Free Lunch on the Discrete Lipschitz Class

Pei Jiang

NCLab Report No. NCL-TR-2009005
July 2009

Natural Computing Laboratory (NCLab)
Department of Computer Science
National Chiao Tung University
329 Engineering Building C
1001 Ta Hsueh Road
HsinChu City 300, TAIWAN
http://nclab.tw/

國立交通大學

資訊科學與工程研究所

碩 士 論 文

NFL 定理在離散化 Lipschitz 函數集合上之探討

Free Lunch on the Discrete Lipschitz Class

研 究 生：江沛

指導教授：陳穎平　教授

中 華 民 國　九 十 八 年 六 月

NFL 定理在離散化 Lipschitz 函數集合上之探討
Free Lunch on the Discrete Lipschitz Class

研 究 生：江沛　　Student：Pei Jiang

指導教授：陳穎平　　Advisor：Ying-ping Chen

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2008

Hsinchu, Taiwan, Republic of China

中華民國九十八年六月

NFL 定理在離散化 Lipschitz 函數集合上之探討

學生：江沛　　指導教授：陳穎平

國立交通大學資訊科學與工程研究所

摘　　要

No-Free-Lunch(NFL)定理指出當對所有問題作平均時，所有最佳化演算法的表現都是一致的，意即各個最佳化演算法的總體效能並無法定義孰優孰劣。然而 NFL 定理並不意味著泛用型最佳化演算法(general-purpose optimizers)無用武之地，只要問題存在有可供演算法利用的結構，仍有可能找到一最佳化演算法在某一問題集合上具有優勢。這份論文提出了一個問題的集合，稱之為離散化 Lipschitz 函數集合(discrete Lipschitz class，DLC)，且此一集合可視為透過規範搜尋空間鄰近區域的差值來模擬連續性。此論文探討了 DLC 和 NFL 定理間之關係，並證明一最佳化演算法 subthreshold-seeker 之推廣形式可在 DLC 上效能勝於 random search。同時，此論文也設計了一抽樣測試法透過實驗驗證在一更實際之架構下 subthreshold-seeker 在 DLC 上之表現明顯優於 random search。因此，這份論文說明了儘管最佳化演算法並無法同時對所有問題都有優異的效能，但仍然有可能在一廣泛且深具意義的問題集合上取得優勢。

關鍵字：No-Free-Lunch 定理、Lipschitz 連續性、泛用型最佳化演算法、subthreshold-seeker、抽樣測試法

Abstract

The No-Free-Lunch theorem states that all algorithms have the identical performance on average over all functions and there is no algorithm able to outperform others on all problems. However, such a result does not imply that search heuristics or optimization algorithms are futile if we are more cautious with the applicability of these methods and the search space. In this paper, within the No-Free-Lunch framework, we firstly introduce the discrete Lipschitz class by transferring the Lipschitz functions, i.e., functions with bounded slope, as a measure to fulfill the notion of continuity in discrete functions. We then investigate the properties of the discrete Lipschitz class, generalize an algorithm called subthreshold-seeker for optimization, and show that the generalized subthreshold-seeker outperforms random search on this class. Finally, we propose a tractable sampling-test scheme to empirically demonstrate the superiority of the generalized subthreshold-seeker under practical configurations. This study concludes that there exists algorithms outperforming random search on the discrete Lipschitz class in both theoretical and practical aspects and indicates that the effectiveness of search heuristics may not be universal but still general in some broad sense.

keywords:

No-Free-Lunch Theorem, Lipschitz continuity, discrete Lipschitz class, subthreshold-seeker, sampling-test scheme

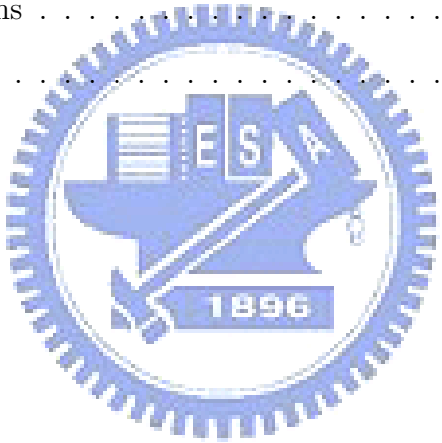
誌　　謝

現在時間是 2009 年七月七日下午三點十八分，我在自然計算實驗室待了兩年的座位看著螢幕上一篇即將完結的論文，和即將完結的學生歲月——從今以後再也不能買學生票了，想來還真有點感傷。因此，以下將一一條列那些致使我接下來的人生得多付幾十塊錢才能看場電影的重大功臣，聊表個人微不足道的巨大謝意：

