search process along with their mechanisms. During the early stages of a search, individuals can perform personal refinements through learning. An individual is selected not for the direct traits encoded in its genes, but based on the optimal point in its neighborhood that can be reached by learning. This selection is equivalent to applying a "smoothing" transformation on the fitness landscape (Fig. 4.1(b)). Similarly, an individual's fitness is not its initial solution's fitness, but is assigned according to the higher values of some of its neighbors. By considering learning as a part of the evaluation process, the individual is assigned its neighbor's higher fitness such that the "actual landscape" potentially attains a higher fitness. This equivalent landscape is denoted by the dashed line in Fig. 4.1(b), and extends the original peaks to plateaus. A Baldwinian evolution on the original landscape is thus equivalent to an evolution without learning on the equivalent "smoothed" landscape. During this phase, learning considerably improves fitness, learning costs are constantly high, and the population fitness increases rapidly (Fig. 4.1(a)).

Once the majority of individuals have converged on good solutions, selective pressure on the learning cost causes "genetic assimilation". Fig. 4.1(c) shows multiple individuals reaching the same learning destination, all of which have the same improved fitness. However, a cost penalty in selection eliminates individuals with high learning costs. As evolution proceeds, individuals' initial solutions converge on the same destination, and learning costs decrease. In this process, learned traits are automatically "encoded" into the genotypes. During this phase, learning improvement is low, the learning cost decreases, and the population fitness is maintained at the same level (Fig. 4.1(a)).

This two-phase description is generally accepted in Baldwinian algorithm studies. However, a problem with the description is that learning cost penalties may have effects on the "smoothing" phase. In many Baldwinian algorithms, cost penalty is implemented through the addition of a negative term to the fitness function: $f = f_{raw} - p(b)$, where $f_{raw}$ is the learning-improved fitness, $f$ is the final fitness used during selection and $p(b)$ is the cost penalty, which is an increasing function of the learning cost/budget $b$. In conventional studies, the penalty term $p(b)$ is usually considered to be trivial and is ignored in the first phase. However, the penalty term will always have an effect on the first phase because the penalty changes the fitness function and thus the selection rule. We must therefore determine when the effects are not trivial and what their influence is.

When a cost penalty is included in Baldwinian evolution, learning intensity is affected by two opposing driving forces. The first is fitness improvement, known as the smoothing effect, which enables individuals to receive their neighbors' higher fitness values. Individuals with high learning intensity can explore large areas when learning, and thus can achieve high fitness improvements. This is a well-known episode. However, the second driving force—the cost penalty—brings further complications. Learning provides fitness improvement, but also demands high computational effort and incurs a cost penalty $p(b)$. The cost penalty degrades a "smoothed landscape", as shown in Fig. 4.2. Thus, the true equivalent landscape is not a "smoothed" one, but a "smoothed and degraded" one. If we consider that individuals have different learning intensity levels, and that learning costs are a part of, or determined by, their genotypes, then the "smoothed and degraded landscape" can be extended along the cost di-

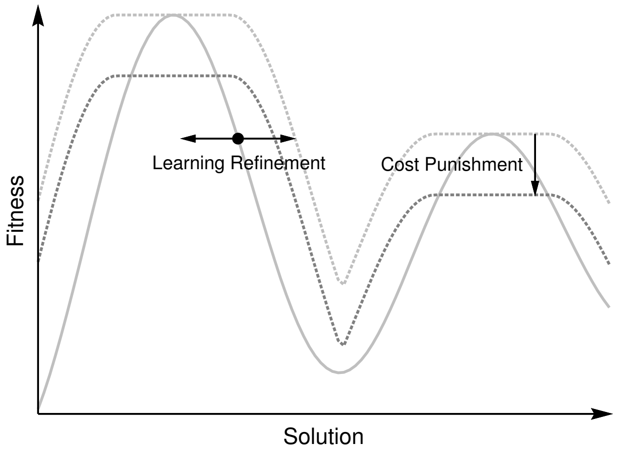


Fig.4.2: Equivalent landscape with smoothing and cost punishment.

mension (Fig. 4.3). As learning intensity increases, peaks are transformed into plateaus, but with lower altitudes.

From Fig. 4.3, it is easy to understand that a cost penalty may limit learning intensity. When the cost penalty is too high, the refinements through learning can be smaller than the applied penalties. Learning-improved fitness then has little effect on evolution, and the search becomes inefficient. In such a case, cost punishment has a considerable influence on the "smoothing" phase.

Problems can also exist even with quantitatively trivial cost penalty terms. One question is, how can a trivial cost penalty guarantee genetic assimilation? Genetic assimilation necessitates cost penalties, but the introduction of penalties does not ensure assimilation. Furthermore, if genetic assimilation does occur, will the penalty influence the "smoothing" phase at the same time? Since fitness evaluation is merely an intermediate step of the total selection operation, answers to these questions are not given by simple fitness comparisons. Landscape equivalences provide some clues, but more methods are required.

In this chapter we attempt to study what happens during selection: how learning improvements and cost penalties affect the survival of individuals during selection, when learning costs increase and decrease, and the influences of the scale of cost penalty and type of selection. Learning costs are encoded into genotypes so that the costs are independent of the other genes forming the genotype. We verify that the "smoothing" effect and genetic assimilation exist during the entire search process in Baldwinian evolution, and that the balance between them determines the learning cost dynamics. Furthermore, individuals with low learning costs survive over those with high costs by two mechanisms. Which mechanism dominates is highly dependent on the employed selection scheme. The remainder of the chapter is as follows. In

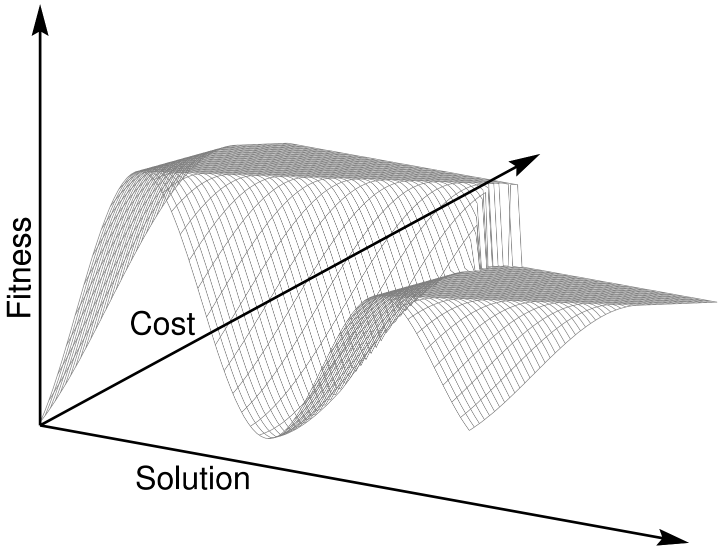


Fig.4.3: Equivalent landscape extended along the cost dimension.

Section 4.2 we introduce the NK model as our benchmark. In Sections 4.3 and 4.4, we present experiments with 2-tournament selection, and with a roulette wheel selection for comparison, respectively. A discussion on the experiment results and possible future works is given in Section 4.5. Finally, the conclusions of this study are drawn in Section 4.6.

## 4.2 Model

### 4.2.1 NK Model with Plasticity and Learning Budget

As well as we did in the previous chapters, plasticity-encoding strings are added into genotypes. Furthermore, a limitation of learning cost—which we call the "budget" hereinafter—is also included in genotypes in this study. A genotype consists of two $N$-bit binary strings and a budget $b$, as shown in Fig. 4.4. $G_I$ encodes the initial solution, which is modified throughout learning, and is evaluated according to the look-up tables. However, $G_I$ is inherited as part of the initial value of an individual's offspring, irrespective of the parents' learning. An additional string $G_P$ encodes plasticity, which is a mask that determines those bits of the learning solution that can be modified. Positions containing a value of 1 in $G_P$ denote bits in the learning solution that may be changed; otherwise, bits are fixed. Learning is then a series of such trials, and is completed once $b$ trials have been conducted.

Experiments are performed on three fitness landscapes to examine various levels of epistasis

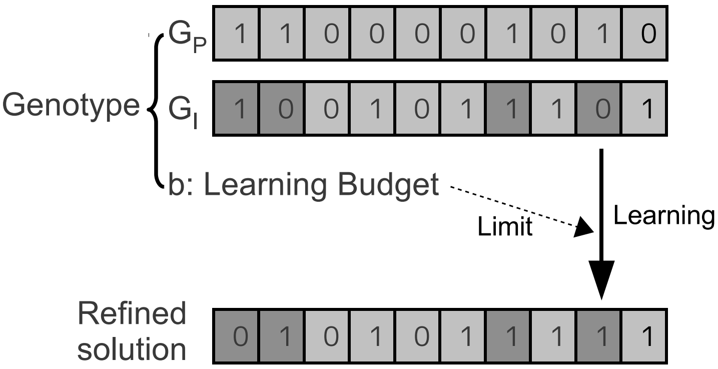


Fig.4.4: Genotype with plasticity.

Table4.1: Experiment Parameters

| | |
|---|---|
| $N$ | 20 |
| $K$ | 2/5/15 |
| Population Size | 500 |
| Crossover Rate | 0.50 |
| Mutation Rate | 0.05 |

and complexity. Several of the experiment parameters are listed in Table 4.1. Two-point crossover and one-bit mutation are employed in the experiments. Moreover, 2-tournament selection is employed in Section 4.3 and, and a roulette wheel selection is employed in Section 4.4. The learning settings are now introduced in next subsection.

### 4.2.2 Learning Settings

The learning scheme in our experiments is a series of bitwise climbing trials. An individual's learning starts from solution $S_0 = G_I$ and passes through improved solutions $S_1, S_2, \cdots$ until a pre-specified number of trials have been performed. The final solution is then the learning destination and its fitness is set as the fitness of the individual.

An example is shown in Fig.4.5 in which the position $i = 4$ is selected in the $k$-th learning trial based on the previous result $S_k$. The $i$-th (4th) bit in the solution is modified to produce a variant $S'_k$, and $S'_k$ and $S_k$ compared such that the fitter one is set as $S_{k+1}$. Such trials are repeated for different modifying positions until a given number of trials have been conducted. The final solution is then the learning destination and its fitness is set as the fitness of the individual. Note that the plastic positions in an individual's genotype, from most to least significant, are $i_1, i_2, \cdots, i_m, 1 \leq i_j \leq N.$, and the sequence of bits to be modified during learning is $i_1, i_2, \cdots, i_m, i_1, i_2, \cdots$. Referring to Fig. 4.4, the sequence is 1, 2, 7, 9, 1, 2, 7, 9, $\cdots$.

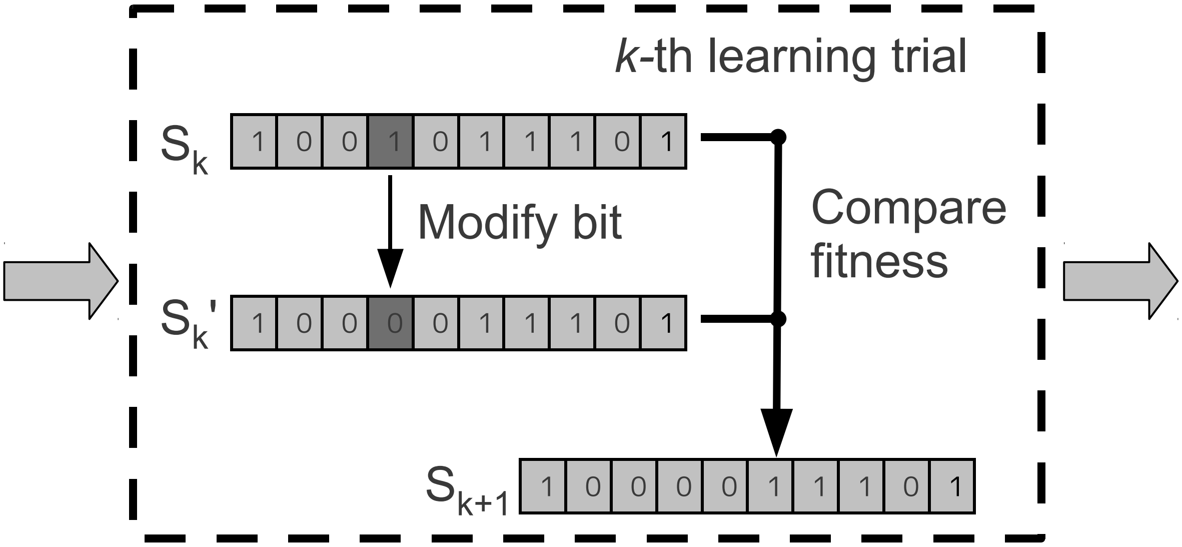


Fig.4.5: $k$-th trial in learning process.

Here, we use the encoded budget $b$ to limit the number of iterations of learning trials. Setting $b$ enables individuals to have unique learning costs, and places these costs under selective pressure. In conventional studies, learning costs may not be encoded explicitly, but instead will be related to the number of plastic bits or to other information contained in $G_I$ and $G_P$. In this chapter, we intend to isolate learning costs from these other complexities and examine how they change when viewed as an independent factor. We also apply a mutation to the encoded learning costs to maintain their diversity during evolution. When generating a new child, a random variable $X$ is added to the inherited learning budget: $b_{child} = b_{parent} + X$, where $X$ is the difference between a binomial distributed random variable and its expectation: $X = Y - 3, Y \sim B(6, 0.5)$.

Following conventional methods, we set a cost penalty as a negative term in the fitness function: $p(b) = C{\cdot}b$. Four values of the coefficient $C$ are used in our experiments: 0, $\epsilon$, 0.001 and 0.01. $\epsilon$ helps us to examine the situation when the learning cost really is trivial. The value $\epsilon$ is set to a sufficiently small value, that the difference in cost penalty between two individuals is always smaller than the raw fitness difference: $\forall i \neq j, |p(b_i) - p(b_j)| = C \cdot |b_i - b_j| < |f_{raw,i} - f_{raw,j}|$. Thus, when $C = \epsilon$, a high-level learning result always provides a fitness advantage.

## 4.3 Components of Selective Pressure

During the selection process of a Baldwinian algorithm, an individual may survive due to its good initial solution, its effective learning, its low learning cost or, as is commonly the case, a combination of these three factors. Initial solutions do not directly influence learning intensity,

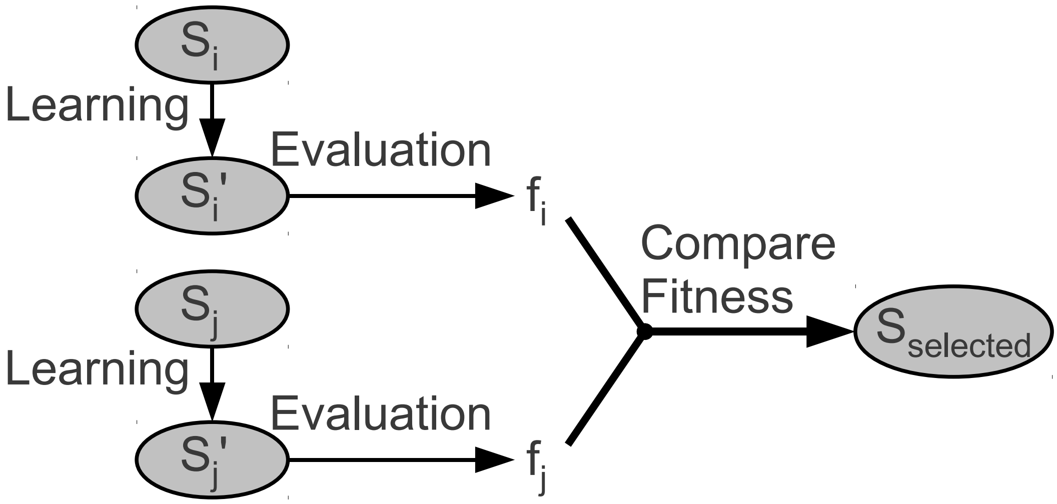


Fig.4.6: Tournament selection with size 2.

while fitness improvements and cost penalties have opposing impacts. Fitness improvements promote learning intensity increases because additional learning produces fitness improvement. Conversely, cost penalties decrease learning intensity since additional learning increases punishment. During a search, the trade-off between these two factors determines the learning intensity dynamics.

Whether an individual has been selected for its initial solution, fitness improvement or learning cost is difficult to distinguish. In fact, an individual's survival is always a consequence of all three factors. We can therefore observe an increase or decrease in learning intensity, but not the factors behind such a change. However, we can clearly recognize the influence of cost penalties when an individual survives over other individuals at the same learning destination. The reason for the individual's survival must be its cost punishment, because the raw fitness values are the same for each member. The phenomenon is what we term "genetic assimilation". If we can count such surviving individuals, we can ascertain when genetic assimilation begins, and how much it contributes to learning intensity decreases.

To analyze the effects of the above factors, we use 2-tournament selection in this section. In each step of the 2-tournament selection, two individuals are sampled from the population, their fitness values are compared, and the individual with higher fitness is selected (Fig. 4.6). These steps are then repeated until a sufficient number of individuals have been selected. This selection scheme facilitates analysis of selective pressure components. Each selected individual can be compared with its beaten opponent to determine the reason for its victory, without the influence of other individuals in the population. We can thus count how many individuals have higher learning intensity than their opponents, how many individuals have lower learning intensity

Table4.2: Search performance on $K = 2$ landscape with 2-tournament selection

| $C$ | Converging Fitness | Eval. Calls to Reach 99% Fitness | Maximum Total Cost | Eval. Calls to Reach Max. Cost |
|---|---|---|---|---|
| 0 | 0.814999<br>(±0.000591) | 68408<br>(±8430) | $\infty$ | — |
| $\epsilon$ | 0.814449<br>(±0.000564) | 67717<br>(±7675) | 5091<br>(±463) | 65066<br>(±10744) |
| 0.001 | 0.814500<br>(±0.000529) | 66270<br>(±7119) | 4467<br>(±374) | 50535<br>(±8464) |
| 0.01 | 0.814531<br>(±0.000661) | 45680<br>(±4762) | 2665<br>(±106) | 7851<br>(±3854) |

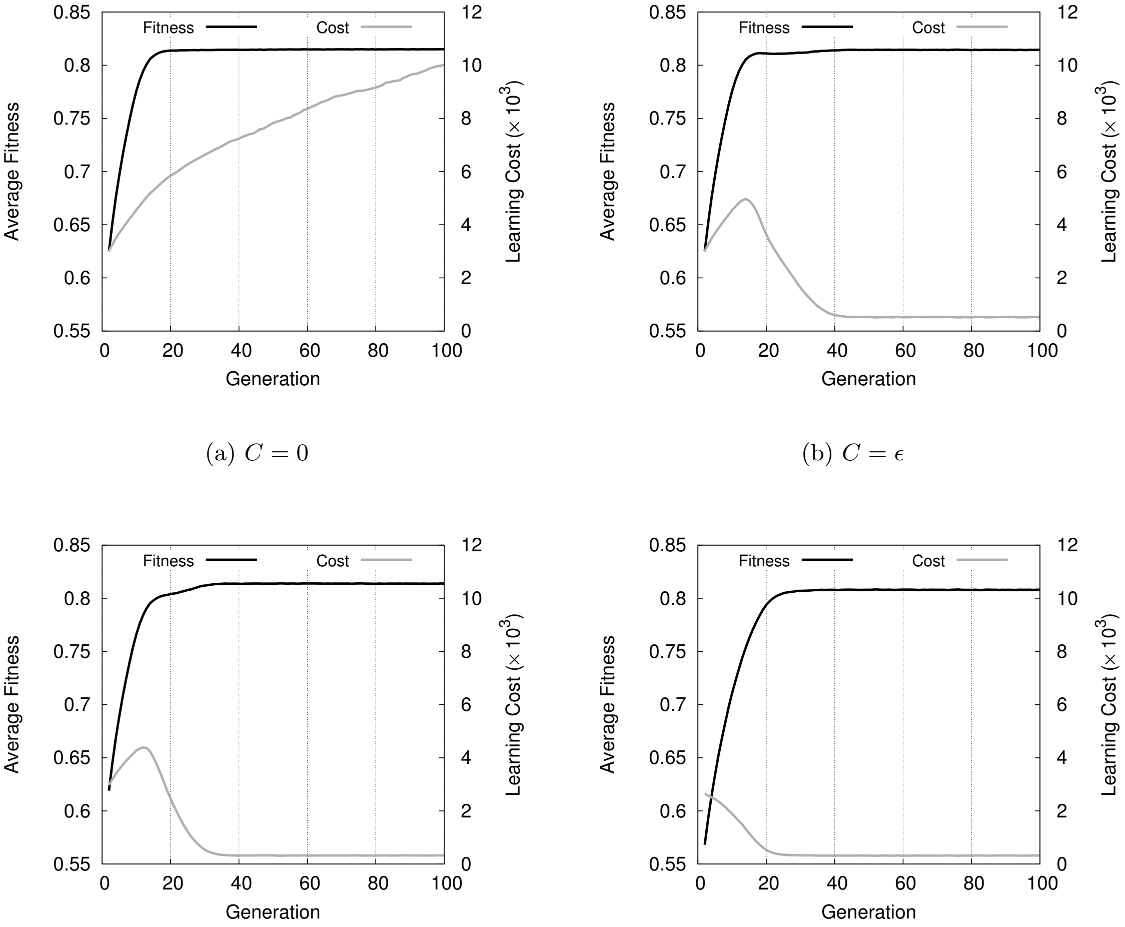
and how many individuals beat opponents at the same learning destination. The results are, of course, dependent on the unique characteristics of 2-tournament selection. What is observed here is not guaranteed to be reproduced under a different selection scheme. Experiments with 2-tournament selection at least provide insight into the underlying processes.