1. 感謝父母的生育教養，那是一切的根源。
2. 感謝陳穎平老師的開明和適切指導，讓我能自在探索興趣，一窺學術世界的堂奧。
3. 感謝口試委員：張時中老師、鄭士康老師、于天立老師的洞見和寬容，讓我有機會精進這篇論文。
4. 感謝計算機中心校務資訊組提供我學習的機會，還有優渥的工讀金，讓我不用為房租而煩惱。
5. 感謝三位室友一同度過兩年來的點點滴滴。
6. 感謝實驗室學長姐、同學、學弟妹的同舟共濟。
7. 感謝諸多親友的支持和陪伴。
8. 最後，我還要感謝亞利桑納響尾蛇在這兩年無緣季後賽，讓我能心無旁騖的進行研究。

要感謝的人太多了，我很想繼續條列下去，但說明你們對我的幫助足以寫成另一本論文。

Contents



List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Back to 1980s, in the field of evolutionary computation, there is a belief that while evolutionary algorithms may not perform as well as the specialized algorithm for a specific optimization problem, they are more widely applicable and have superior overall performance. However, in 1995, Wolpert and Macready proposed the No-Free-Lunch (NFL) theorem [1, 2] which formally states that every algorithm performs equally well on average over all functions. A direct implication of NFL is that, given any performance measure, the better performance of an algorithm on some problems always accompanies with the worse performance on others. The number of problems on which the algorithm performs well is exactly the number of those on which it does not perform well. In other words, there is no such thing as robustness under the NFL framework, or all algorithms are considered robust. Therefore, it is no surprise that the proposition of the NFL theorem causes a great deal of controversy in the optimization and heuristic search community [3], as the NFL theorem sets a limitation on the pursuit of general-purpose optimizers.

Indeed, the implications of the NFL theorem seem to disagree with empirical observations of the effectiveness of optimization algorithms and search heuristics, since general-purpose optimizers such as gradient-based methods, simulated-annealing, and biologically inspired algorithms do have their share of significance in real-world applications. On the other hand, the NFL theorem is a mathematical theorem, which means that it is absolutely true when all the hypotheses are given. As a consequence, previous studies intending to address the incoherence between theoretical results and empirical observations are mostly

aiming at the hypotheses of the NFL theorem, especially the notion of "all functions". Droste et al. [4, 5] systematically described a few scenarios of functions and claimed that the scope of the NFL theorem is too enormous to be realistic. Streeter [6] proved that the NFL theorem does not hold over the problems with sufficiently bounded description length. Beyond identifying a subset of problems to which the NFL result can not be applied, Christensen and Oppacher [7] started with a more direct standpoint by proposing the submedian-seeker and demonstrated such an algorithm can outperform random search on certain types of functions. Thereafter, Whitley and Rowe [8] simplified and extended Christensen and Oppacher's work and showed that a more generic subthreshold-seeker can outperform random search on uniformly sampled polynomials in the sense of the number of subthreshold points visited in a given time span.

In the aforementioned studies, the topics may be different, but a common goal is shared – addressing the issue of how general optimization algorithms and search heuristics can be. This study serves the same purpose. Borrowing the notion of Lipschitz functions in real analysis, we introduce the discrete Lipschitz class as an attempt to capture the continuity of a discrete search space and examine this class within the context of the NFL theorem.

1.2 Research Objectives

The property of similarities in objective values within a neighborhood is possessed by many real-world problems, and this thesis primarily aims to address how such a problem structure facilitates the search process, especially in the aspects of:

1. The NFL theorem, as well as its relating studies, provides a pattern to investigate the discrete Lipschitz class in the first place. Starting from the very definitions of the NFL theorem, this study formulates the discrete Lipschitz class on an abstract level as the groundwork from which succeeding inferences can be drawn.

2. As a class of optimization problems, the discrete Lipschitz class will be analyzed under an algorithmic view. In particular, a generalized subthreshold-seeker is proved to outperform random search on the discrete Lipschitz class in theory.

3. Different from previous efforts on various subsets of functions, one major benefit of the discrete Lipschitz class is that it can be regarded as a population from which problems can be sampled. This thesis proposes a sampling-test scheme and conducting numerical experiments with comparisons, as well as demonstrates the theoretical result can be carried over into practice.