The following subsections introduce experiments on three landscapes. On each landscape, we examine the effects of using the four punishment coefficient values: 0, $\epsilon$, 0.001 and 0.01. We compare the search processes when using each penalty coefficient and the proportions of each type of selection survivals. All results are the averages of 50 runs.

### 4.3.1 Low Epistasis Landscape

First, we present the experimental results for a $K = 2$ fitness landscape with low-level epistasis. This is the simplest landscape in our experiments; genotype bits have few connections to each other, and optimization is relatively straightforward.

Fig. 4.7 shows the search performance when using each penalty coefficient. The black lines denote fitness values, and the gray lines denote learning costs/budgets of the total population, which are measured in terms of evaluation calls. Because of the simplicity of the landscape, all of the Baldwinian searches converge to high fitness values in about 20 generations. However, different learning intensities are attained during each search, which affects computational speeds and can be measured through the number of evaluation calls. When $C = 0$, learning is cost-free, and learning intensity grows without restriction. A high computational cost is paid for convergence, and even after convergence the learning intensity continues to increase. The mutation on the learning budget has zero expectation, and therefore this unlimited grow must be explained. The main reason may be that, with greater learning, children are more likely to repeat their parents' achievements and survive in the population. When $C = \epsilon$, the learning



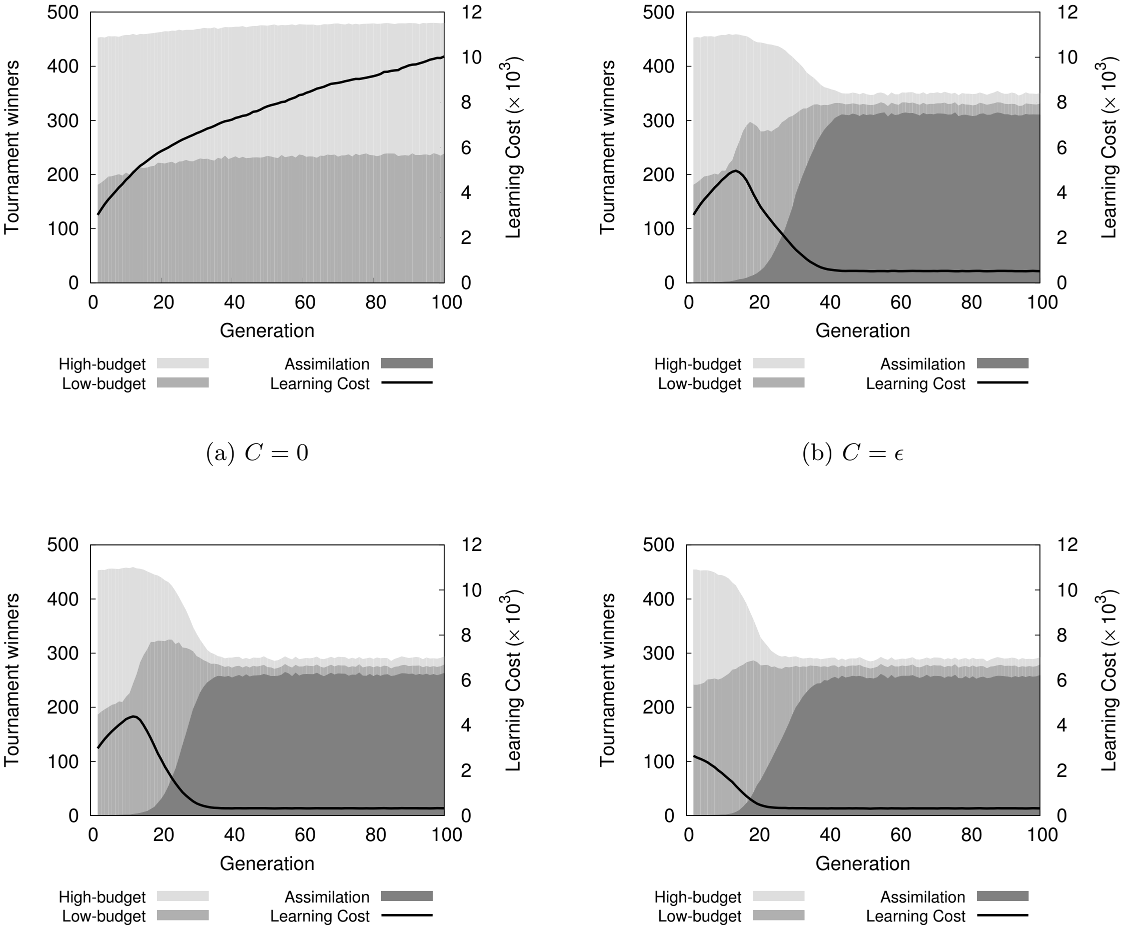
(a) $C = 0$

(b) $C = \epsilon$

(c) $C = 0.001$

(d) $C = 0.01$

Fig.4.7: Search performance on $K = 2$ landscape with 2-tournament selection.



(a) $C = 0$

(b) $C = \epsilon$

(c) $C = 0.001$

(d) $C = 0.01$

Fig.4.8: Selected individual types on $K = 2$ landscape with 2-tournament selection.

intensity is effectively limited, and is similar to the limitation provided when $C = 0.001$. When $C = 0.01$, cost punishment is high, and the learning intensity immediately begins to decrease.

Table 4.2 lists further data on search performance. Final fitness values are approximately equal when using all four punishment coefficient values. Convergence time varies according to the different limitations placed on the learning intensity, however. When $C = 0$, $\epsilon$ or 0.001, the convergence time is nearly unchanged, whereas when $C = 0.01$, convergence is faster. Peak values of the population's total learning cost are also different. When $C = 0$, the learning cost continues to increase long after convergence, and so we denote it by $\infty$ in the table. Search using $C = \epsilon$ costs more than using $C = 0.001$, and considerably more than using $C = 0.01$. These results imply that even low fitness punishment can limit learning intensity effectively under 2-tournament selection. In addition, when $C = 0.001$ or 0.01, learning costs reach their peak long before convergence. Decreases in learning costs are therefore not limited to the period after convergence, but may occur in the early stages of search.

Furthermore, Fig.4.8 shows the results of an experiment that analyzes the selective pressure components. The black lines denote the learning costs. Dark grey regions denote individuals that beat opponents with higher learning budgets at the same learning destination, which we term "assimilation winners". Moderate grey regions denote individuals that beat opponents with higher learning budgets, but at different learning destinations (i.e., not assimilation winners). We term such winners "low-budget winners". Light grey regions denote individuals that beat opponents with lower learning budgets, which we term "high-budget winners".

With zero cost punishment, genetic assimilation does not occur, and learning intensity is unrestricted. Throughout the search, even after convergence, there are more high-budget winners than low-budget winners, and the proportions of these two groups remain at a fixed level. As a result, the learning intensity continually increases at almost constant speed. When $C = \epsilon$, the situation changes. Initially, high-budget winners are in greater number than low-budget winners. But, as evolution proceeds, the number of high-budget winners falls and the number of low-budget winners grows. Moreover, assimilation winners appear before the search converges, and their number increases rapidly. After convergence, over 50% of the population are assimilation winners, and the learning cost converges to a constant level. For searches using $C = 0.001$ and 0.01, the results are similar to those when $C = \epsilon$.

This early appearance of assimilation winners implies that genetic assimilation is not limited to the period after convergence. Considering the mechanisms involved, this implication is quite natural. If individuals learn the same target phenotype and compete with each other, their cost penalties determine the selection result. Thus, we have local genetic assimilation. Genetic assimilation may even happen in initial stages of search, and its influence is then felt during the

Table4.3: Search performance on $K = 5$ landscape with 2-tournament selection

| $C$ | Converging Fitness | Eval. Calls to Reach 99% Fitness | Maximum Total Cost | Eval. Calls to Reach Max. Cost |
|---|---|---|---|---|
| 0 | 0.798539<br>(±0.003607) | 150217<br>(±77551) | $\infty$ | — |
| $\epsilon$ | 0.796337<br>(±0.004332) | 131606<br>(±29429) | 5491<br>(±574) | 89639<br>(±20203) |
| 0.001 | 0.797025<br>(±0.003093) | 134484<br>(±32258) | 4852<br>(±610) | 77036<br>(±23783) |
| 0.01 | 0.792158<br>(±0.008827) | 73805<br>(±89750) | 2640<br>(±85) | 6870<br>(±1715) |

entire smoothing phase. Because of the high population diversity during the search phase, the frequency of genetic assimilation occurrences is low, however, and so its effect is not obvious. After convergence, the diversity of population is reduced, and so genetic assimilation becomes the dominant force affecting learning intensity. From this point of view, rather than arising in two separate phases, smoothing and genetic assimilation are opposing forces that compete throughout the search process.

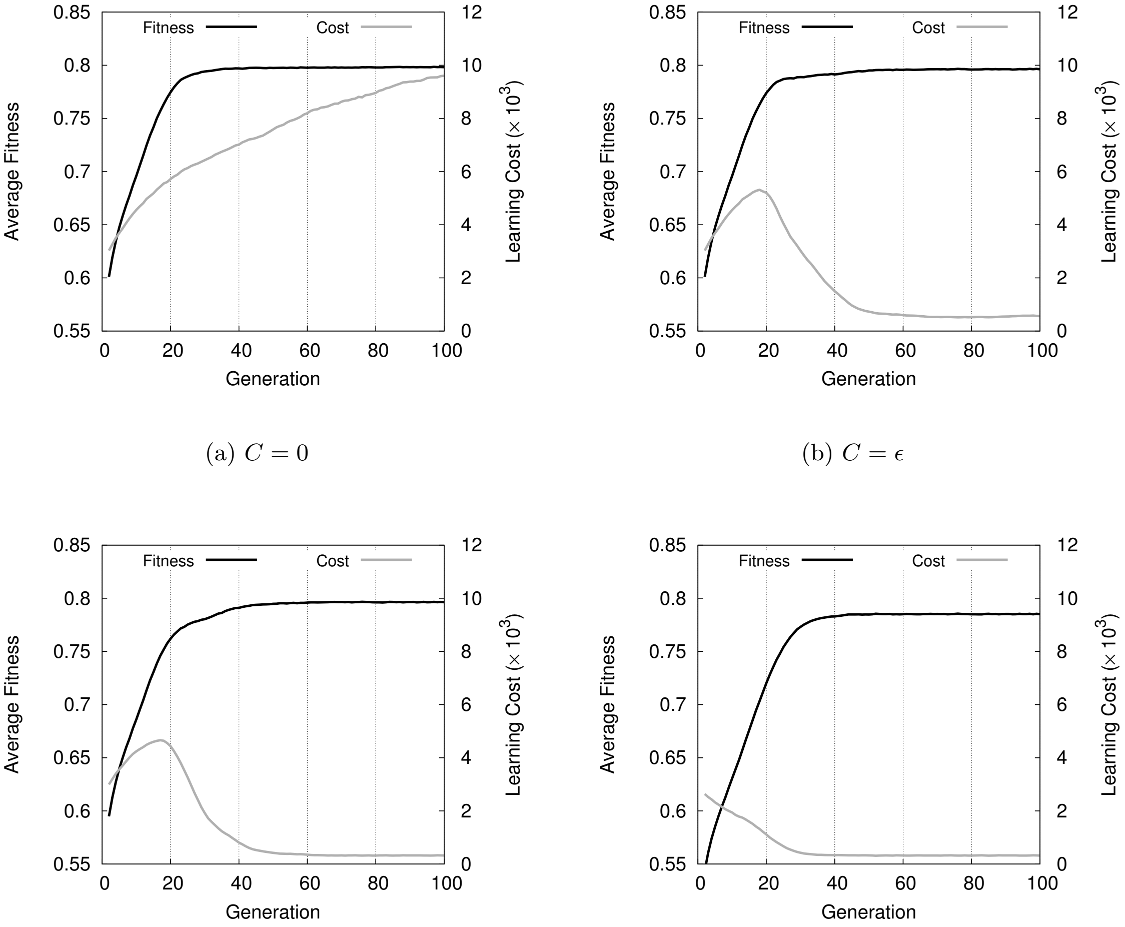
Conversely, genetic assimilation is not the only force that limits the growth of learning intensity. As mentioned in Section 4.2, a cost penalty can degrade a smoothed landscape. With a high penalty, the landscape is considerably altered, and the balance between the selective pressure components changes. From Fig. 4.8, with a high punishment coefficient $C$, high-budget winners reduce in number, whereas low-budget winners increase. The effect of a cost penalty is hence reflected in the learning intensities and search performance.

### 4.3.2 Moderate Epistasis Landscape

We next examine the results for a $K = 5$ fitness landscape with higher epistasis and complexity than the previous landscape.

Fig. 4.9 shows the search performance when using each punishment coefficient. Search is slower and final fitness values are lower on this landscape; however, the general trends of all four searches are similar to those on the $K = 2$ landscape. When $C = 0$, the learning intensity grows continuously at almost constant speed. When $C = \epsilon$, the learning intensity is effectively limited and is similar to the case when $C = 0.001$. Finally, when $C = 0.01$, the learning intensity decreases from search commencement.

Table 4.3 lists further search performance data. Final fitness values are higher under low cost punishment, but the differences between the values are small. In contrast, the variation in



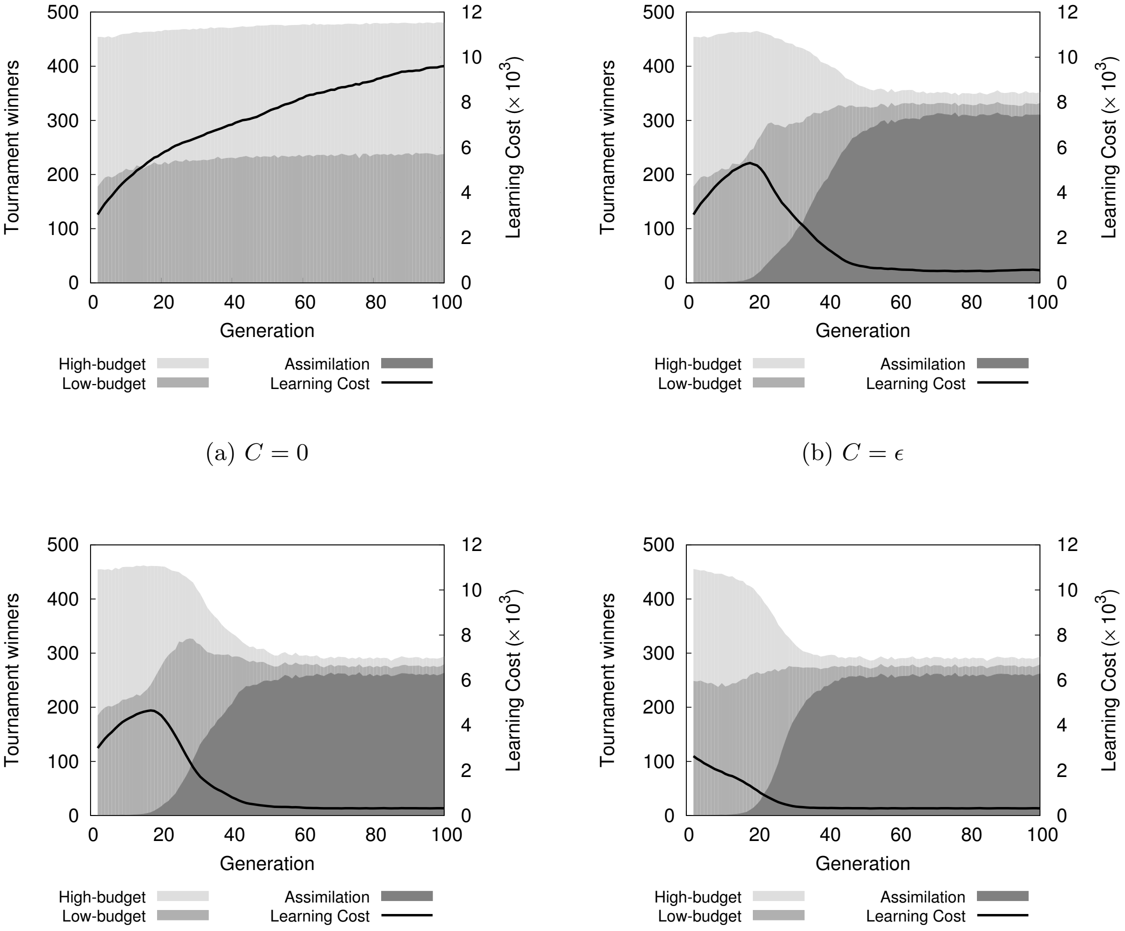
(a) $C = 0$

(b) $C = \epsilon$

(c) $C = 0.001$

(d) $C = 0.01$

Fig.4.9: Search performance on $K = 5$ landscape with 2-tournament selection.



(a) $C = 0$

(b) $C = \epsilon$

(c) $C = 0.001$

(d) $C = 0.01$

Fig.4.10: Selected individual types on $K = 5$ landscape with 2-tournament selection.

Table4.4: Search performance on $K = 15$ landscape with 2-tournament selection

| $C$ | Converging Fitness | Eval. Calls to Reach 99% Fitness | Maximum Total Cost | Eval. Calls to Reach Max. Cost |
|---|---|---|---|---|
| 0 | 0.743802<br>(±0.017854) | 279967<br>(±104931) | $\infty$ | — |
| $\epsilon$ | 0.741298<br>(±0.017781) | 278825<br>(±115613) | 6577<br>(±1071) | 164819<br>(±64131) |
| 0.001 | 0.740881<br>(±0.017076) | 231537<br>(±79510) | 5499<br>(±710) | 141442<br>(±47828) |
| 0.01 | 0.718408<br>(±0.020291) | 98473<br>(±23769) | 2571<br>(±92) | 6485<br>(±957) |

convergence time becomes more distinct on this landscape. Search using $C = 0$ results in the slowest convergence, searches using $C = \epsilon$ and 0.001 have almost the same speed, whereas the convergence time when $C = 0.01$ is half of that when using the other three coefficients. Peak values of the population's total learning cost, and the numbers of evaluation calls required to reach the peaks, are similar to those on the $K = 2$ landscape.

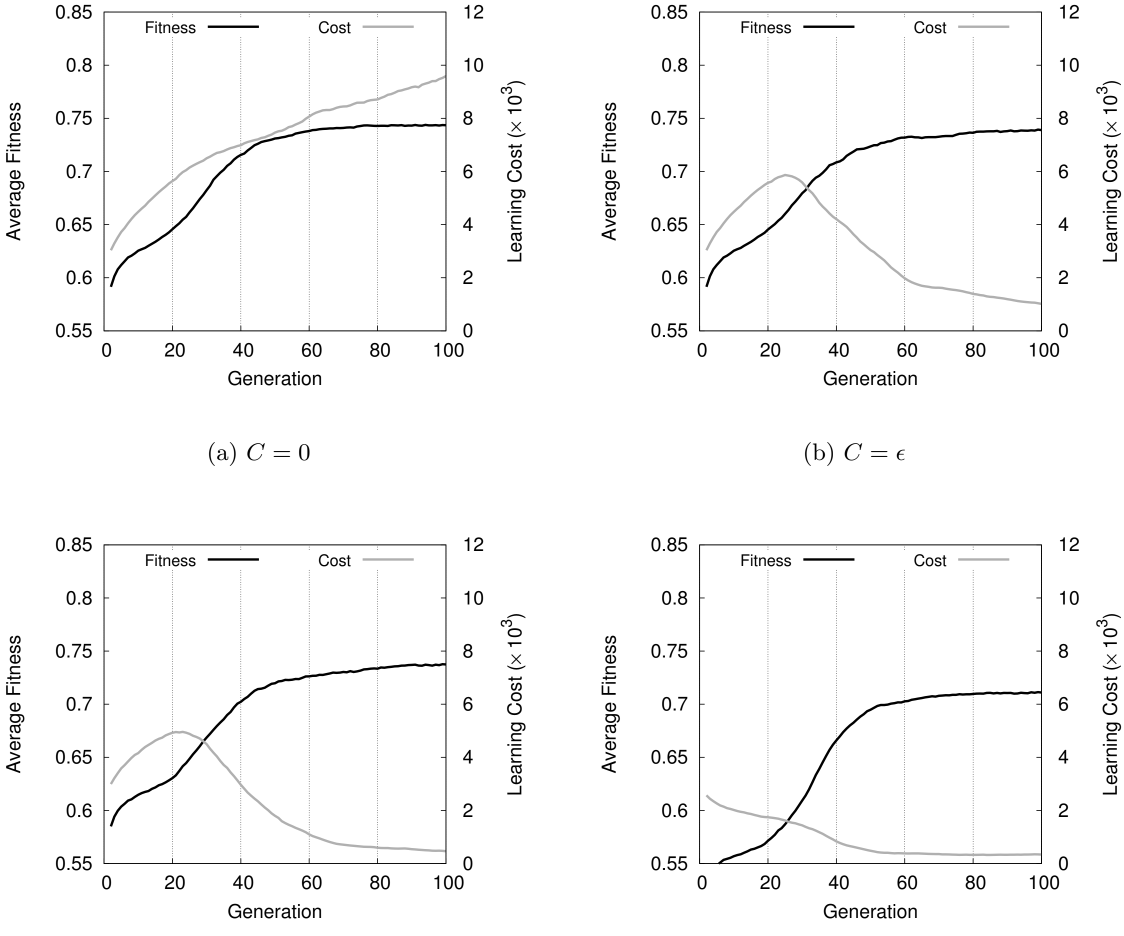
Fig. 4.10 shows the numbers of selected individuals according to winner types. The figures are similar to those on the $K = 2$ landscape, but the numbers vary more slowly through the generations. We see that genetic assimilation can start in the early stages of search, and that higher punishments increase the number of low-budget winners. The results once more imply that smoothing and genetic assimilation have influence throughout a search, and that high punishment substantially transforms the landscape.

### 4.3.3 High Epistasis Landscape

We finally examine a highly complex $K = 15$ fitness landscape. The epistasis of the landscape is hence very high. Any crossover or mutation in such a case is likely to break building blocks and prevent children from following their parents. Thus, finding good solutions is difficult.

Fig. 4.11 shows the search performance when using each penalty coefficient. All of the Baldwinian searches converge slowly on this complex landscape, and the final fitness values are low. The behaviors of the four searches are similar to those on the other two landscapes, but the differences between them are more distinct. In particular, when $C = 0.01$, the final fitness is considerably lower than when using the other three coefficients.

Table 4.4 lists further search performance data. The variation in convergence time is very clear on this landscape. Convergence is slowest when search using $C = 0$, is still slow when using $C = \epsilon$, and is slightly faster when using $C = 0.001$. Conversely, the convergence time



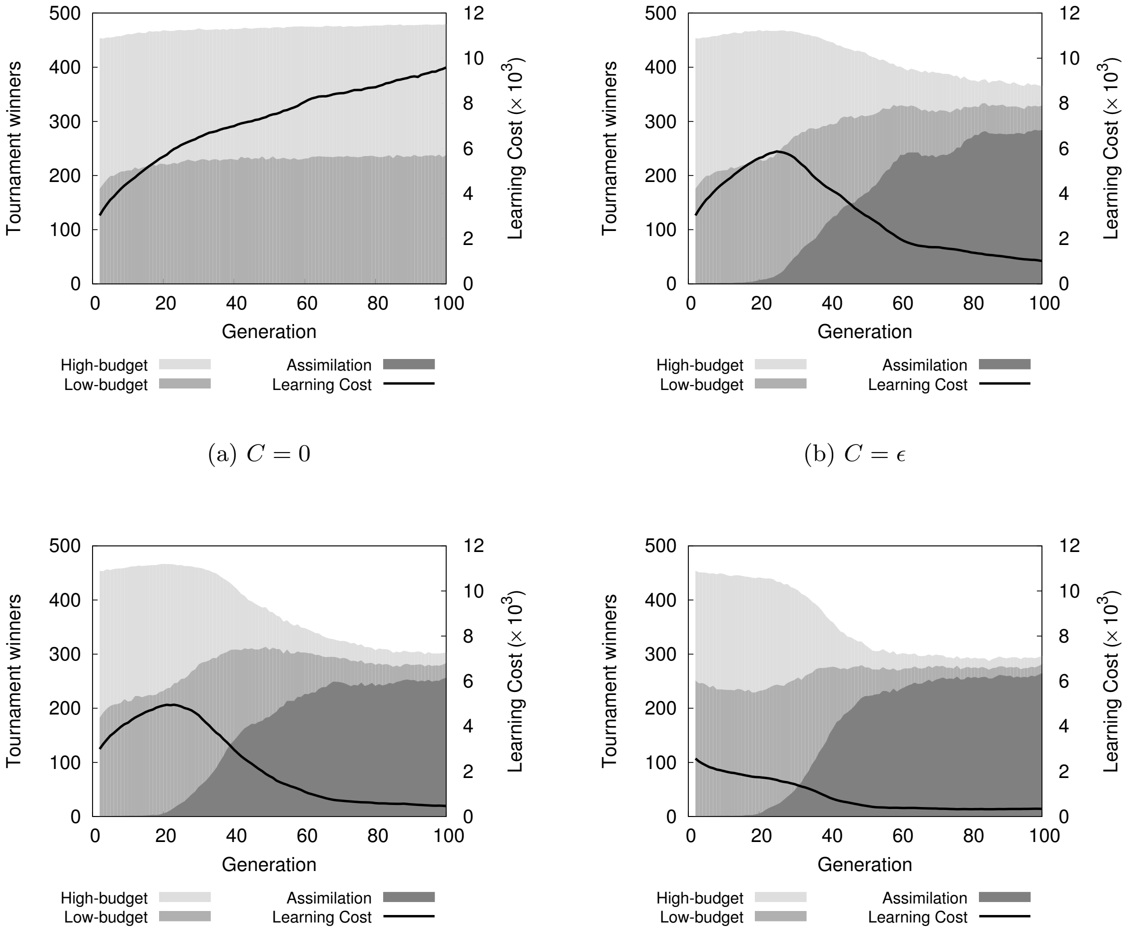
(a) $C = 0$

(b) $C = \epsilon$

(c) $C = 0.001$

(d) $C = 0.01$

Fig.4.11: Search performance on $K = 15$ landscape with 2-tournament selection.



(a) $C = 0$

(b) $C = \epsilon$

(c) $C = 0.001$

(d) $C = 0.01$

Fig.4.12: Selected individual types on $K = 15$ landscape with 2-tournament selection.

when $C = 0.01$ is less than half of that when using the other three coefficients. Comparisons among learning cost peak values, and those among required evaluation calls, show similar trends as on the other two landscapes. The difference here is that the costs when $C = \epsilon$ and 0.001 are higher than before.

Fig. 4.12 shows the numbers of selected individuals according to winner types. The figures are similar to those on the other two landscapes, although the numbers vary even more slowly. We confirm the appearance of genetic assimilation in the early stages of search, and the increase in low-budget winners brought by using high punishments.

In this section, we compare the effects of four penalty coefficients. When $C = 0$, the learning intensity is unrestricted and therefore continuously grows. When $C = \epsilon$, selection provides competitions between individuals reaching the same learning destination. Convergence is not accelerated to any great extent, but learning intensity is constrained and genetic assimilation occurs. When $C = 0.001$, the situation is similar to that when $C = \epsilon$. When $C = 0.01$, punishment is high, and learning intensity is highly constrained. Search is fast in this case, but the final fitness is low.

We draw two conclusions from these results. First, genetic assimilation occurs throughout a search, although its dominance comes after convergence. Second, selection limits learning intensity by providing competitions between individuals with the same learned traits, and by transforming the landscape to the advantage of low cost individuals. Moreover, with 2-tournament selection, competitions between individuals with the same learned traits are suitably frequent to restrict learning intensity and cause genetic assimilation. Searches are therefore unlikely to improve even with higher punishments.

## 4.4 A Condition for Genetic Assimilation

In the previous section, we employed a 2-tournament selection scheme. That selection offers many opportunities for individuals with the same learned traits to compete against each other, resulting in evident genetic assimilation even for a trivial punishment coefficient. However, such competitions are not guaranteed under all selection schemes. Under a scheme that does not promote such competitions, the prospect of genetic assimilation becomes doubtful. In this section, we utilize a roulette wheel selection in place of the 2-tournament selection, and examine the changes that occur in searches.

As shown in Fig.4.13, in roulette wheel selection, every individual is mapped onto a segment, and all these segments form the roulette wheel. Selection of an individual is achieved by performing a uniform sampling on the roulette wheel, where an individual mapped to a longer

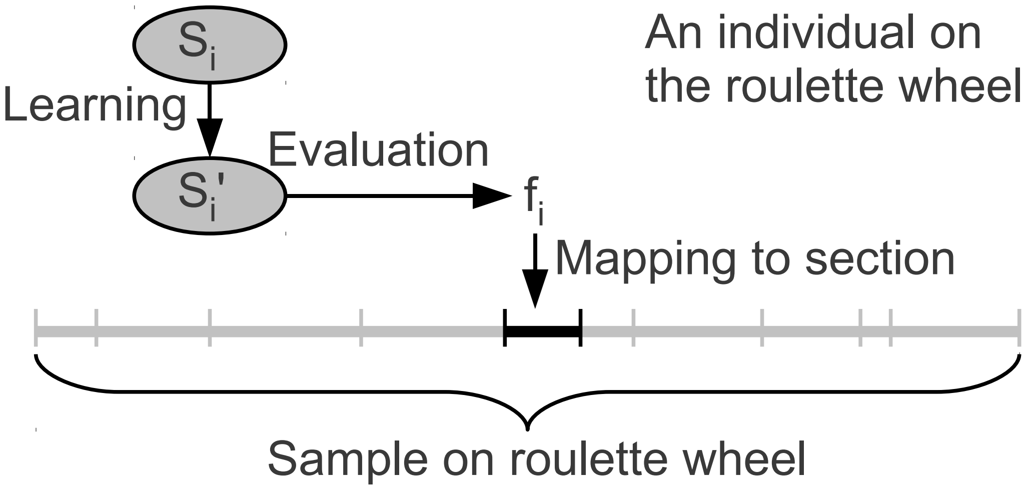


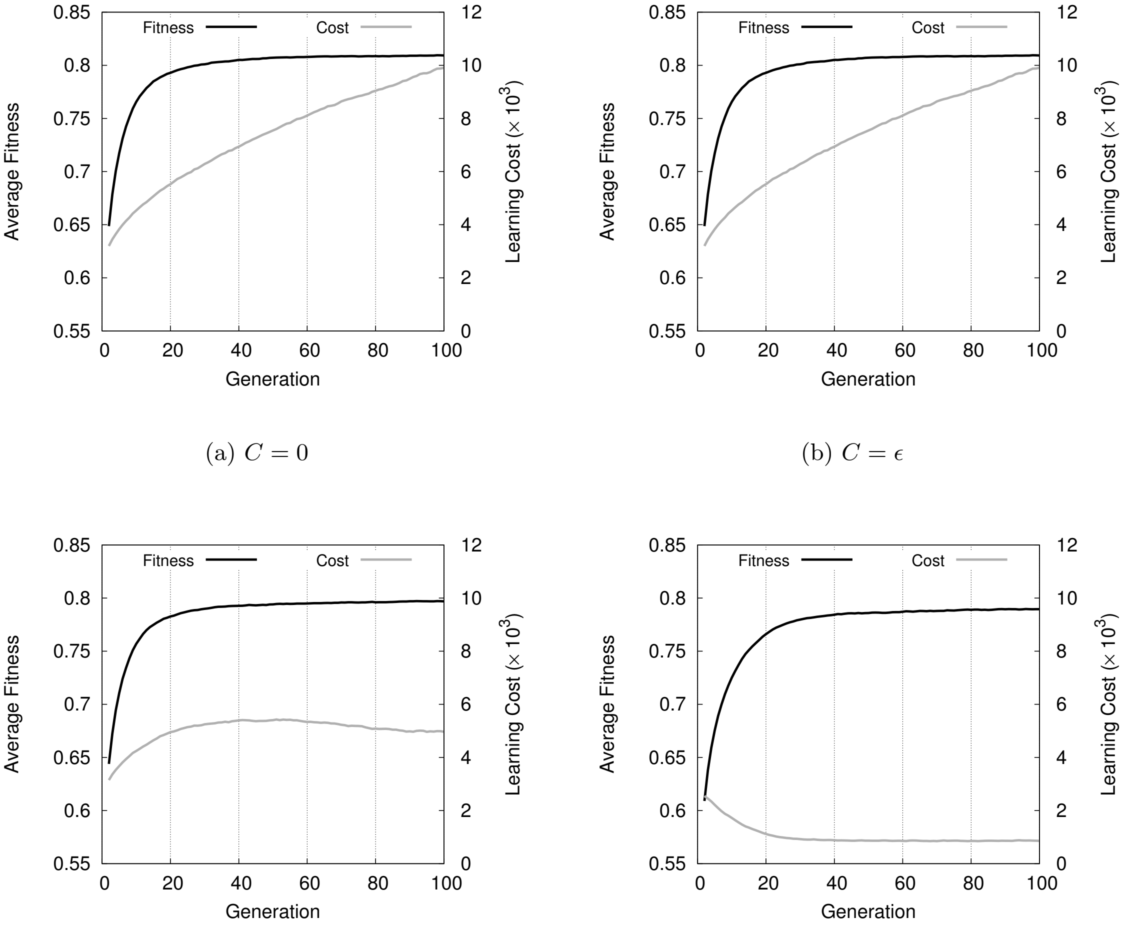
Fig.4.13: Roulette wheel selection.

segment has greater chance of survival. The length of a segment $l_i$ is calculated according to fitness $f_i$. In our experiments, we set $l_i = (max(f_i - 0.5, 0))^2$, to ensure an appropriate level of selective pressure.

In contrast to 2-tournament selection, roulette wheel selection considers all individuals simultaneously. Each individual competes with all of the other members of the population. Therefore, when several individuals reach the same learning destination and the punishment is low, they are mapped onto segments of approximately equal lengths. They are thus treated equally during selection, and competition among them is unlikely to occur. Furthermore, judging which selective pressure component provides the largest contribution when an individual is selected is difficult. We therefore do not categorize survivals by winner types, and examine only search performance.

Experiments are performed on three landscapes, and we compare the effects of using four punishment coefficient values: 0, $\epsilon$, 0.001 and 0.01. All results are the averages of 50 runs.

Fig. 4.14 shows the search performance with roulette wheel selection when using each penalty coefficient on the $K = 2$ landscape. All of the Baldwinian searches converge to high fitness values in about 30 generations. When $C = 0$, the learning intensity still grows without limitation. When $C = \epsilon$, the search performance is almost the same as when $C = 0$. In contrast to the performance with 2-tournament selection, this “trivial” penalty does not restrict learning intensity or cause genetic assimilation. Because the segment lengths, and therefore the survival probabilities, are only slightly changed. When $C = 0.001$, the learning intensity is limited after convergence, but the decrease in learning cost is very slow and genetic assimilation is insubstantial. Furthermore, the final fitness is lower than that when $C = 0$ and $\epsilon$. When $C =$



(a) $C = 0$

(b) $C = \epsilon$

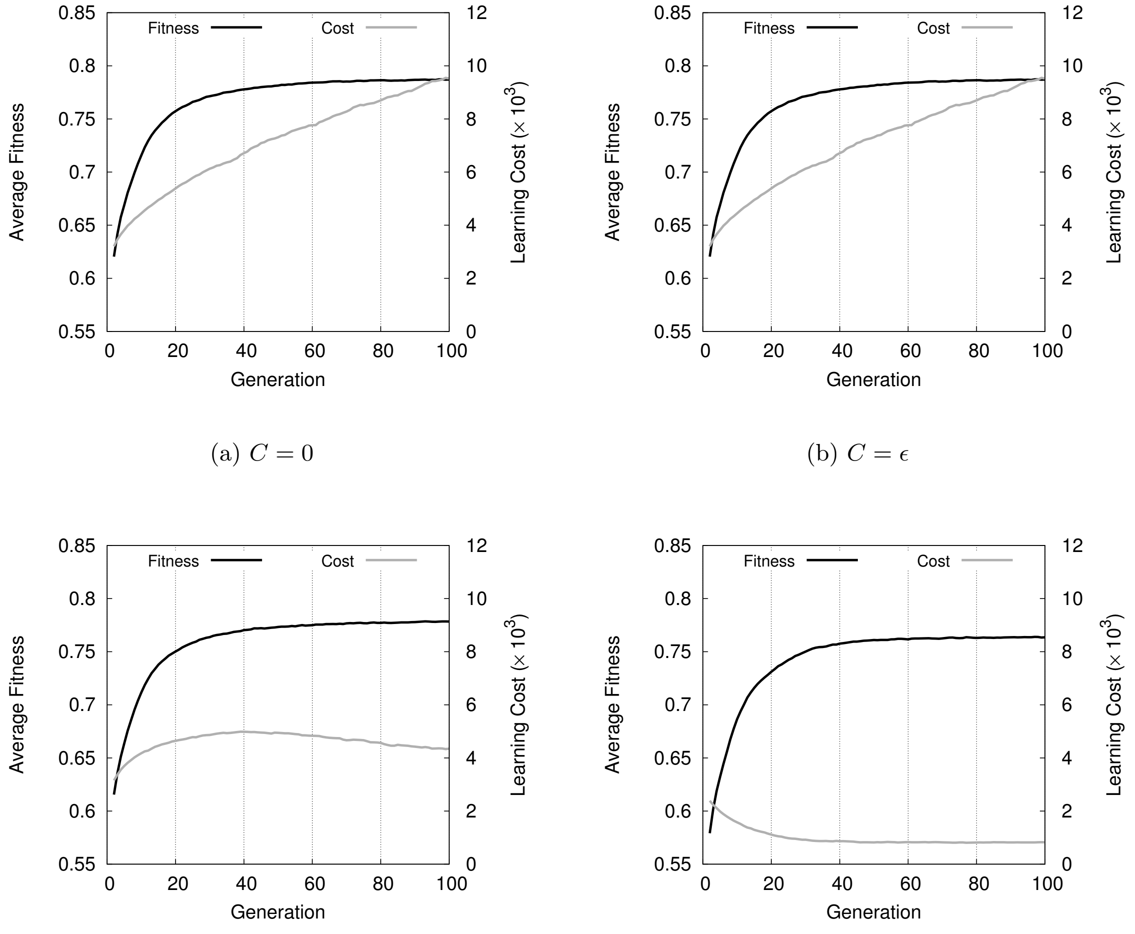
(c) $C = 0.001$

(d) $C = 0.01$

Fig.4.14: Search performance on $K = 2$ landscape with roulette wheel selection.

Table4.5: Search performance on $K = 2$ landscape with roulette wheel selection

| $C$ | Converging Fitness | Eval. Calls to Reach 99% Fitness | Maximum Total Cost | Eval. Calls to Reach Max. Cost |
|---|---|---|---|---|
| 0 | 0.810296<br>(±0.001348) | 173933<br>(±46262) | ∞ | — |
| $\epsilon$ | 0.810296<br>(±0.001348) | 173933<br>(±46262) | ∞ | — |
| 0.001 | 0.808358<br>(±0.001550) | 145233<br>(±40240) | 5900<br>(±510) | 334082<br>(±157153) |
| 0.01 | 0.807941<br>(±0.002513) | 64848<br>(±18006) | 2563<br>(±126) | 7395<br>(±2232) |



(a) $C = 0$

(b) $C = \epsilon$

(c) $C = 0.001$

(d) $C = 0.01$

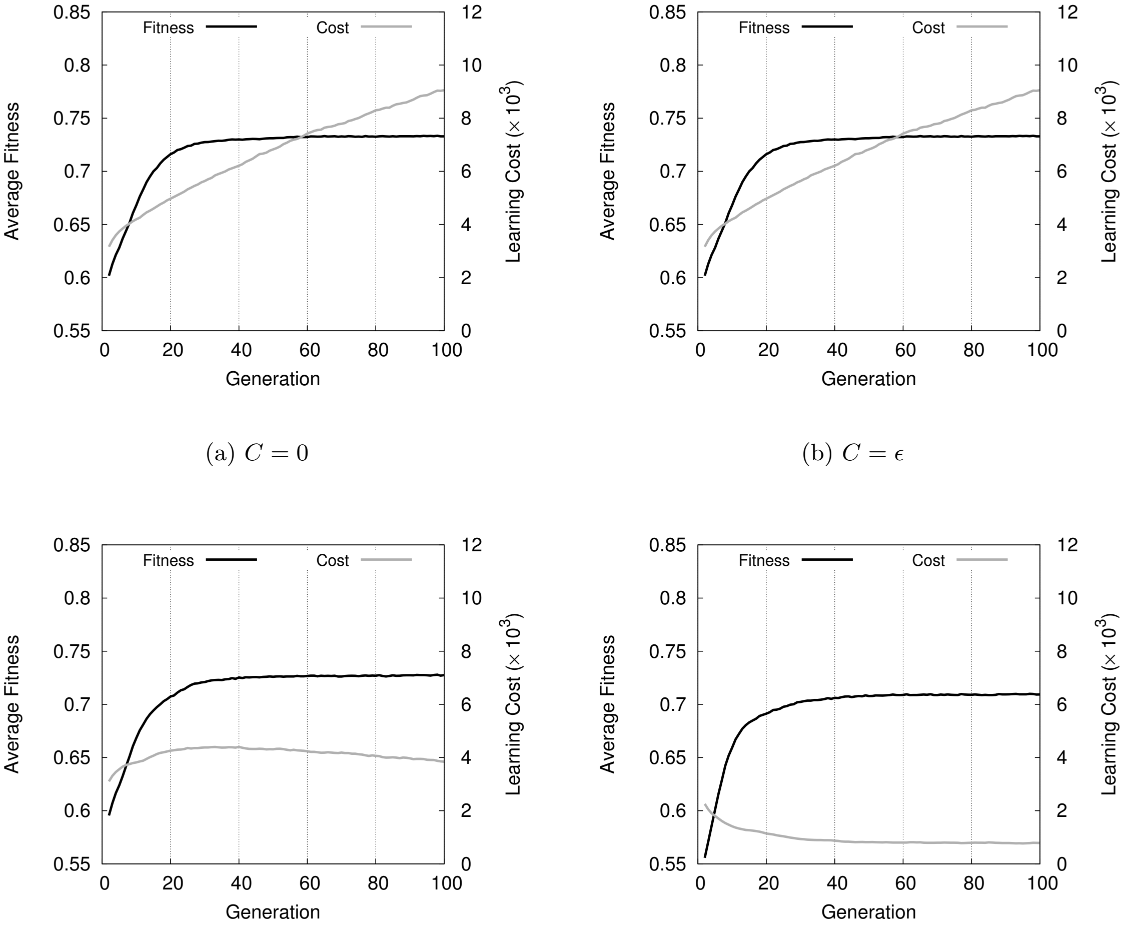
Fig.4.15: Search performance $K = 5$ landscape with roulette wheel selection.