1.3 Road Map

This thesis, which consists of seven chapters, is organized as follows:

- Chapter 1 comprises the motivation, objectives and the road map of this thesis so as to adumbrate the contents in the following chapters.
- Chapter 2 briefly reviews the NFL framework to establish and unify the terminology and definitions as preliminaries.
- Chapter 3 introduces the discrete Lipschitz class and describes the relationship between the class and the theorem with a focus on the condition under which the NFL theorem holds over the discrete Lipschitz class.
- Chapter 4 generalizes the subthreshold-seeker and discusses its performance on the discrete Lipschitz class in comparison with random search.
- Chapter 5 proposes a sampling-test scheme on one-dimensional discrete Lipschitz class as an alternative way to examine the effectiveness of optimizers in practice.
- Chapter 6 extends the results in Chapter 5 and explores the possibility of realistically studying higher-dimensional discrete Lipschitz class.
- Chapter 7 summarizes this study and interprets its practical meanings, as well as suggests possible future work.

Chapter 2

A Brief Review of NFL

The No-Free-Lunch (NFL) theorem, in short, states that all algorithms have the same overall performance. As plain as this statement may seem, there are several aspects to be clarified. Firstly, "algorithms" in the realm of NFL are restricted to the scope of "non-repeating black-box algorithms". The term "black-box algorithm", referred to as "blind search" in some literatures, is used to describe the class of evaluation-based algorithms only employing the result of function evaluations as information. The requirement of non-repeating ensures that the search process can be viewed as a permutation of the elements in search space, and revisiting points merely increases the running time without rendering any assistance for identifying the optimum. In fact, when the performance is averaged over all functions, based on NFL, the best an algorithm can do is try not to re-sample.

The concept of "all functions" is another intriguing point for its inherent vagueness. One of the fundamental results in computability is that the set of problems is uncountably infinite. If we consider the the collection of feasible regions of optimization problems as a language, we can easily use the diagonalization method to show that such a language is not recursive. The NFL framework takes a more practical stand here and bypasses this difficulty by considering those functions defined on a finite domain with a finite codomain.

2.1 NFL Framework

Within the NFL framework, the concepts of optimization problems and search algorithms can be formalized in the following definitions:

Definition 1. *Given two finite sets $\mathcal{X}$ and $\mathcal{Y}$, the* ***set of all functions*** *$\mathcal{F}_{\mathcal{X},\mathcal{Y}}$, with respect to $\mathcal{X}$ and $\mathcal{Y}$, is defined as $\mathcal{F}_{\mathcal{X},\mathcal{Y}} := \{f \mid f : \mathcal{X} \to \mathcal{Y}\}$.*

Definition 2. *A* ***trace*** *of length m is a sequence $T_m := ((x_i, y_i))_1^m = ((x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$ with distinct x_i's. "$x \in T_m$" denotes that $x = x_i$ for some $i \in \{1, 2, \ldots, m\}$. Let T_0 be the empty sequence and $\mathcal{T}^\ell$ be the set containing all the traces of a length smaller than or equal to ℓ.*

Definition 3. *Let A_T, where $T \in \mathcal{T}^{|\mathcal{X}|-1}$, be a random variable over $\mathcal{X}$ satisfying that $Prob\{A_T = x\} = 0$ for all $x \in T$. An* ***algorithm*** *A is a collection of such random variables, i.e., $A = \{A_T \mid T \in \mathcal{T}^{|\mathcal{X}|-1}\}$.*

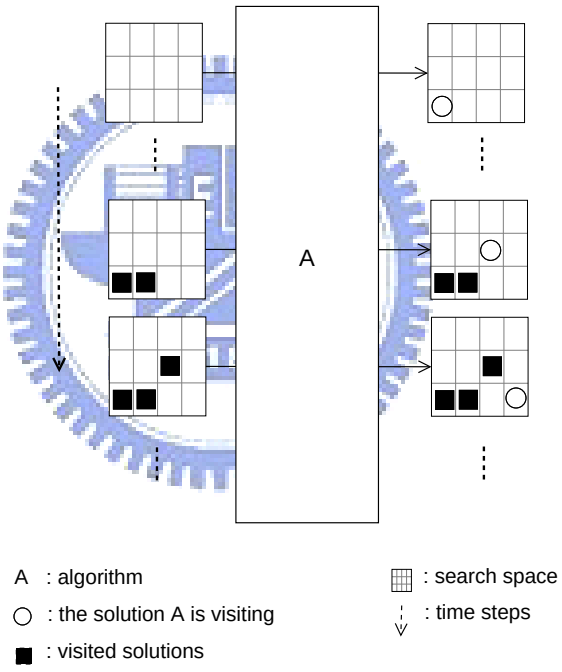


Figure 2.1: An illustration of an algorithm

Definition 4. *The* ***search process*** *of A on f, $S(A, f)$, is the stochastic process $(X_i, Y_i := f(X_i))$ over $\mathcal{X} \times \mathcal{Y}$ defined by $X_1 \sim A_{T_0}$ and $X_{k+1} \sim A_{((X_i,Y_i))_1^k}$. Let $S(A, f, k) := ((X_i, Y_i))_1^k$, and $S_y(A, f, k) := (Y_i)_1^k$ is called the* ***performance vector****.*