0.01, genetic assimilation occurs, but learning is strongly inhibited from search commencement such that the final fitness is even lower than in the other cases. Table 4.5 lists further search performance data. As the penalty coefficient $C$ is increased, search becomes more rapid, but the final fitness value is reduced.

Fig. 4.15 shows the search performance on the $K = 5$ landscape. Lower fitness values are attained by all searches on this landscape, and the trends of the four searches are similar to those on the $K = 2$ landscape. When $C = 0$ or $\epsilon$, learning intensity grows without limitation. When $C = 0.001$, learning intensity is limited but genetic assimilation is insignificant, and the final fitness is reduced. When $C = 0.01$, genetic assimilation occurs, but learning is strongly inhibited and the fitness is poor. Data in Table 4.6 further supports these conclusions. The four Baldwinian searches behave similarly on the $K = 2$ and $K = 5$ landscapes.

Table4.6: Search performance on $K = 5$ landscape with roulette wheel selection

| $C$ | Converging Fitness | Eval. Calls to Reach 99% Fitness | Maximum Total Cost | Eval. Calls to Reach Max. Cost |
|---|---|---|---|---|
| 0 | 0.787858<br>(±0.009423) | 257816<br>(±130333) | $\infty$ | — |
| $\epsilon$ | 0.787858<br>(±0.009423) | 257816<br>(±130333) | $\infty$ | — |
| 0.001 | 0.788634<br>(±0.008212) | 220978<br>(±171354) | 5534<br>(±763) | 320361<br>(±238885) |
| 0.01 | 0.780647<br>(±0.013147) | 67811<br>(±36818) | 2427<br>(±226) | 7669<br>(±6224) |



(a) $C = 0$

(b) $C = \epsilon$

(c) $C = 0.001$

(d) $C = 0.01$

Fig.4.16: Search performance on $K = 15$ landscape with roulette wheel selection.

Table4.7: Search performance on $K = 15$ landscape with roulette wheel selection

| $C$ | Converging<br>Fitness | Eval. Calls to<br>Reach 99% Fitness | Maximum Total<br>Cost | Eval. Calls to<br>Reach Max. Cost |
|---|---|---|---|---|
| 0 | 0.733877<br>(±0.018830) | 122482<br>(±102508) | $\infty$ | — |
| $\epsilon$ | 0.733877<br>(±0.018830) | 122482<br>(±102508) | $\infty$ | — |
| 0.001 | 0.736112<br>(±0.018030) | 92570<br>(±45346) | 5059<br>(±1085) | 224143<br>(±169212) |
| 0.01 | 0.726722<br>(±0.022471) | 58487<br>(±80389) | 2318<br>(±356) | 9302<br>(±9072) |

Fig. 4.16 shows the search performance on the $K = 15$ landscape, and further performance data are listed in Table 4.7. Because of the high epistasis of this landscape, fitness values become even lower, and search becomes very slow. However, the trends of the four searches are the same as on the other two landscapes. A trivial penalty no longer limits the learning intensity, and a trade-off is found between search speed and final fitness.

In this section, we have compared the effects of four penalty coefficients under a roulette wheel selection scheme. When $C = 0$, the learning intensity continuously increases. A trivial penalty when $C = \epsilon$ does not the limit learning intensity, and the performance is the same as when $C = 0$. When $C = 0.001$, learning is limited, but genetic assimilation is not obvious. The final fitness attains a lower value than that when $C = 0$ or $\epsilon$. When $C = 0.01$, learning is inhibited and genetic assimilation occurs, but the final fitness is reduced considerably.

The search performance under roulette wheel selection is considerably different from that under 2-tournament selection. First, a trivial penalty when $C = \epsilon$ no longer restricts learning. The lack of competitions between individuals learning the same traits means that genetic assimilation does not occur, and learning intensity always increases. Second, although learning is limited and genetic assimilation can occur with a high penalty, the price for this is a reduction in final fitness. The landscape is considerably transformed in this case and the fitness of individuals with high learning costs becomes particularly low. All individuals in the population are affected by this transformation, including those learning the same traits, who are treated differently according to their learning costs. However, the majority of individuals have unique learning destinations before convergence, and penalties mainly constrain learning rather than promoting genetic assimilation.

In summary, competitions between individuals learning the same traits is a necessary condition for genetic assimilation to occur. Such competitions can be provided by the selection scheme, or result from landscape transformations caused by enough cost punishment. However, cost penalties also inhibit learning.

## 4.5 Discussion

In the experiments, the effects of selective pressure components were examined. We found that "smoothing" and genetic assimilation occurred at all stages of a search. These two key factors of the Baldwin effect are thus two competing driving forces, rather than two separate search periods. The two-phase explanation does, however, describe the search behaviors when one of the forces dominates, but does not describe how the trade-off between them varies over time. By considering the factors as two dynamically changing forces, the fact that genetic assimilation sometimes initiates long before convergence is not surprising result.

Investigating the condition for genetic assimilation, we verified that learning cost punishment is necessary for, but does not guarantee, genetic assimilation. Genetic assimilation arises from competitions between individuals learning the same traits, and the preference for such competitions varies substantially according to the selection scheme. Under tournament selection schemes, competitions are performed between pairs or among small groups of individuals. Therefore, even if there are only a few individuals reaching the same learning destination, they have the opportunity to compete with one other, and trivial punishment is sufficient for genetic assimilation to occur. Conversely, under roulette wheel selection schemes based on fitness, an individual always compete with all other member of the population, and survival depends only on its fitness. Competitions between individuals learning the same traits are not naturally provided by such a selection scheme. Fitness values are almost unchanged with a trivial penalty in this situation. Thus, individuals reaching the same learning destination are treated similarly during selection, and genetic assimilation cannot occur.

A relatively high punishment of cost does, however, induce genetic assimilation under a roulette wheel selection scheme. When punishment is suitably high, individuals reaching the same learning destination are treated differently, according to their learning costs, and compete with one another. Nevertheless, high punishment inhibits learning, and may prevent the search from finding a better solution. To ensure genetic assimilation, therefore, utilizing selection schemes that intrinsically provide competitions among individuals learning the same traits, such as tournament selection schemes, is better than relying on a high learning cost penalty that influences all individuals.

This chapter has contributed to the understanding of Baldwinian evolution. The effects of learning cost penalties on selection results and search performance have been studied. We found that “smoothing” and genetic assimilation continuously influence a search, and that the potential for cost penalty to cause genetic assimilation is highly dependent on the selection scheme. Hence, a cost punishment function that is universally appropriate does not exist. A cost penalty is a constituent factor of the entire selection process, and its effect cannot be considered in isolation from the selection scheme.

For future work, continuous problems must be addressed. In the experiments, we utilized a discrete benchmark, and thus we could easily recognize individuals converging around the same peak. Such recognition in continuous problems is difficult. The influence of genetic assimilation and the cost penalty's landscape transformation may become unintelligible for a continuous problem. This is one reason for our choice of the NK model. However, we expect that similar phenomena are found in continuous problems. On the other hand, continuing with the current study, we can investigate how to measure competitions between individuals learning the same traits, and how to control these competitions to build efficient Baldwinian algorithms.

## 4.6 Conclusion

This chapter has presented a cost-performance study for Baldwinian evolution. By examining the effects of cost penalties on learning intensity, we found that that “smoothing” and genetic assimilation are dynamic driving forces that affect the entire search process, rather than arising in two separate periods. Furthermore, whether a cost punishment causes genetic assimilation is highly dependent on the selection scheme's provision of competitions among individuals reaching the same learning destination, A cost punishment's design must therefore be matched to the characteristics of the selection scheme. Our results improve the understanding of Baldwinian evolution, and provide a reference point for possible future applications. As mentioned in the discussion part, the measurement of similarity is a complex problem, thus we do not have corresponding experiments examining cost penalties in the appendix.

# Chapter 5

# Discussion

In the previous chapters, we presented three sets of experiments, and collected some conclusions about Baldwinian evolution's mechanisms. The findings provide new mechanism knowledge, as well as design rules that may be useful in application attempts. In this chapter, we introduce how these three parts connect to each other, the new aspects of viewing the evolution-learning hybridization, and what we have found beyond conventional studies. We also discuss how these ideas can be extended further, to reveal more about Baldwinian evolution. We follow the spirit of analysis, study isolated algorithm components instead of entire algorithms, and study individual dynamics instead of group behaviors.

## 5.1 From Algorithms to Components

Baldwinian evolution is a complex hybridization of population-based global search and individual local search. Individuals in the population perform their unique learning processes, then are selected by their learned fitness, but in the next reproduction operation, their children receive only their chromosomes as the starting points of learning. There are some genes in the chromosomes that control or influence the learning behavior, such as plasticity codes. There are also factors influencing selection other than refined fitness, such as learning cost penalties. A Baldwinian algorithm is a combination of all these components, and is supposed to work in dynamic environments.

Considering all these components together, Baldwinian evolution becomes a complex scenario. Many driving forces influence at the same time, and what we observe is just the output of the whole system. We can describe and explain the performance of the whole system, but it is difficult to figure out how it happens, which components of the algorithm is responsible for the occurrence, and how we can attain expected behaviors by tuning the algorithm components. As far as we consider them together, we are not decoupling them, and what we attain is knowledge about how they work together, not effects of each component.

Of course, even without decoupling these components, there is much to study with Baldwinian evolution. Designing appropriate genotypes for real world problems, which assigns suitable parts of the solution to evolution and learning; optimizing or adapting parameters of the algorithm, in order to enhance search performance; or discussing the information contents maintained by genotypes and the environment, to propose new models explaining biological Baldwinism theory: there are many possible topics, and many studies are already conducted. These works enrich the knowledge about Baldwinian evolution greatly, however, they are not solving all the problems. The fundamental components are considered as a single entity. They are labelled altogether as "Baldwinian evolution", and the studies tend to find the characters of the entity, or describe the entity from new aspects, whileas keeping the mechanisms inside as a black box. Our studies are trying to open the black box, in order to find possible answers to some fundamental questions, and possible alternative explanation to known facts.

There is a question, that when we divide the "Baldwinian evolution" entity and go along with only some of the parts, the attained algorithm is no longer "Baldwinian evolution". This is a fair comment, but the division is necessary. A wheel is only a part of an automobile, and when the wheel is taken off, it no longer functions as a part of the automobile. Study and improve the wheel is not contributing to the automobile, however, if we give the wheel a more durable tire, it contributes to an automobile when assembled to it. This is similar with our studies on

Baldwinian algorithms' components. When we conduct experiments without plasticity codes, or without learning cost penalties, the algorithms are not exactly full Baldwinian evolution. However, they contain the necessary gears of Baldwinian evolution. Implementing them isolated enables us to find out their own influences, and the same gears are likely to work in the same way when combined with more other components.

Humans are curious. They ask why when encountering fresh phenomena in their life, and try to find simple and universal explanations, according to the psychological manner of understanding and memorizing. As their knowledge increases, old answers become new questions. Human's knowledge is like a growing tree: it stretches the branches towards the sun, as well as spreading the roots deep into the ground. It is especially apparent with mathematics: Peano describes natural numbers as labels corresponding to five axioms, but later set theory mathematicians still feel it not fundamental enough, and move the foundation deeper to set cardinalities; Newton explains function derivatives as velocities, however, mathematicians find it not precise, and replace it with Cauchy's dynamic limit process description. There seems to be no ultimate end for the foundations of our theories, and we always have to analyze the current foundation and chop it into tinier pieces or concepts. This thesis is such a trial, dividing the whole Baldwinian algorithms into their constructing pieces.

We also have a practical reason for dividing Baldwinian algorithms into basic components. Although in the natural world, organisms' evolution and learning is always influenced by all the components together, in evolutionary computation, we design and implement our algorithms piece by piece. In algorithms, we can tune one component without changing others, and ultimately, we do not have to make the algorithms exact counterparts to biological evolution. To solve real world problems, we just need to imitate the nature, extract the effective mechanisms and implement them. When we implement a learning scheme or a crossover operator, we have to know about the certain part's effect, rather than the effect of the whole set.

We decide to study isolated components of Baldwinian evolution, and Fig. 5.1 shows the involved components. In Chapter 2, we build the smallest system only including the rule "learned traits are not inherited", and find how such a rule brings the fact of children learn to catch up with their parents. Also in Chapter 2 plasticity codes are added into genotypes, and the result shows that such inheritable learning indications can help in search, but cannot cancel the fact of children have to pay learning costs catching up. Then in Chapter 3, we substitute the learning scheme and crossover operator of Chapter 2's system, investigate these two factors used by default in Chapter 2. In Chapter 4, learning cost penalties are added into evolution and learning budgets as part of genotypes to Chapter 2's system, and study effects of the new components.

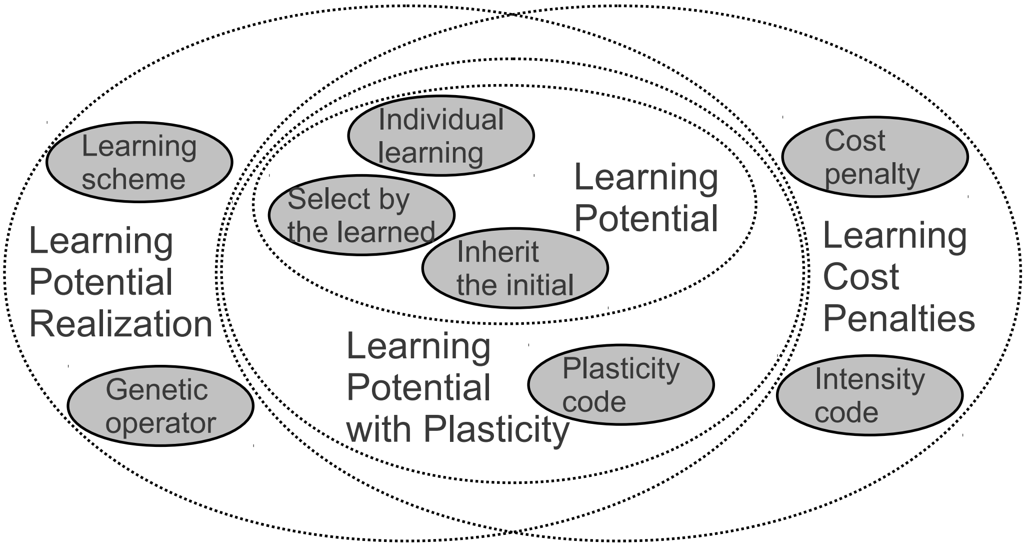


Fig.5.1: Baldwinian evolution components studied in this thesis.

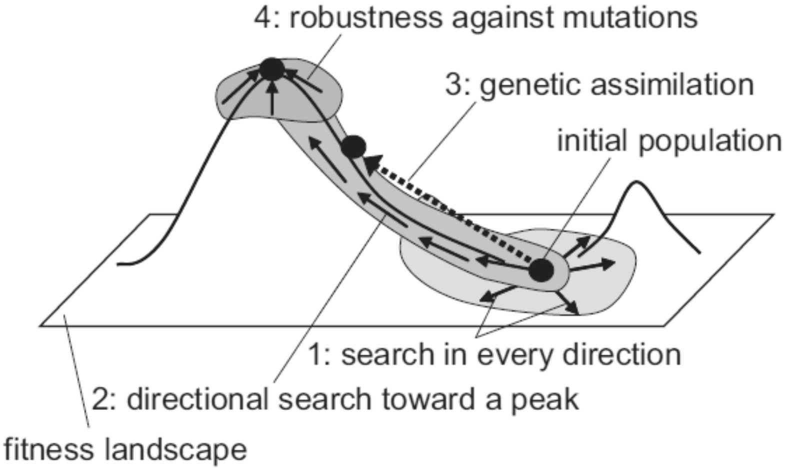

4: robustness against mutations
3: genetic assimilation
initial population
1: search in every direction
2: directional search toward a peak
fitness landscape

Fig.5.2: The roles of learning on fitness landscape (by Suzuki *et al* [51]).

We study the components one by one, and try to avoid influence from other components. We also conduct our experiments on static landscapes, to avoid complexity from environmental dynamics or noise. Such studies are not showing Baldwinian algorithms' advantages comparing to Lamarckian algorithms or evolutionary algorithms without learning, because the problem is not selected for Baldwinian evolution, and the examined algorithms are even completed Baldwinian evolution. However, they are the most basic composing factors, and are likely to perform in the same way in complicated situations.

In fact, by analyzing the basic components' effects, our results can provide some new explanations to known phenomena. These phenomena are not fully explained, or we can present a

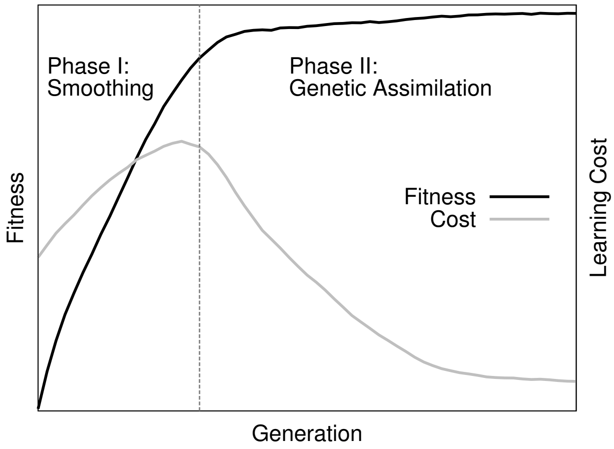


Fig.5.3: Two phase process of the Baldwin effect

new way of understanding. For example, Suzuki reported that Baldwinian evolution may be not exactly two phases, there can be a climbing period before convergence [51]. Fig. 5.2 shows his four-phase description. Referring to this thesis, using the concept of "learning potential", there can be a more fundamental and universal explanation. Children are always catching up with parents in learning, rather than explore further. That is why the population performs like climbing slowly with a limited learning capability, and why after convergence they learn against mutations: they are doing exactly the same thing of following parents, the various behaviors are results of the population's different positions on the fitness landscape. Furthermore, the two-phase description can be extended to four phases, then it is possible to have even more periods. However, replace the idea of dividing the process into consequent periods, there is also a method to analyze the conflicting driving forces and find a universal view of all the phases, as mentioned in Chapter 4. At least, our research provides alternative view points to understand known phenomena, and enriches the diversity of research.

Both the two-phase description (Fig. 5.3) and the four-phase description (Fig. 5.2) can explain the process of Baldwinian evolution, however, neither of them can be the only effective way. Each explanation has its emphasizing aspects and simplifies some other parts of the scenario. Depending on the landscape, algorithm, and topic of concern, different explanations are needed. This thesis is presenting such an alternative explanation to the process of Baldwinian evolution, and we expect it useful in search efficiency issues.

The latest experiments in this thesis, i.e. the experiments on learning cost penalties, have a relatively complete version of Baldwinian evolution. However, it does not mean the finish of studying Baldwinian algorithms' components. Similar experiments can be conducted on other problems, for example dynamic landscapes, to verify the findings. Contents of genotypes can be discussed further, beyond plasticity and learning budgets, to select or design more beneficial components. There is much to do with the consideration of analysis, and such studies are promising to build a deeper foundation for our understanding of Baldwinian evolution.

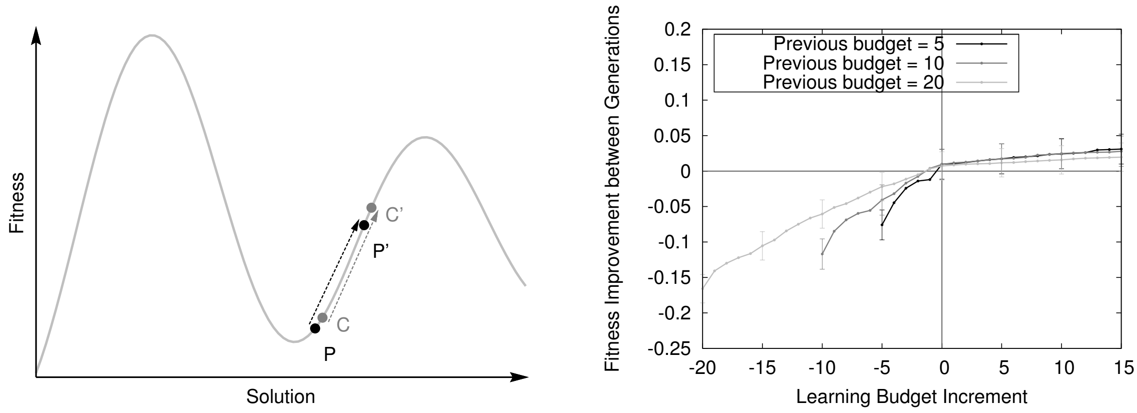
## 5.2 From Groups to Individuals

Our special view point, is not only studying constructing components instead of the whole Baldwinian evolution, but also investigating individual behaviors instead of group performance. Evolution occurs on groups of candidate solutions, and diverse individuals enables the group to change and compete pursuing high fitness. By analyzing individuals' behaviors, we explain the performance of the group.

In evolutionary or memetic algorithms, there is always a population, i.e. multiple candidate solutions. Through selection, crossovers and mutations, and possibly learning, the population evolves, and eliminating poor fitness individuals pushes the group towards high fitness. This is the mechanism of evolutionary search, and a necessary condition for fitness improvements to happen is: there are different individuals in the population that by eliminating the low fitness part of them, the average quality of the population rises. Furthermore, crossovers and mutations increase diversity of the population and enable such selection improvements last till convergence. The effectiveness of evolutionary search is based on the group with diversity. How this diversity appears, and how individuals are similar or different to each other, then become natural questions.

Adding individual learning and limiting the effect only in selection makes Baldwinian evolution more complicated than Lamarckian evolution or evolution without learning. Beside genotype diversity, randomness in learning schemes make the final fitness even more diverse, and very difficult to predict. In this case, studying the evolution process by only conventional methods and conventional means of measurements may be insufficient. It is possible that some phenomena are observed, but we cannot figure out which component in Baldwinian evolution causes them, and how to modify the algorithm to tune such phenomena. The solution is to divide the algorithm into tinier pieces: if observing the group is not enough to explain, then compare the individuals.

In optimization studies, such as continuous function optimization, it is common to measure the best fitness of the population. This is direct, and is often what we need in solving real world problems. However, for investigations of evolution's mechanisms, this index might be not enough. Evolutionary search is not hill climbing, and the best solution of the current generation is not necessarily related to the best solution of the previous generation. Each individual is



(a) Compare learning of children and their parents

(b) Record improvements step by step

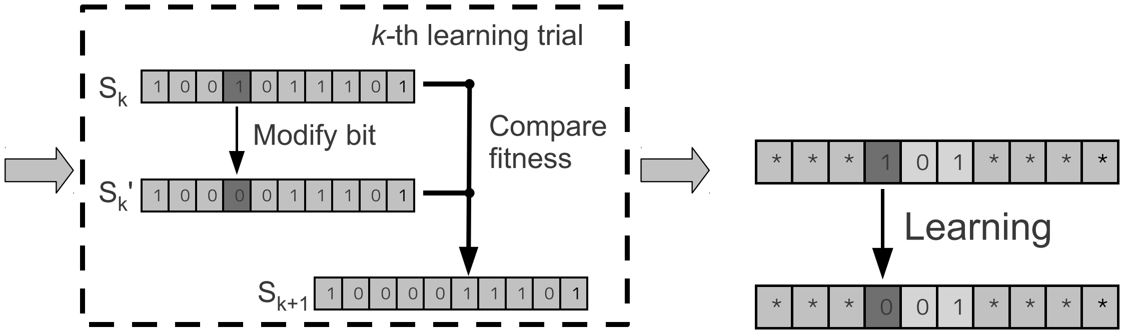
Fig.5.4: Step-wise cost-performance analysis.

produced based on the gene pool determined by the previous selection, its constructing genes may come from multiple parents, and the parents are not necessarily to be the best. The increase of best fitness is substantially a side effect of the whole population's quality. Measuring the best fitness can reflect the performance of the evolution, however, it is only one aspect of the evolution improvements.

This problem is especially obvious in Baldwinian evolution. Children have to learn to catch up with their parents, and when they fail to repeat the parents' achievements, they may even end up with a lower fitness than the parents. This is not rare in Baldwinian algorithms, for learning is not a simple repetition. In such a case, best fitness becomes insufficient to describe the population's move. This is why we always measure average fitness values in our experiments. Because of crossovers and mutations, the average values are not likely to converge to the exact optimum, and such values are not showing how good a solution we have found to solve the problem, however, we can trace the move of the group by average fitness.

Measuring average fitness is only the start, we study the population's behaviors from many smaller scopes than before. The first change is measuring learning improvements step by step. Step-wise fitness improvement is not a new concept in learning schemes that are sequences of iterative operations, however, taking this view to Baldwinian learning, leads to our first works in Chapter 2.

Baldwinian algorithms are much slower than Lamarckian algorithms. This fact is well-known, but most researchers have not explained more than the basic fact that "Baldwinian learning's results are not inherited". This is a correct explanation, however, hardly helps us in applying Baldwinian algorithms. Our study in Chapter 2 provides a more detailed cost-performance



(a) Step-wise learning trials

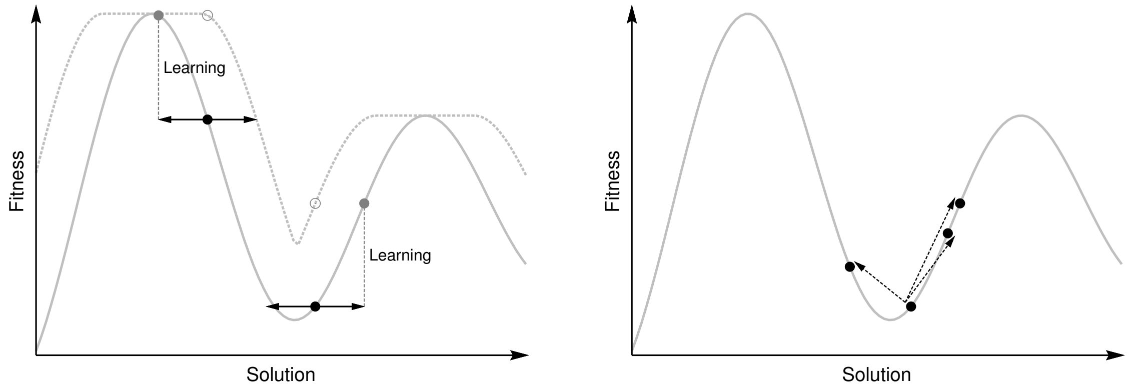
(b) Similarity by neighborhoods

Fig.5.5: Compare children's and their parents' learning steps.

analysis on the question, and presents an alternative, maybe more effective explanation to the phenomenon. We record average fitness improvements of each learning step, and compare to the fitness of the previous generation, such as in Fig. 5.4(b). By such comparisons, we find the exact effects of each step in the learning sequence, and confirmed that children have to pay the same learning efforts as their parents did, to catch up with the parents, and make some further improvements. The fitness improvements beyond their parents are much lower than the improvements during the whole learning process, which implies Baldwinian learning's main role is to maintain the potential inheritance, rather than exploring on the landscape.

This conclusion is an alternative explanation to the role of Baldwinian learning, and the difference from conventional explanations is that it provides suggestions to search efficiency issues. Inherited learning potential is attained by certain learning processes. This fact implies that if the the learning process is changed from outside settings, such as selecting different individuals to perform learning, or changing the learning budget significantly and frequently, the search performance will fall down. This is the global effect of collecting individual behaviors. Taking conventional explanations, it is also possible to reach such conclusions. However, our analysis provides a shortcut.

In Chapter 3, we go even further, and measure similarities of children's and their parents' learning steps, in addition to fitness comparisons. As shown in Fig. 5.5, in each learning step, the change in fitness is only related to some of the substrings or neighborhoods. Counting such neighborhoods enables us to judge whether a child is performing similar learning as that of its parents'. Such comparisons confirm our claim of the learning similarity changes caused by learning scheme or crossover operator substitutions, and are a necessary part of the study on learning potential's realization. Based on the learning similarity measurement applied to



(a) Smoothing effect

(b) Individual difference

Fig.5.6: Global simplification and divergence of individuals.

individuals one by one, we confirm the population's preference of following parents, and find rules for the whole search's performance.

A universal measurement of learning similarity is nearly impossible, and our measurement in fact makes use of the special structure of the NK model. It cannot be used in other problems, such as the continuous benchmark functions in our appendix. However, it is still an advance. By analyzing individual behaviors, we again explain the performance of the group. If we only have the learning schemes and crossover operators and have to judge the similarity level only from their designs, it would not be so precise and reliable.

Chapter 2 and Chapter 3 provide an alternative explanation to extend the "smoothing" metaphor. The conventional description of Baldwinian evolution's search phase is a static global simplification, as shown in Fig. 5.6(a). Considering the equivalent landscape as a "smoothed" landscape, has the hidden assumption that one initial phenotype is mapped to a single fitness value, or, if admitting dynamics or noises, mapped to a predictable varying value or a stochastic distribution. In other words, one phenotype is mapped to one entity, one point on the surface of the equivalent landscape. If the learning process is completely determined by the initial phenotype, there is a single mapped fitness value, and the metaphor simplifies the scenario properly. However, usually it is not the case. Learning schemes have uncertainties, as shown in Fig. 5.6(b), one initial solution corresponds to multiple possible refined solutions at random. Of course, in this case the equivalent landscape can be designed as with random noises, but this metaphor is not making the problem easier to deal with. It is replacing one kind of difficulties with another, without revealing the inside rules. We attempt to figure out more about what happens with such random learning schemes, and that leads to our comparisons between children

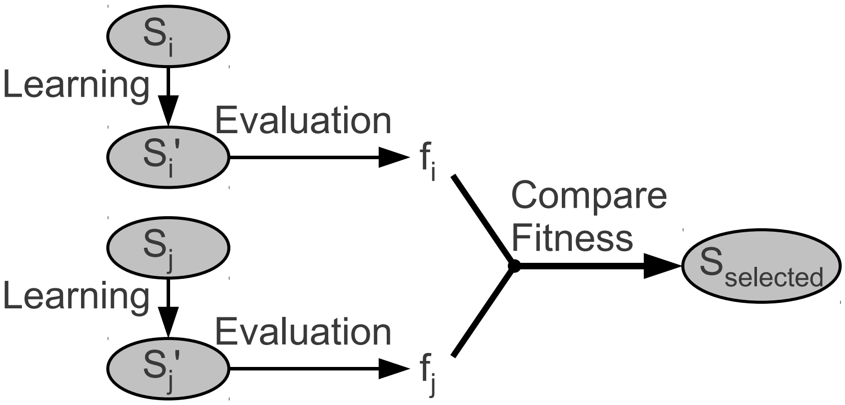


Fig.5.7: Tournament selection with size 2.

and their parents, which bring back the ignored divergence of individuals. Changing the view point, we see more.

Chapter 4 presents another microscopic measurement. We classify selection survivals by their reasons of winning in tournaments. This is also a special measurement, guaranteed by the structure of the employed 2-tournament selection (Fig. 5.7). Such a classification helps us to recognize composing factors of the selective pressure, and how these factors work as conflicting driving forces. The corresponding explanation of Baldwinian evolution's process is thus the influence of dynamically changing forces, instead of the conventional two-phase description. The two-phase description is an effective model, however, it simplifies the scenario too much, and our explanation are intended to complete some of the necessary parts.

In fact the conclusion of Chapter 4 provides more significant directions to application attempts. Learning cost punishment is a direct search efficiency controlling module, and the conclusion provides a way to avoid unnecessary cost-performance conflicts. Only when we investigate what happens to the individuals in selection, we can describe the composing factors of selective pressure, and consequently find what causes genetic assimilation whileas what inhibits learning improvements.

In this thesis, we measure individual behaviors to explain the group's performance, instead of measuring global indices directly. We compare children's fitness improvements through learning with their parents', compare children's learning steps with their parents', and categorize survivals by their reasons of victory in selection. By comparing between generations and between individuals, we reveal mechanisms of the whole search.

In our studies, we employ microscopic measurements on individuals. Some of them are designed according to the special structure of the problem and the algorithm, such as the learning similarity and the selection survival categories. Although they cannot be used invariably in some other problems or algorithms, the concept of monitoring individuals' various behaviors is emphasized. Depending on the search scenario, similar measurements can be designed, and collected data will contribute to the understanding of the mechanisms. If we cannot draw a universal conclusion for the general cases, we would suggest, starting from some instances also makes an advance.

We studied some instances of optimization in this thesis. The main body contains experiments on the NK models, and the appendix contains some on continuous benchmark functions. These studies are not proposing that in all situations Baldwinian algorithms will perform like this. We can never cover absolutely "all" possible situations in any research. The result just imply that, in the scope of our examinations, Baldwinian algorithms perform as we concluded, and considering the structure, they are expected to perform similarly in other situations. More experiments under other situations can be conducted, and investigating by other individual-wise analysis is possible. There is much that can be done.

# Chapter 6

# Conclusion

In this thesis, Baldwinian evolution's mechanisms are studied, in order to attain in-depth understanding of the theory, and ultimately to design effective and efficient algorithms for real world tasks. Differing from conventional researches, we divide algorithms to components, and groups to individuals. The change in the view point enabled us to reveal new knowledge about Baldwinian evolution. The findings provide directions to applications, as well as materials for biology and philosophy.

In this thesis, we briefly introduced the background and proposal, presented experiments on three topics about Baldwinian evolution's mechanisms, then discussed our view point and its products. Our results are useful for real world algorithm designs, and may provide ideas for biology and philosophy.

In Chapter 2, we measure computational costs and fitness improvements, and emphasize that what is essential for search efficiency is not improvements during learning, but improvements between generations. The appendix provides results with continuous optimization benchmarks to further support the conclusion. Under this view, we compare children's fitness improvement with that of their parents, and find that children are catching up with the previous generation through their learning, rather than explore beyond their parents. The experimental results imply that, Baldwinian learning's role is to maintain a certain level of learning potential, rather than to further explore on the fitness landscape. We also verified that plasticity encoding, which is popular in Baldwinian algorithms, does not change this basic fact.

This part of experiments determines the role of Baldwinian learning as passing on refinement potential, and Baldwinian evolution as a kind of evolution towards individuals having promising learning potential. Although the fundamental difference between Baldwinian and Lamarckian evolution is just the inheritance of learned traits, the learning's role significantly changes. In Lamarckian evolution, parents' learning is "done". Their children do not have to repeat the parents' learning, or to know how the parents made such achievements, because all what found are already encoded into genotypes and passed on to the children. In Baldwinian evolution, however, the children need to know about their parents' learning, otherwise they often cannot achieve good fitness, their parents' achievements will be lost by the population.

For design issues, the concept of "learning potential" implies that, algorithms settings should protect individuals' capabilities to catch up with their parents. For example, it is not wise to assign the individuals frequently varying budget limits, or taking only a random set of individuals to perform learning. When an individual has too low learning intensity comparing to its parents, it cannot realize the parents' achievements, and the parents' learning efforts become a waste. On the other hand, it is helpful to put learning controlling information in genotypes, such as plasticity codes. Such genes could provide a brief guidance to individuals and tell them how their parents managed to reach high fitness. These directions are not quantitative, but can avoid some inefficiency from the beginning of algorithm designs, by selecting suitable components of the algorithm, such as the genotype form and learning intensity control.

For biology/philosophy, this may imply that acquired learning and its products are to guarantee the groups' robustness to the environment, rather than changing the group significantly.

With acquired learning, organisms may fit the changes in the environment, even some sudden and significant changes. This is much more reliable than random gene mutations, since mutations are rare and blind. On the other hand, children learn from their parents, make advantage of their parents' surviving experience and find their own ways of living. This is what we can see in the nature, and many animals even have apparent behaviors of parents training their children. With more learning, children are more difficult to improve much beyond their parents, however, the organisms attain more talents to live in the world. Although our computational models are too simple to describe the real creatures, such inferences may be inspirations.

The study verifying learning potential brings some clues, however, the question is just a beginning. Learning potential makes sense only when it is realized by the children, however, this realization is not always guaranteed. This is the topic of Chapter 3, and the appendix provides some further data supports. By comparing locally same learning steps of children and their parents, we analyze whether children follow or not parents' learning, and find that learning schemes and genetic operators influence in different ways. A learning scheme's randomness causes non-inheritable variations, prohibits following, decelerates search, breaks inherited potential and results in low fitness. A crossover's breaking effect causes inheritable variations, prohibits following but improves diversity, thus there is a trade-off between search speed and final fitness. These two aspects can be explained by a single principle, that only the factors being inherited or repeated in children's learning can guide evolution.

This conclusion is an extension of Chapter 2's conclusion, and place the question on a general stage. What does the inheritance of "learning potential" consist of? This study answers that, the potential passing through generations is in fact a collection of characters that included in genes or repeated through determined learning. Of course it is different from Lamarckian evolution which simply passes on all learned traits. In Baldwinian evolution, only a part of learned characters or learning behaviors, which is strongly dependent on components of the algorithm, are passed on as the inheritance, and this inheritance asks for computational efforts. How learning guides evolution depends on how much the children can repeat, through the indirect path. This implies that the key problem of Baldwinian evolution's search efficiency is how to make it "like" Lamarckian evolution. Of course this has to trade off with Baldwinian evolution's robustness, especially when applied in dynamic environments.

For design issues, "let children follow their parents' learning" can be a criterion. When designing the learning scheme in a Baldwinian algorithm, we should consider not only how effective can the learning scheme improve an individual, but also how good can a child follow its parents with the same learning scheme. A learning scheme with too little randomness may prevent the search from finding better solutions, but a highly random learning scheme cannot maintain the learned good solutions in later generations. When using Baldwinian algorithms, a balance is necessary, and may often appear as a limitation of learning randomness.

For biology/philosophy aspect, the findings about following parents' learning again emphasizes the need of "education" between generations. Only when parents' beneficial learning results are repeated by their children, the inheritance becomes received, and the group can maintain its finding and evolve on. Although such education may cost much because there is too much to learn, it is still the fastest way to improve.

Another essential component in Baldwinian algorithms is learning cost penalties. We include it and form a complete scenario of Baldwinian evolution in Chapter 4, By analyzing three types of selection winner individuals, we revealed how learning cost penalties contributes to and inhibits Baldwinian searches. When the selection scheme provides adequate competition opportunities for individuals that learned the same traits, a quantitatively trivial cost penalty can cause genetic assimilation, without inhibiting learning in the search phase. Otherwise, genetic assimilation's occurrence asks for a high cost penalty that transforms the fitness landscape significantly, and makes individuals with different learning costs distinct in fitness. This high cost penalty also brings a side-effect of inhibiting learning, which may prevent the algorithm from discovering even better solutions.

Learning is not free, especially in Baldwinian algorithms. In an individual's lifetime, it take one evaluation for evolution/initial solution, and multiple evaluation calls for the learning process. The learning cost becomes extremely high in Baldwinian algorithms, in which all the individuals have to take learning to catch up with their parents and maintain the search findings. This calls for a limitation on learning intensity, and the direct solution is a cost penalty in the fitness function. However, the effect of such a penalty depends on the selection scheme and may degrade the search performance. Ultimately, what we need in Baldwinian evolution is the selective pressure on learning cost, not the punishment.

For algorithm designs, realizing a cost penalty is only one possible way to limit learning cost also can help in search efficiency of Baldwinian algorithms. To enable genetic assimilation, and to control computational costs, it is more important to design an appropriate selection scheme, rather than to tune the scale of learning cost punishment. When having many opportunities for individuals converging to the same peak to compete with each other, learning intensity can be reduced, and learning steps that are not improving can be avoided. Of course, what we have verified is still a limited conclusion on discrete problems, in which similar individuals are likely to learn exact the same results. In continuous problems, classifying "similar" or not is a very complex preliminary problem.

In the natural world, of course, learning has costs, and in many types, as Turney once listed [52]. The costs are more complex, and, in fact, the proportion of evolution's (especially reproduction) and learning's costs is much different from that in algorithms. Furthermore, learning costs and selection are not separate in the natural world. The computational models can be approximations of Baldwinian evolution, however, from the aspect of learning costs, such approximations are far from accurate. We need to modify the models to describe organisms, or on the other hand, re-design learning-selection parts according to biological facts may improve Baldwinian algorithms.

The results in this thesis imply the necessity of analyzing Baldwinian evolution from the aspect of individual behaviors. An evolutionary/memetic algorithm is based on a group of candidate solutions. We usually focus on the average performance or best individual of the group, however, the diversity of the group is also essential. Diversity enables the group to improve through generations, and individuals' behaviors make the behavior of the group. Taking macroscopic views, we have the "smoothing" metaphor, the two-phase description, works on development processes, etc. These findings solve many problems. But only when we focus on the similarities and differences of the group members, we can solve some other problems, such as how children follow their parents, or whether we can assign learning intensities arbitrarily.

This thesis contributes to the understanding of Baldwinian evolution, as well as to algorithm design issues. We collected some design rules, mainly qualitative, since appropriate algorithms forms are prior to appropriate parameters. Knowing these rules does not guarantee efficient Baldwinian algorithms, but they can be directions to avoid some inefficiencies. Such findings can reduce blind trials and confusions, and provide a foundation for further studies.

Considering the whole thesis as an auxiliary material for biologists, the contribution is difficult to determine. In biology, pure Lamarckism is incorrect, but Baldwinism is also not verified. Furthermore, recent biological studies imply some local Lamarckian phenomena. It seems that in the natural world, the true combination of evolution and learning is neither Lamarckian nor Baldwinian, but between the two. Lamarckian algorithm studies may help in understanding the Lamarckian parts, and Baldwinian studies, such as this thesis, may help with the Baldwinian parts. On the other hand, further discoveries in biology may also provide new inspirations for evolutionary/memetic computation.

# Appendix A

# Continuous Optimization Experiments

In the main body of this thesis, Baldwinian evolution's mechanisms are studied with the NK model. We have three fitness landscapes with different epistasis and complexity levels, however, the experiments are not covering all situations. In order to accumulate more evidences for our conclusions, and to examine how the phenomena can change in continuous optimization, we conducted some additional experiments and describe them here in this appendix. Some phenomena occur in different forms comparing to using the NK model, however, all the results support our conclusions presented in the thesis.

TableA.1: Functions in IEEE CEC 2005 Test Suite

| No. | Function |
|---|---|
| $F_1$ | Shifted Sphere Function |
| $F_2$ | Shifted Schwefel ' s Problem 1.2 |
| $F_3$ | Shifted Rotated High Conditioned Elliptic Function |
| $F_4$ | Shifted Schwefel ' s Problem 1.2 with Noise in Fitness |
| $F_5$ | Schwefel ' s Problem 2.6 with Global Optimum on Bounds |
| $F_6$ | Shifted Rosenbrock ' s Function |
| $F_7$ | Shifted Rotated Griewank ' s Function without Bounds |
| $F_8$ | Shifted Rotated Ackley ' s Function with Global Optimum on Bounds |
| $F_9$ | Shifted Rastrigin ' s Function |
| $F_{10}$ | Shifted Rotated Rastrigin ' s Function |
| $F_{11}$ | Shifted Rotated Weierstrass Function |
| $F_{12}$ | Schwefel ' s Problem 2.13 |
| $F_{13}$ | Expanded Extended Griewank ' s plus Rosenbrock ' s Function (F8F2) |
| $F_{14}$ | Shifted Rotated Expanded Scaffer ' s F6 |
| $F_{15}$ | Hybrid Composition Function |
| $F_{16}$ | Rotated Hybrid Composition Function |
| $F_{17}$ | Rotated Hybrid Composition Function with Noise in Fitness |
| $F_{18}$ | Rotated Hybrid Composition Function |
| $F_{19}$ | Rotated Hybrid Composition Function with a Narrow Basin for the Global Optimum |
| $F_{20}$ | Rotated Hybrid Composition Function with the Global Optimum on the Bounds |
| $F_{21}$ | Rotated Hybrid Composition Function |
| $F_{22}$ | Rotated Hybrid Composition Function with High Condition Number Matrix |
| $F_{23}$ | Non-Continuous Rotated Hybrid Composition Function |
| $F_{24}$ | Rotated Hybrid Composition Function |
| $F_{25}$ | Rotated Hybrid Composition Function without Bounds |

In this appendix, the benchmark is the IEEE CEC 2005 test suite [50], which consists of 25 continuous function minimization problems. This standard test suite covers various types of real-parameter optimization problems, and is broadly used in continuous optimization studies. The functions are listed in Table A.1, and in the technical report [50] much further details are described. In our experiments, we use all the 25 functions with 30 dimensions.

Our experiments employ methods from a recent study on memetic algorithms with local search chains [36], which presents a Lamarckian algorithm which combines evolutionary search with iterative local search trials. In their work, based on a real-coded steady state genetic algorithm [19], they apply the BLX-$\alpha$ crossover operator with $\alpha = 0.5$ and the negative assortative mating strategy [14], and the BGA mutation operator [38]. They then apply CMA-ES (covariance matrix adaptation evolution strategy) [16, 17] as the local search method.

In our experiments, we change the algorithm to Baldwinian and reset the population size to 100, while keeping the BLX-$\alpha$ crossover, the BGA mutation, and the CMA-ES local search scheme. However, to enable the comparison between children and their parents, we use a generation-based genetic algorithm with 2-tournament selection instead of the steady state genetic algorithm, since the latter varies only a single individual at a time. In addition, we apply a PBLX crossover operator, which is a variation of the BLX crossover [36], and a Solis-Wets local search scheme [49], for comparisons of different crossovers/learning schemes.

In this appendix, we first examine the catching-up phenomena in Baldwinian evolution, corresponding to the contents of Chapter 2, then introduce experiments examining the influence when changing the learning scheme and crossover operator, corresponding to the contents of Chapter 3. The measurement of learning similarity or genetic assimilation in continuous optimization is highly complicated, and calls for niche and classification techniques. Therefore, we do not examine the influence of cost penalties (Chapter 4) in this appendix. In all the sections, results presented are averages of 25 random runs.

## Learning Potential

First of all, we examine fitness improvements comparing to the previous generation and comparing to before learning. All individuals are assigned the same learning intensity, and the learning improvements are shown in Fig.A.1, A.2 and A.3. In the figures, $x$ axis shows the generation number, and $y$ axis shows the fitness improvements. The squares show improvements between generations, noted as $f(b) - f_{pre}$ according to Chapter 3, whileas the triangles show improvements during the current generation's learning, noted as $f(b) - f(0)$. It can be observed that, in a few generations at the beginning, improvements between generations may be greater than improvements in the current generation. But after about the 5th generation, improvements in the current generation rise over, and improvements between generations fall down to around zero. With different benchmark functions, fitness varies much, but the trends are the same.

The results can be described that learning significantly refines the children from their initial solutions, but does not push the whole population apparently forward. This is coincident to the conclusion with the NK model, that children's learning improvements just enable them to catch up with their parents. Although changes in algorithm, optimization target (maximization/minimization) and other characters of the model make the phenomenon in a different form, the fact "learned traits are not inherited" brings the same outcomes.

In addition, Table A.2 shows the fitness and potential scales of the 12th generation. From the aspect of solving these optimization problems, the Baldwinian algorithm's performance may be

TableA.2: Fitness and potential scales of the 12th generation using BLX and CMA-ES

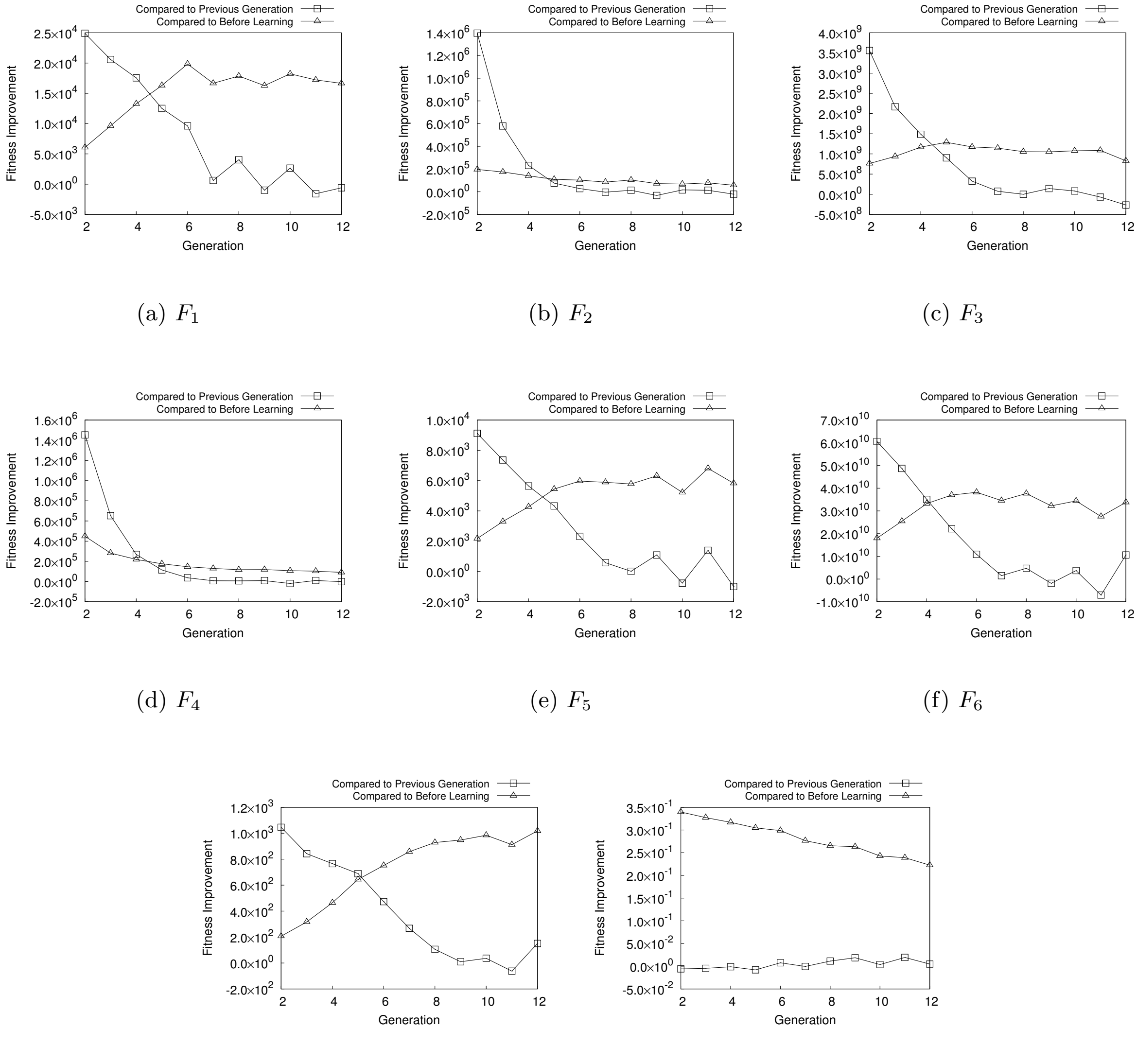
| Function | Average Fitness | Potential Scale | Function | Average Fitness | Potential Scale |
|---|---|---|---|---|---|
| $F_1$ | 9.709393e+04 | 1.665257e+04 | $F_{14}$ | 1.453008e+01 | 2.596800e-01 |
| $F_2$ | 1.946172e+05 | 5.775100e+04 | $F_{15}$ | 1.281630e+03 | 1.010990e+02 |
| $F_3$ | 2.877471e+09 | 8.298220e+08 | $F_{16}$ | 1.205503e+03 | 1.380340e+02 |
| $F_4$ | 1.685714e+05 | 9.062690e+04 | $F_{17}$ | 1.148282e+03 | 2.605620e+02 |
| $F_5$ | 4.627802e+04 | 5.822240e+03 | $F_{18}$ | 1.406825e+03 | 5.533400e+01 |
| $F_6$ | 8.060630e+10 | 3.376360e+10 | $F_{19}$ | 1.414746e+03 | 5.102500e+01 |
| $F_7$ | 1.151713e+04 | 1.018040e+03 | $F_{20}$ | 1.413493e+03 | 5.571300e+01 |
| $F_8$ | 2.131534e+01 | 2.226300e-01 | $F_{21}$ | 1.542668e+03 | 7.542800e+01 |
| $F_9$ | 5.196022e+02 | 4.810150e+01 | $F_{22}$ | 1.861563e+03 | 2.858430e+02 |
| $F_{10}$ | 8.518058e+02 | 1.082650e+02 | $F_{23}$ | 1.542616e+03 | 7.616400e+01 |
| $F_{11}$ | 4.754357e+01 | 4.962420e+00 | $F_{24}$ | 1.555970e+03 | 6.807100e+01 |
| $F_{12}$ | 2.114259e+06 | 5.850660e+05 | $F_{25}$ | 1.940548e+03 | 4.558800e+01 |
| $F_{13}$ | 1.410002e+06 | 9.211570e+05 | | | |

poor. After 12 generations, about 90,000 evaluation calls are taken, the average fitness values are still far above zero and are not decreasing fast. Basically, the tested algorithm here is not designed specially for finding the optimum rapidly, but for investigations of mechanisms. For such a reason, we record the average fitness of the population, instead of best fitness which is more commonly used in continuous optimization research.

## Learning Scheme's Effect

By changing the learning scheme to Solis-Wets' method, we examine the effect of the learning scheme. The learning improvements are shown in Fig.A.4, A.5 and A.6, and Table A.3 lists average fitness and potential scales. With this learning scheme, improvements in the current generation are much higher than improvements between generations from the beginning. With some benchmark functions, the two values are even distinct in the order of magnitude. The fact that children's learning improvements just enable them to catch up with their parents is verified again, and this learning scheme brings much variation in the learning potential scales. The data show that with Solis-Wets local search, individuals have higher potentials passed on through generations, and, the found solutions are better than using CMA-ES. This result is also coincident to our findings with the NK model, that when learning brings better following effect and higher inherited potential, the search also becomes more efficient.

In continuous optimization problems, solutions are in form of real value vectors. As a result, there are no exactly same learning steps in the search, and figuring out whether two individuals



(a) $F_1$

(b) $F_2$

(c) $F_3$

(d) $F_4$

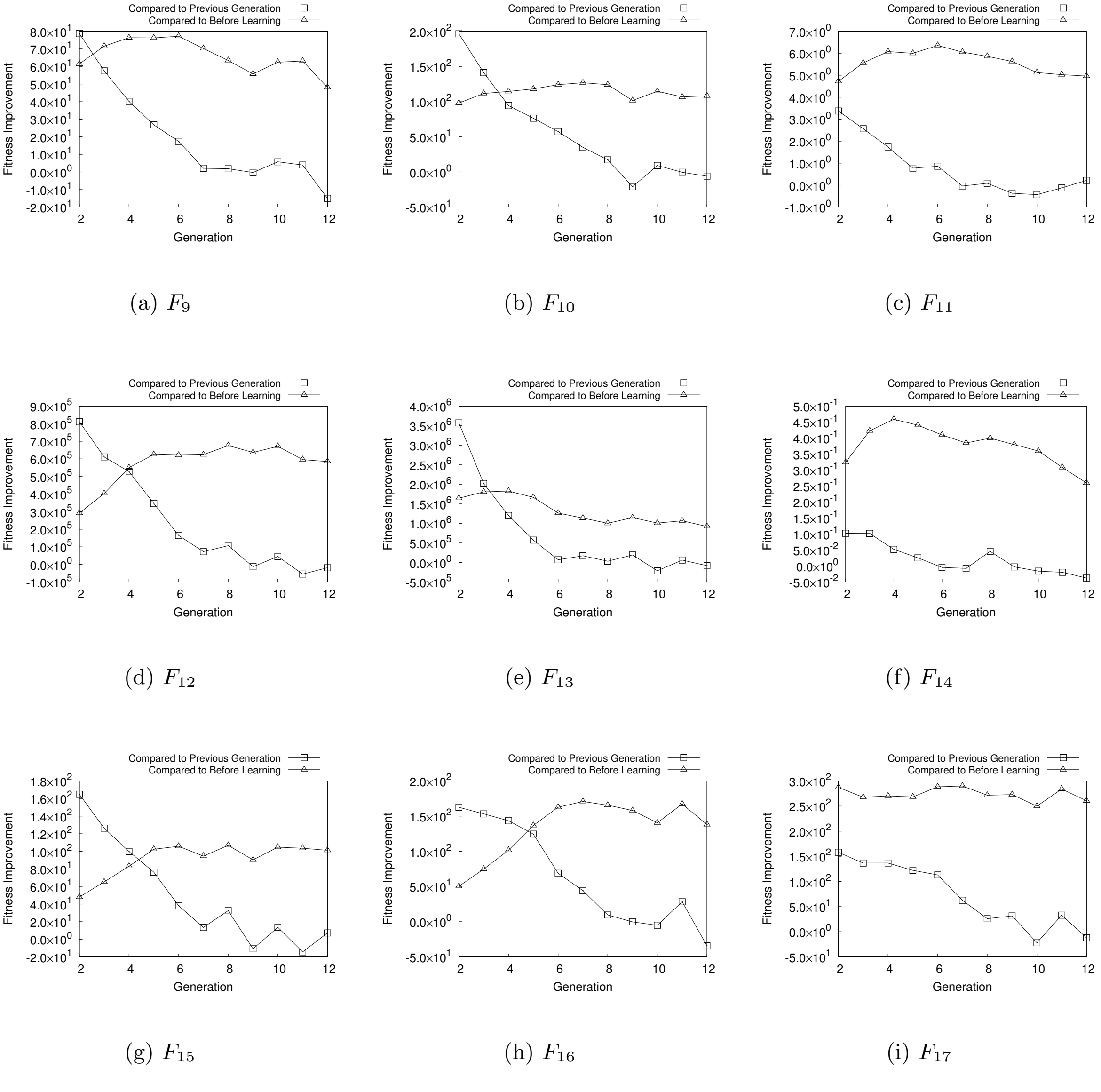
(e) $F_5$

(f) $F_6$

(g) $F_7$

(h) $F_8$

Fig.A.1: Fitness improvements with functions $F_1 \sim F_8$ using BLX and CMA-ES.

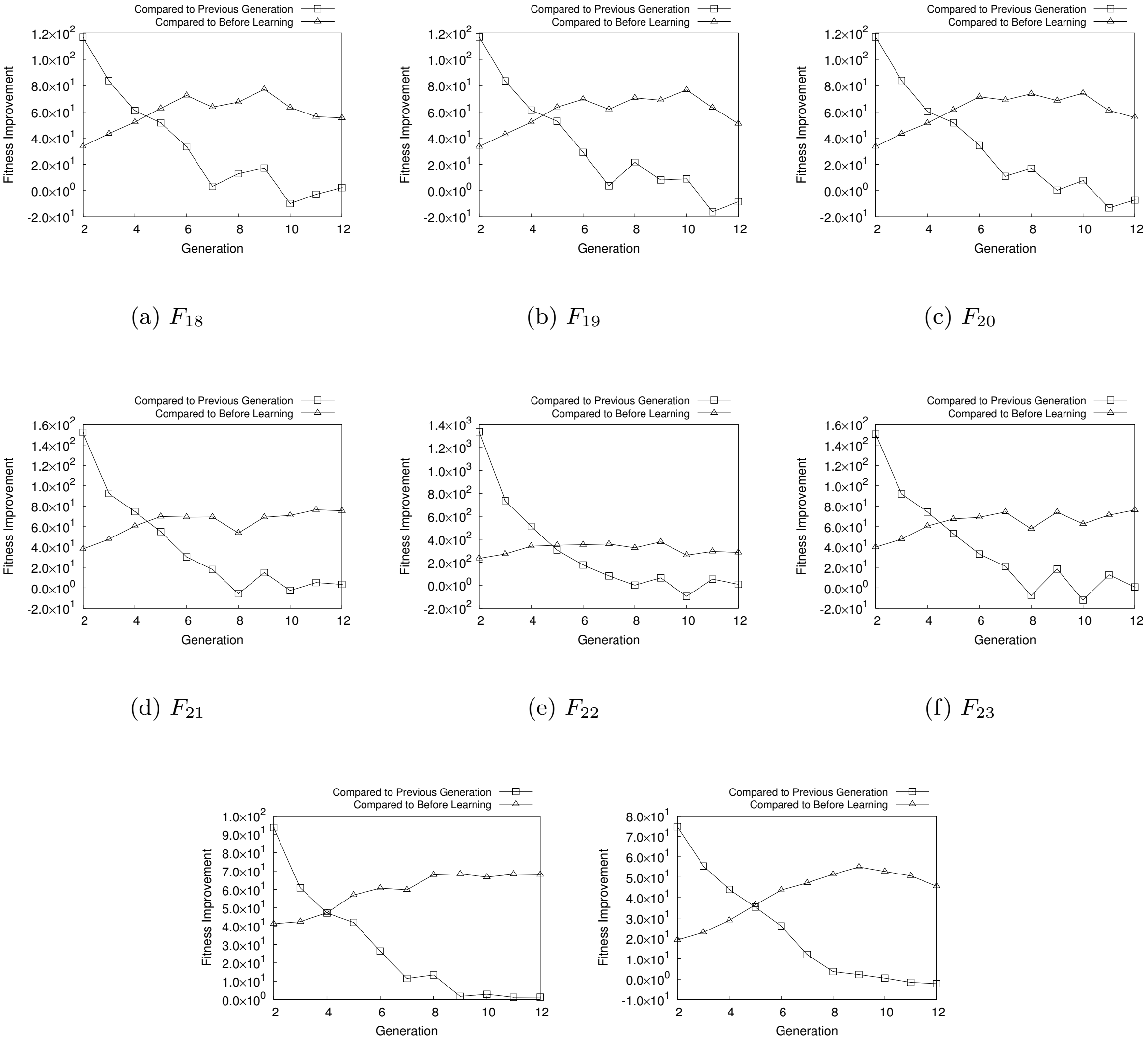


(a) $F_9$ (b) $F_{10}$ (c) $F_{11}$

(d) $F_{12}$ (e) $F_{13}$ (f) $F_{14}$

(g) $F_{15}$ (h) $F_{16}$ (i) $F_{17}$

Fig.A.2: Fitness improvements with functions $F_9 \sim F_{17}$ using BLX and CMA-ES.



(a) $F_{18}$

(b) $F_{19}$

(c) $F_{20}$

(d) $F_{21}$

(e) $F_{22}$

(f) $F_{23}$

(g) $F_{24}$

(h) $F_{25}$

Fig.A.3: Fitness improvements with functions $F_{18} \sim F_{25}$ using BLX and CMA-ES.

TableA.3: Fitness and potential scales of the 12th generation using BLX and Solis-Wets

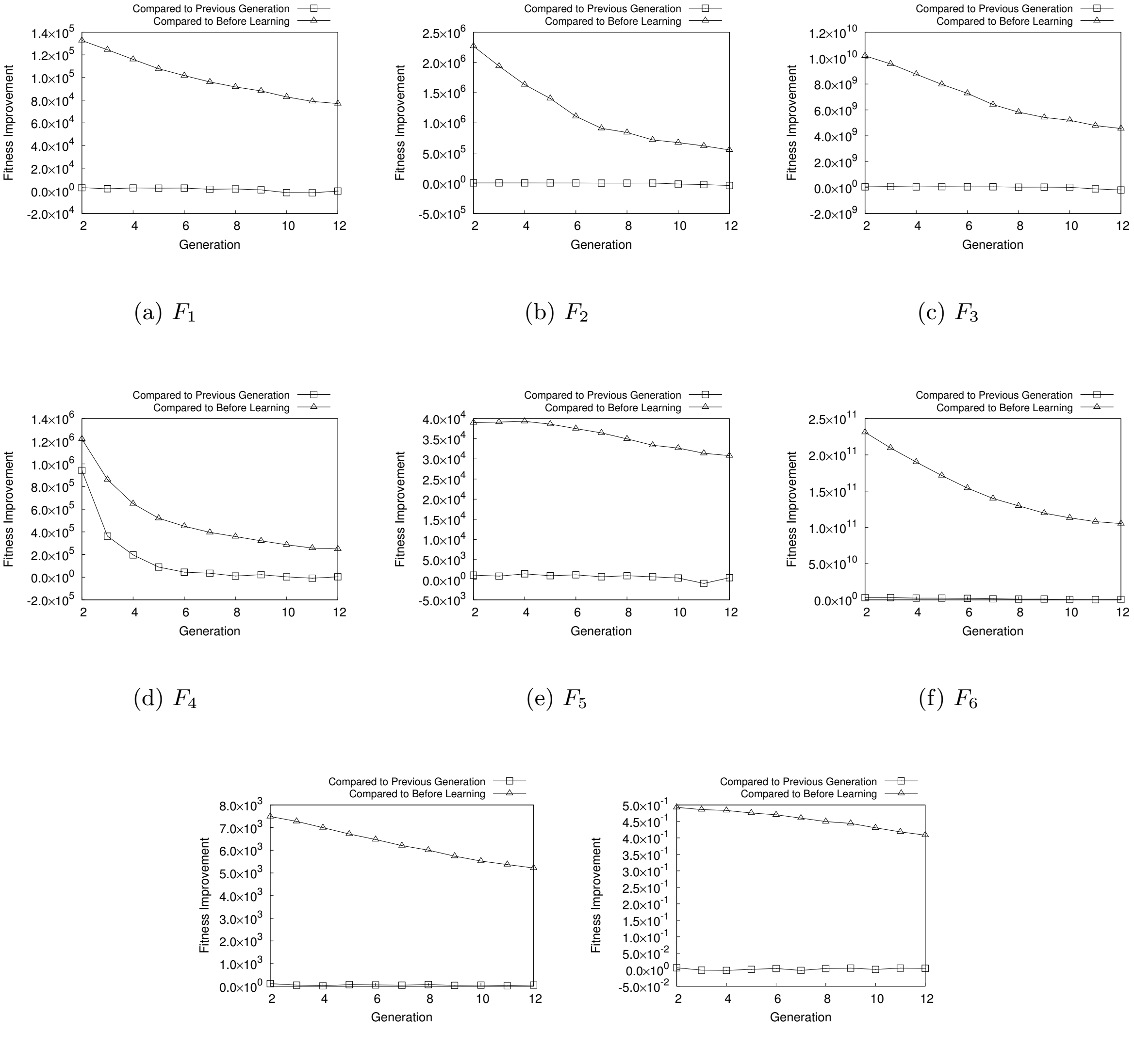
| Function | Average Fitness | Potential Scale | Function | Average Fitness | Potential Scale |
|---|---|---|---|---|---|
| $F_1$ | 3.636323e+04 | 7.704447e+04 | $F_{14}$ | 1.412549e+01 | 6.539300e-01 |
| $F_2$ | 1.383203e+05 | 5.521164e+05 | $F_{15}$ | 7.403768e+02 | 8.916352e+02 |
| $F_3$ | 6.594291e+08 | 4.564474e+09 | $F_{16}$ | 7.153493e+02 | 9.153727e+02 |
| $F_4$ | 1.455673e+05 | 2.494592e+05 | $F_{17}$ | 1.196496e+03 | 2.336700e+02 |
| $F_5$ | 3.143922e+04 | 3.080013e+04 | $F_{18}$ | 1.232921e+03 | 4.565930e+02 |
| $F_6$ | 9.076444e+09 | 1.052143e+11 | $F_{19}$ | 1.216724e+03 | 4.781790e+02 |
| $F_7$ | 7.803948e+03 | 5.220032e+03 | $F_{20}$ | 1.253179e+03 | 4.147060e+02 |
| $F_8$ | 2.119526e+01 | 4.085500e-01 | $F_{21}$ | 1.357268e+03 | 4.508030e+02 |
| $F_9$ | 4.270676e+02 | 1.259049e+02 | $F_{22}$ | 1.435987e+03 | 3.138172e+03 |
| $F_{10}$ | 7.589375e+02 | 1.980897e+02 | $F_{23}$ | 1.473278e+03 | 3.825440e+02 |
| $F_{11}$ | 4.240163e+01 | 9.854610e+00 | $F_{24}$ | 1.503081e+03 | 3.342580e+02 |
| $F_{12}$ | 7.603130e+05 | 2.063375e+06 | $F_{25}$ | 1.898579e+03 | 5.259630e+02 |
| $F_{13}$ | 2.144400e+05 | 2.412709e+06 | | | |

perform similar learning is a demanding task. There is no precise means of "same", and the concept of "nearby" is highly dependent on the landscape characters. According to the continuity and gradient situation, different landscapes have different thresholds for "nearby", and even different regions on the same landscape may have different ways of judging the similarity. Therefore, we skip the examination of learning similarities as an immediate step, and focus on only the input and output of the search.

### Crossover's Effect

We then use PBLX crossover with CMA-ES local search method, to examine the effect of the crossover operator. The learning improvements are shown in Fig.A.7, A.8 and A.9, and Table A.4 lists average fitness and potential scales. Using this crossover operator, the fitness improvements are almost the same as using the BLX crossover operator. Furthermore, the fitness of the 12th generation is also similar to that of using BLX crossover. It is verified again here that the crossover is not likely to change the inherited potential, as it brings inheritable variations to the gene pool.

As mentioned before, in continuous optimization, it is difficult to judge similarities between individuals or their learning processes. Furthermore, it is also difficult to describe the effect of breaking building blocks. Basically the individuals are always changing in reproduction. Effective building blocks can be a region on a dimension, a volume determined by several dimensions in the vector, or an exact starting point. As a result, we do not have a method to



(a) $F_1$

(b) $F_2$

(c) $F_3$

(d) $F_4$

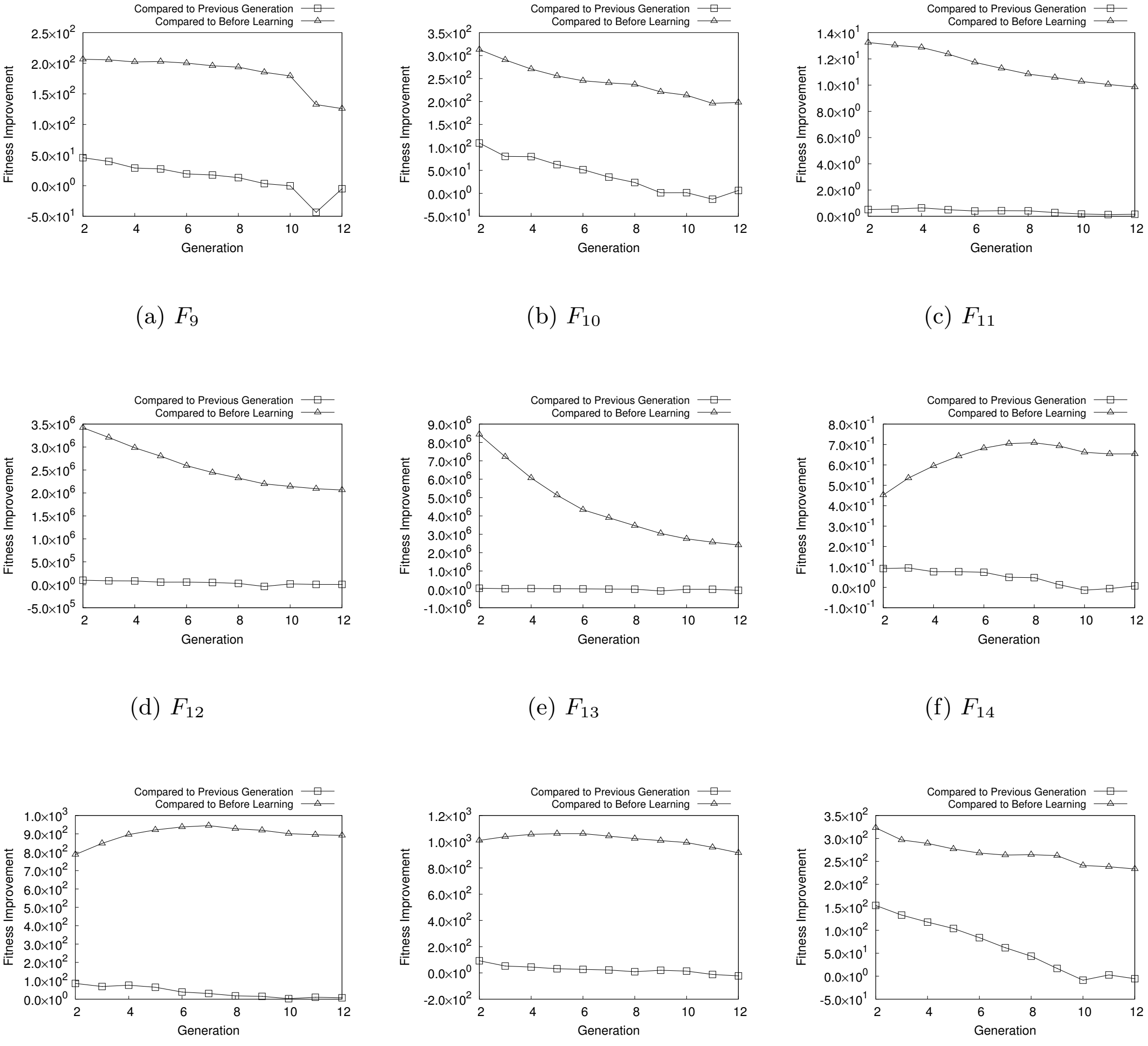
(e) $F_5$

(f) $F_6$

(g) $F_7$

(h) $F_8$

Fig.A.4: Fitness improvements with functions $F_1 \sim F_8$ using BLX and Solis-Wets.



(a) $F_9$

(b) $F_{10}$

(c) $F_{11}$

(d) $F_{12}$

(e) $F_{13}$

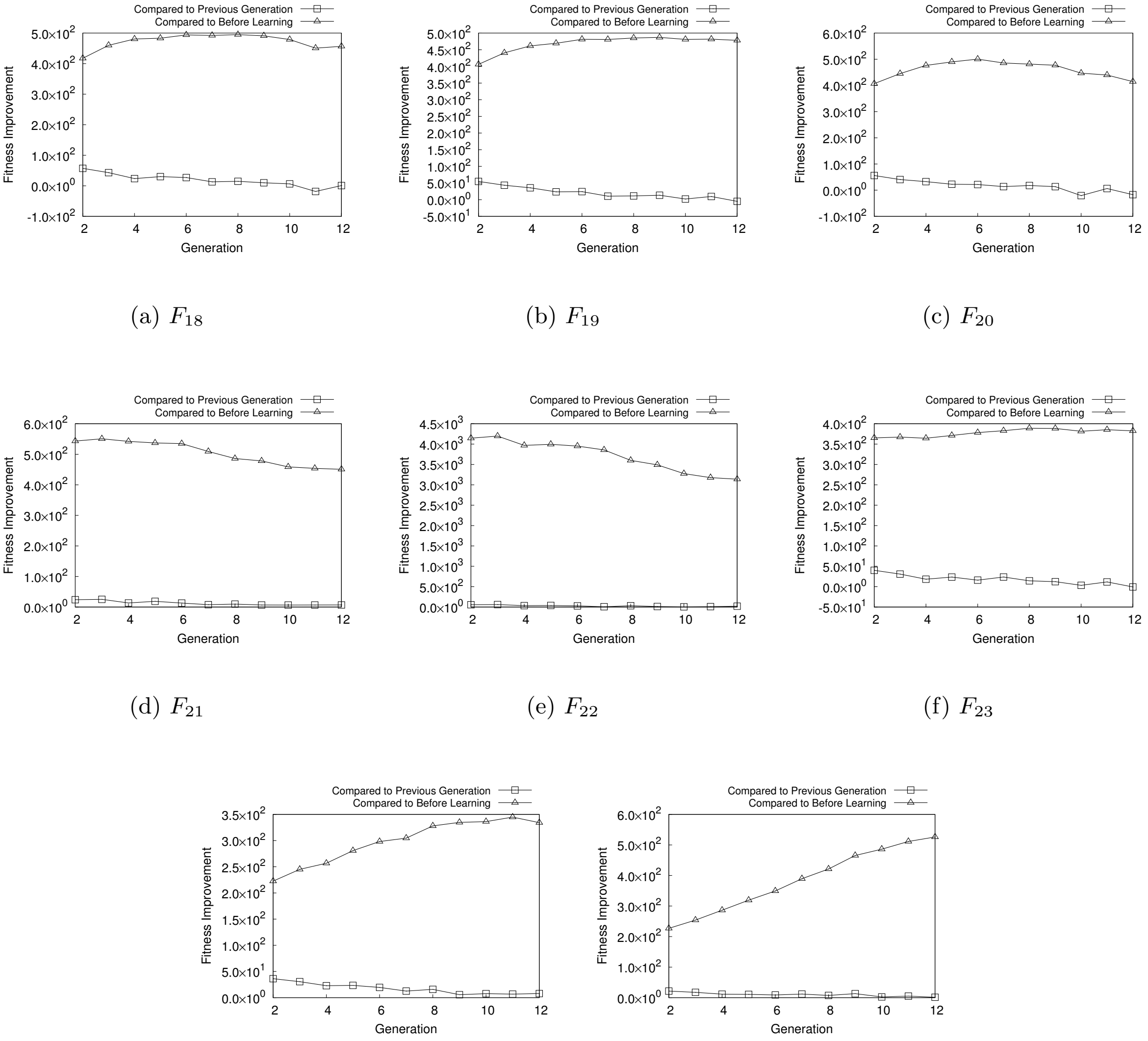
(f) $F_{14}$

(g) $F_{15}$

(h) $F_{16}$

(i) $F_{17}$

Fig.A.5: Fitness improvements with functions $F_9 \sim F_{17}$ using BLX and Solis-Wets.



(a) $F_{18}$

(b) $F_{19}$

(c) $F_{20}$

(d) $F_{21}$

(e) $F_{22}$

(f) $F_{23}$

(g) $F_{24}$

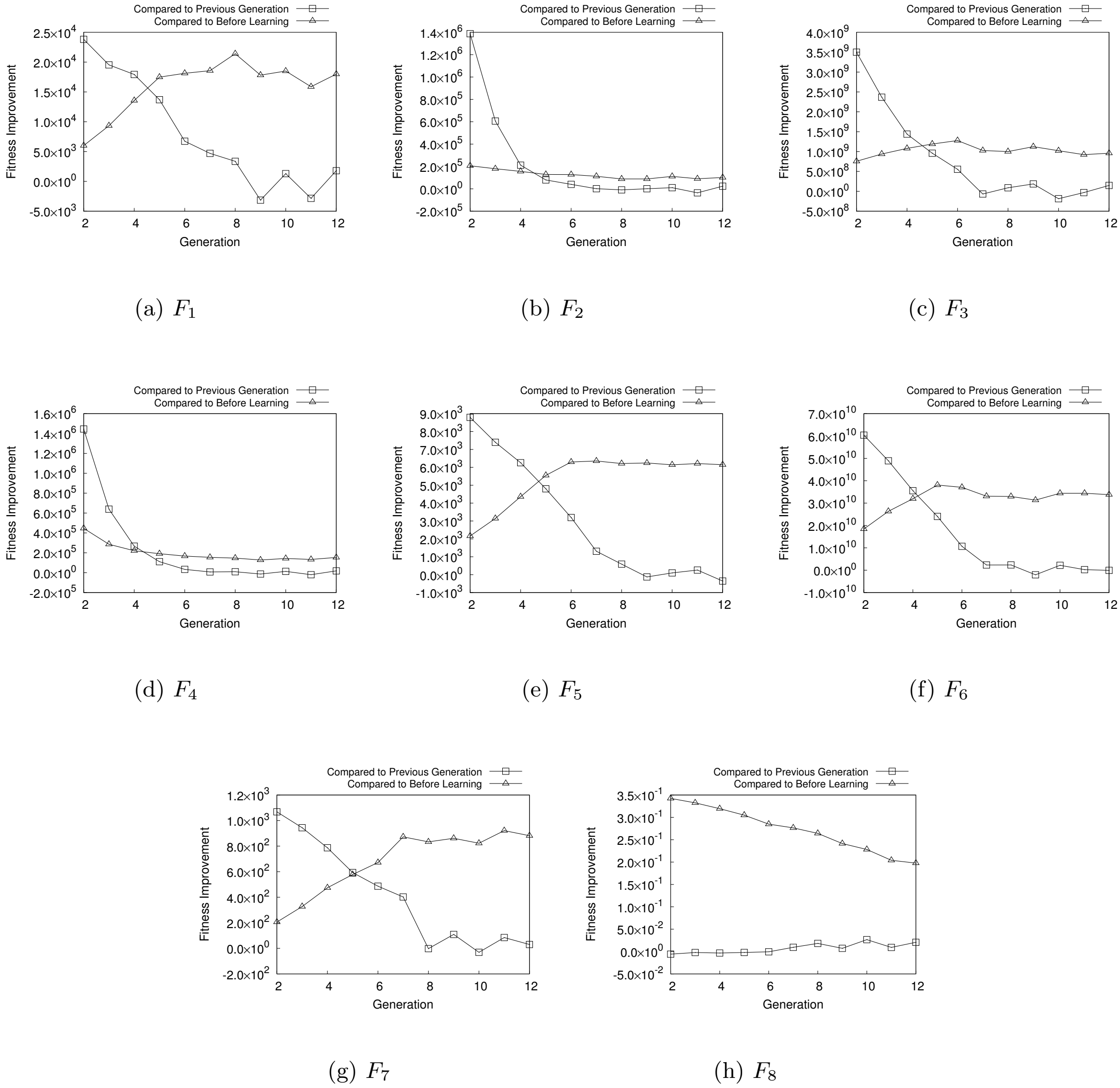
(h) $F_{25}$

Fig.A.6: Fitness improvements with functions $F_{18} \sim F_{25}$ using BLX and Solis-Wets.

TableA.4: Fitness and potential scales of the 12th generation using PBLX and CMA-ES

| Function | Average Fitness | Potential Scale | Function | Average Fitness | Potential Scale |
|---|---|---|---|---|---|
| $F_1$ | 9.935136e+04 | 1.799054e+04 | $F_{14}$ | 1.451056e+01 | 3.347700e-01 |
| $F_2$ | 1.971154e+05 | 1.013557e+05 | $F_{15}$ | 1.226237e+03 | 1.175930e+02 |
| $F_3$ | 2.279776e+09 | 9.587780e+08 | $F_{16}$ | 1.127452e+03 | 1.427300e+02 |
| $F_4$ | 1.910765e+05 | 1.533672e+05 | $F_{17}$ | 1.259544e+03 | 2.791570e+02 |
| $F_5$ | 4.433963e+04 | 6.146320e+03 | $F_{18}$ | 1.398195e+03 | 8.314400e+01 |
| $F_6$ | 8.454966e+10 | 3.378334e+10 | $F_{19}$ | 1.411134e+03 | 6.883700e+01 |
| $F_7$ | 1.137070e+04 | 8.813900e+02 | $F_{20}$ | 1.401532e+03 | 7.041300e+01 |
| $F_8$ | 2.128284e+01 | 1.977700e-01 | $F_{21}$ | 1.528675e+03 | 8.007100e+01 |
| $F_9$ | 5.197329e+02 | 5.060470e+01 | $F_{22}$ | 1.917113e+03 | 2.903630e+02 |
| $F_{10}$ | 8.751866e+02 | 1.047364e+02 | $F_{23}$ | 1.547031e+03 | 6.429800e+01 |
| $F_{11}$ | 4.687180e+01 | 4.416600e+00 | $F_{24}$ | 1.569445e+03 | 5.296800e+01 |
| $F_{12}$ | 2.197234e+06 | 4.861910e+05 | $F_{25}$ | 1.940685e+03 | 5.465100e+01 |
| $F_{13}$ | 1.592331e+06 | 9.603020e+05 |  |  |  |

compare the exact preference levels of breaking building blocks. In this part of experiments, what is verified is that crossover operators are not changing learning potential scales significantly, by applying variations to the gene pool.

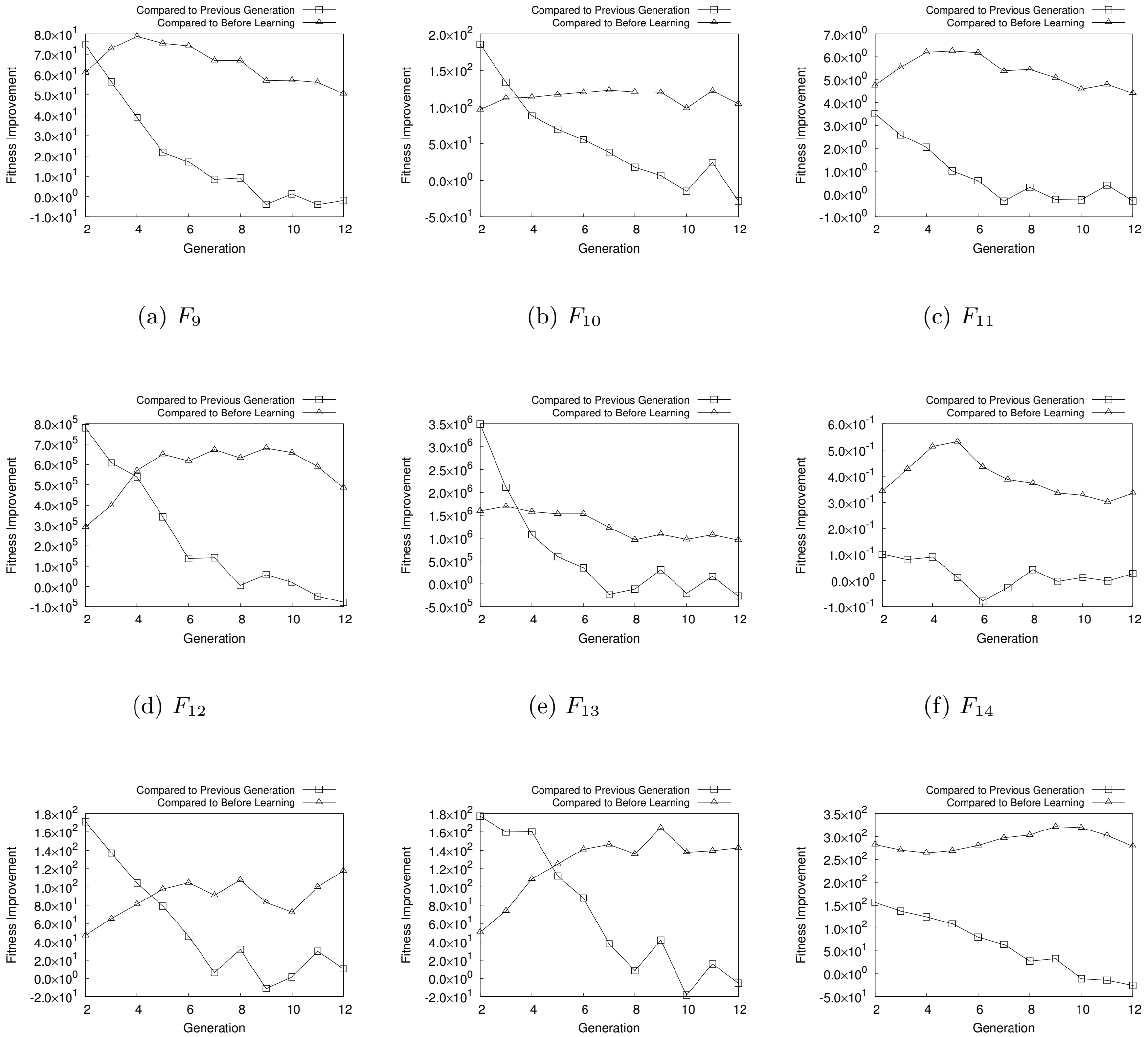


(a) $F_1$ (b) $F_2$ (c) $F_3$

(d) $F_4$ (e) $F_5$ (f) $F_6$

(g) $F_7$ (h) $F_8$

Fig.A.7: Fitness improvements with functions $F_1 \sim F_8$ using PBLX and CMA-ES.

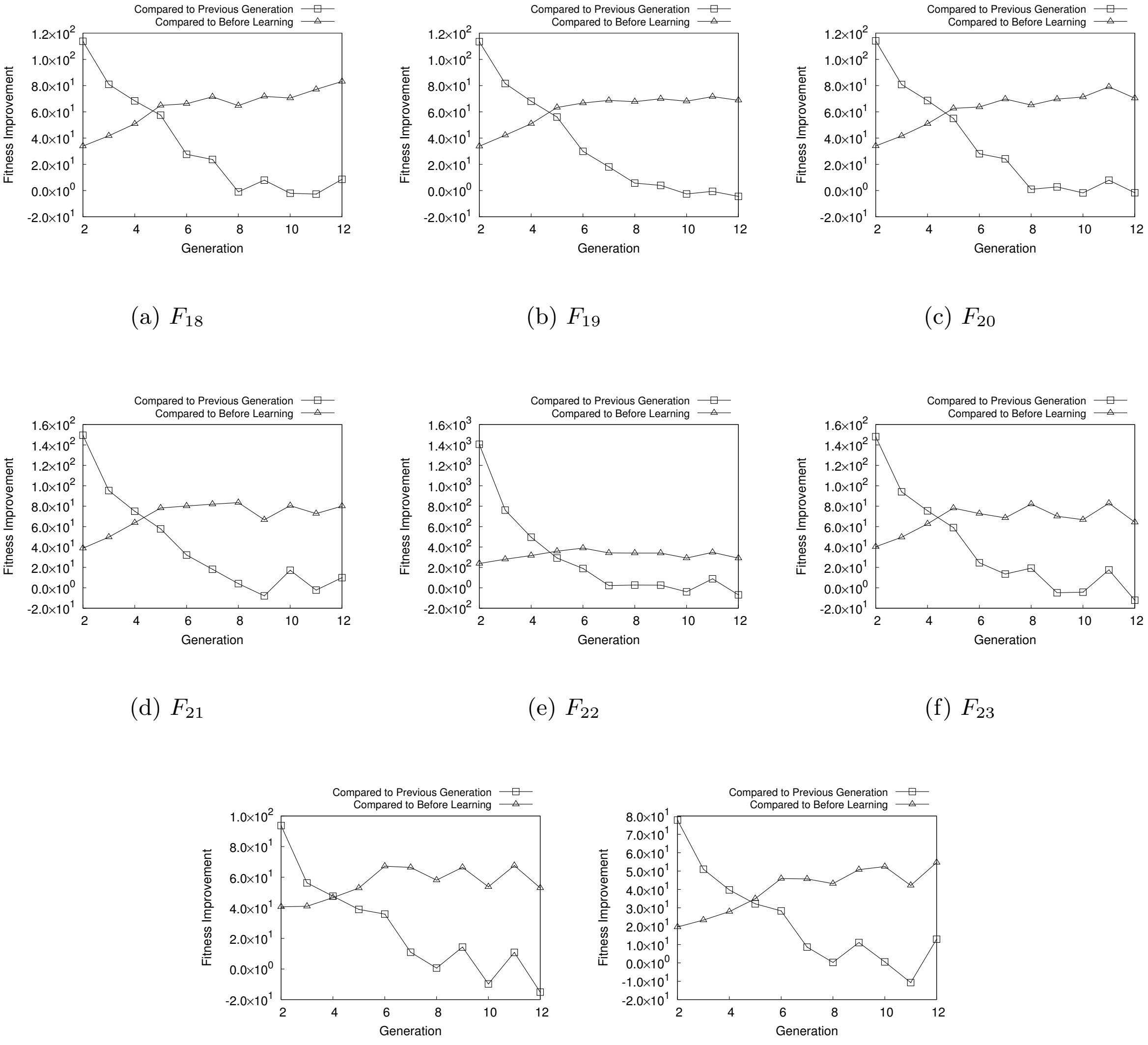


(a) $F_9$ (b) $F_{10}$ (c) $F_{11}$

(d) $F_{12}$ (e) $F_{13}$ (f) $F_{14}$

(g) $F_{15}$ (h) $F_{16}$ (i) $F_{17}$

Fig.A.8: Fitness improvements with functions $F_9 \sim F_{17}$ using PBLX and CMA-ES.



(a) $F_{18}$ (b) $F_{19}$ (c) $F_{20}$

(d) $F_{21}$ (e) $F_{22}$ (f) $F_{23}$

(g) $F_{24}$ (h) $F_{25}$

Fig.A.9: Fitness improvements with functions $F_{18} \sim F_{25}$ using PBLX and CMA-ES.

# Publication List

## Journal Papers

S. Liu, H. Iba, “A Study on Computational Efficiency and Plasticity in Baldwinian Learning,” *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 15(9), pp. 1300—1309, 2011.

S. Liu, H. Iba, “Realization of Learning Potential in Baldwinian Evolution,” *Artificial Life.* (Submitted)

S. Liu, H. Iba, “Learning Intensity and Cost Punishment in Baldwinian Evolution,” *Journal of the Japanese Society for Evolutionary Computation.* (Submitted)

## Conference Papers (with Peer Reviews)

S. Liu, H. Iba, “A Study on the Computational Efficiency of Baldwinian Evolution,” *Proc. of the Second World Congress on Nature and Biologically Inspired Computing (NaBIC 2010)*, pp. 467—472, Kitakyushu, Japan, 2010.

S. Liu, H. Iba, “Imitation Tendencies of Local Search Schemes in Baldwinian Evolution,” *Proc. of Genetic and Evolutionary Computation Conference 2011 (GECCO2011)*, pp. 553—560, Dublin, Ireland, 2011.

B. Tserenchimed, S. Liu, H. Iba, “A Trading Method in FX using Evolutionary Algorithms,” *Proc. of Genetic and Evolutionary Computation Conference 2011 (GECCO2011)*, pp. 139—140, Dublin, Ireland, 2011.

## Conference Papers (without Peer Reviews)

S. Liu, H. Iba, “Baldwin効果を用いた外国為替取引最適化,” 第４回進化計算フロンティア研究会 (SIG-ECF), 2010.

S. Liu, H. Iba, “A Study on Computational Efficiency in Baldwinian Evolution,” 進化計算シンポジウム2010, 2010.

S. Liu, H. Iba, “Learning Imitation Tendencies in Baldwinian Evolution,” 第７回進化計算フロンティア研究会 (SIG-ECF), 2011.

S. Liu, H. Iba, “Genetic Operators May Break Inherited Learning Potential in Baldwinian Evolution,” 進化計算シンポジウム2011, 2011.

# Bibliography

[1] Claus Aranha and Hitoshi Iba. The memetic tree-based genetic algorithm and its application to portfolio. *Memetic Computing*, 1(2):139–151, 2009.

[2] J. Mark Baldwin. A new factor in evolution. *American Naturalist*, 30:441–451, 1896.

[3] N. K. Bambha, S. S. Bhattacharyya, J. Teich, and E. Zitzler. Systematic integration of parameterized local search in evolutionary algorithm. *IEEE Transactions on Evolutionary Computation*, 8(2):137–155, 2004.

[4] Sendhoff Bernhard and Kreutz Martin. A model for the dynamic interaction between evolution and learning. *Neural Processing Letters*, 10(3):181–193, 1999.

[5] C. G. Carrier. Unifying learning with evolution through baldwinian evolution and lamarckism: A case study. In *in Proc. Symp. Computational Intelligence and Learning (CoIL-2000)*, pages 36–41, 2000.

[6] J. H. Chen, D. E. Goldberg, S. Y. Ho, and K. Sastry. Fitness inheritance in multi-objective optimization. In *Genetic and Evolutionary Computation Conference*, pages 319–326, 2002.

[7] Xianshun Chen, Yew-Soon Ong, Meng-Hiot Lim, and Kay Chen Tan. A multi-facet survey on memetic computation. *IEEE Transactions on Evolutionary Computation*, 15(5):591–607, 2011.

[8] S. S. Choi and B. R. Moon. A graph-based lamarckian-baldwinian hybrid for the sorting network problem. *IEEE Transactions on Evolutionary Computation*, 9(1):105–114, 2005.

[9] K. L. Downing. Deisgning neutral networks using genetic algorithms with graph generation system. *Genetic Programming and Evolvable Machines*, 2(3):259–288, 2001.

[10] K. L. Downing. Development and the baldwin effect. *Artificial Life*, 10(1):39–63, 2004.

[11] K. L. Downing. Computational explorations of the baldwin effect. In *Proceedings of the First Norwegian Artificial Intelligence Symposium*, pages 41–52, 2009.

[12] K. L. Downing. The baldwin effect in developing neural networks. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 555–562, 2010.

[13] T. El-Mihoub, Adrian Hopgood, L. Nolle, and A. Battersby. Self-adaptive baldwinian search in hybrid genetic algorithms. In *Computational Intelligence, Theory and Applications*, volume 38, pages 597–602, September 2006.

[14] C. Fernandes and A. Rosa. A study on non-random mating and varying population size in genetic algorithms using a royal road function. In *Proceedings of the 2001 Congress on Evolutionary Computation*, pages 60–66, 2001.

[15] D. E. Goldberg and S. Voessner. Optimizing global-local search hybrids. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 220–228, 1999.

[16] N. Hansen, S. D. Muller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). 11(1):1–18, 2003.

[17] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. 9(2):159–195, 2001.

[18] S. M. K. Hasan, R. Sarker, D. Essam, and D. Cornforth. Memetic algorithms for solving job-shop scheduling problems. *Memetic Computing*, 1(1):69–83, 2009.

[19] F. Herrera, M. Lozano, and J. L. Verdegay. Tackling real-coded genetic algorithms: Operators and tools for the behavioral analysis. 12(4):265–319, 1998.

[20] G.E. Hinton and S.J. Nowlan. How learning can guide evolution. *Complex Systems*, 1:495–502, 1987.

[21] Hisao Ishibuchi, Shiori Kaige, and Kaname Narukawa. Comparison between lamarckian and baldwinian repair on multiobjective 0/1 knapsack problems. In *Proc. of Third International Conference on Evolutionary Multi-Criterion Optimization, EMO 2005, ser. Lecture Notes in Computer Science*, pages 370–385. Springer, 2005.

[22] Y. Jin, M. Olhofer, and B. Sendhoff. A framework for evolutionary optimization with approximate fitness function. *IEEE Transactions on Evolutionary Computation*, 6(5):481–494, 2002.

[23] Y. Jin, M. Olhofer, and B. Sendhoff. Faster convergence by means of fitness estimation. *Soft Computing*, 9(1):13–20, 2005.

[24] C. G. Johnson and J. J. R. Caldalda. The memetic tree-based genetic algorithm and its application to portfolio. *Leonardo*, 35(2):175–184, 2002.

[25] S. Kauffman. *The Origin of Order: Self-Organization and Selection in Evolution.* Oxford University Press, 1993.

[26] H. S. Kim and S. B. Cho. An efficient genetic algorithm with less fitness evaluation by clustering. In *Congress on Evolutionary Computation*, pages 887–894, 2001.

[27] H. Kitano. Deisgning neutral networks using genetic algorithms with graph generation system. *Complex Systems*, 4:461467, 1990.

[28] Morris Kline. *Mathematics: The Loss of Certainty.* Oxford University Press, New York, 1980.

[29] N. Krasnogor and J. Smith. A memetic algorithm with self-adaptive local search: Tsp as a case study. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 987–994, 2000.

[30] N. Krasnogor and J. Smith. A tutorial for competent memetic algorithms: Model, taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation*, 9(5):474–488, 2005.

[31] K. W. C. Ku, M. W. Mak, and W. C. Siu. A study of the lamarckian evolution of recurrent neural networks. *IEEE Transactions on Evolutionary Computation*, 4(1):31–42, 2000.

[32] E. W. Lameijer, T. Baeck, J. N. Kok, and A. P. Ijzerman. Evolutionary algorithms in drug design. *Natrual Computing*, 4(3):177–243, 2005.

[33] S. Liu and H. Iba. A study on computational efficiency and plasticity in baldwinian learning. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 15(9):1300–1309, 2011.

[34] Giles Mayley. Landscapes, learning costs and genetic assimilation. *Evolutionary Computation*, 4:213–234, 1996.

[35] G.F. Miller, P.M. Todd, and S.U. Hedge. Deisgning neutral networks using genetic algorithms. In Morgan Kaufmann, editor, *Proceedings of Third Int. Conf. Genetic Algorithms*, page 379384, San Francisco, 1989.

[36] D. Molina, M. Lozano, C. Garc 鱈 a-Mart 鱈 nez, and F. Herrera. Memetic algorithms for continuous optimization based on local search chains. *Evolutionary Computation*, 8(1):27–63, 2010.

[37] C. Lloyd Morgan. On modification and variation. *Science*, 4:733–740, 1896.

[38] H. Muhlenbein and D. Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm in continuous parameter optimization. 1:25–49, 1993.

[39] Q.H. Nguyen, Y.S. Ong, and N. Krasnogor. A study on the design issues of memetic algorithm. In *Proc. of 2007 IEEE Congress on Evolutionary Computation*, pages 2390–2397, 2007.

[40] Quang Huy Nguyen, Yew-Soon Ong, and Meng Hiot Lim. A probabilistic memetic framework. *IEEE Transactions on Evolutionary Computation*, 13(3):604–623, 2009.

[41] N. Noman and H. Iba. Accelerating differential evolution using an adaptive local search. *IEEE Transactions on Evolutionary Computation*, 12(1):107–125, 2008.

[42] Yew-Soon Ong, Meng-Hiot Lim, Ning Zhu, and Kok-Wai Wong. Classification of adaptive memetic algorithms: a comparative study. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 36(1):141–152, 2006.

[43] H.F. Osborn. Ontogenic and phylogenic variation. *Science*, 4:786–789, 1896.

[44] Ingo Paenke, Tadeusz J. Kawecki, and Bernhard Sendhoff. On the influence of lifetime learning on selection pressure. In *Proceedings of Artificial Life X*, pages 500–506, 2006.

[45] Ingo Paenke, Tadeusz J. Kawecki, and Bernhard Sendhoff. The influence of learning on evolution: A mathematical framework. *Artificial Life*, 15(2):227–245, 2009.

[46] Takahiro Sasaki and Mario Tokoro. Evolving learnable neural networks under changing environments with various rates of inheritance of acquired characters: Comparison between darwinian and lamarckian evolution. *Artificial Life*, 5(3):203–223, 1999.

[47] D. Simon. Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation*, 12(6):702–713, 2008.

[48] R. Smith, B. Dike, and S. Stegmann. Fitness inheritance in genetic algorithms. In *ACM Symposiums on Applied Computing*, pages 345–350, 1995.

[49] F. J. Solis and R. J.-B. Wets. Minimization by random search techniques. 6:19–30, 1981.

[50] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. *Technical report*, 2005.

[51] R. Suzuki and T. Arita. The dynamic changes in roles of learning through the baldwin effect. *Artificial Life*, 13(1):31–43, 2007.

[52] P. Turney. Myths and legends of the baldwin effect. In *Proceedings of the Workshop on Evolutionary Computing and Machine Learning at the 13th International Conference on Machine Learning (ICML-96)*, pages 135–142, 1996.

[53] C.H. Waddington. Canalization of development and the inheritance of acquired characters. *Nature*, 150:563–565, 1942.

[54] J. Watson and J. Wiles. The rise and fall of learning: A neural network model of the genetic assimilation of acquired traits. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002)*, pages 600–605, 2002.

[55] D. L. Whitley, V. S. Gordon, and K. E. Mathias. Lamarckian evolution, the baldwin effect and function optimization. In *Proceedings of International Conference on Parallel Problem Solving From Nature III (PPSN III)*, pages 6–15, 1994.

[56] Z. Zhou, Y. S. Ong, P. B. Nair, A. J. Keane, and K. Y. Lum. Combining global and local surrogate models to accelerate evolutionary optimization. *Systems, Man and Cybernetics, Part C: Reviews and Applications, IEEE Transactions on*, 37(1):66–76, 2007.