Definition 5. *Let $\mathcal{V} := \bigcup_{i=1}^{|\mathcal{X}|} \mathcal{Y}^i$ be the set containing all possible performance vectors. A* ***performance measure*** *is any function mapping $\mathcal{V}$ to $\mathbb{R}$.*

The terminology mostly follows those adopted in [2] and [9] with a few slight modifications applied to avoid the situation that an algorithm is undefinable on a complete trace and to make search processes able to be expressed in a naturally stochastic way. Even though the NFL framework does exert constraints upon the scope of optimization algorithms and problems so that rigorous analysis can proceed, these conditions are not unreasonable for the theorem's intent. Most general-purpose optimizers, equipped without problem-specific knowledge, are essentially black-box algorithms, and the finiteness of the search space agrees with the nature of computer.

Example 1 demonstrates how to represent random search under the NFL framework. It is noteworthy that the non-revisiting property confines random search to the scope of random permutation.

Example 1 (Random search in NFL)**.** *Let R_{T_m} be a random variable that $Prob\{R_{T_m} = x\} = 1/(|\mathcal{X}|-m)$ for all $x \notin T_m$. In the NFL framework, random search can be accordingly defined as $RS := \{R_T \mid T \in \mathcal{T}^{|\mathcal{X}|-1}\}$.*

2.2 NFL Theorem

Now, the NFL theorem can be given as Theorem 1.

Theorem 1 (NFL theorem)**.** *If $v \in \mathcal{V}$ is a performance vector with length ℓ, then*

$$\sum_f Prob\{S_y(A, f, \ell) = v\} = c\,,$$

where c is a constant independent of A.

The complete proof can be found in the original NFL papers [1, 2]. Also, both Droste et al. [5] and Culberson [3] provide simplified proofs.

To rephrase the NFL theorem more plainly, for any performance vector, the expected number of problems on which the performance vector will be generated in the search process is identical for all algorithms. Since the performance measure is a function defined on performance vectors, for any given "score", the expected number of problems on which the score is achieved is exactly the same for all algorithms. Therefore, averaging over all problems, all algorithms performs identically in expectation.

Chapter 3

Discrete Lipschitz Class

3.1 Definition of the Discrete Lipschitz Class

In real analysis, Lipschitz functions refer to the functions with bounded slope. Given a set $\mathcal{C} \subseteq \mathbb{R}$, $f : \mathcal{C} \to \mathbb{R}$ is a *Lipschitz function* if there exists a constant $K > 0$ such that $|f(a) - f(b)| \leq K|a - b|$ for all $a, b \in \mathcal{C}$. The Lipschitz condition is a stronger condition than normal continuity, because any Lipschitz function is uniformly continuous. On the other hand, the functions that are not everywhere differentiable may still be Lipschitz, e.g., $f(x) = |x|$. On a closed interval, the Lipschitz class lies between continuous functions and the functions having continuous derivatives [10].

For the discrete space, there is no such thing as continuity. However, if there is some sort of distance defined in some discrete space, the Lipschitz condition can still be applied, and therefore a natural way to simulate continuity in the discrete space can be obtained. In combinatorics, the spatial structures are typically formed via graph theory. If we view the vertex set as the search space and the edge set as the specification of the geometry, the Lipschitz condition can be transferred here by restricting the difference of objective values between any two adjacent vertexes. The merit of such definition is that we do not put any constraints on the global structure directly such as to demand the functions to be polynomial or the description length to be bounded. Instead, we only expect some similarities of the objective values within a neighborhood in the search space.

Since we will focus on the discrete Lipschitz class in the remainder of this paper, the domain $\mathcal{X}$ is always the vertex set $V(G)$ of a graph G, representing the spatial structure. Hence, the two notations $\mathcal{X}$ and $V(G)$ are used exchangeably. The discrete Lipschitz class

(DLC) can now be introduced.

Definition 6 (Discrete Lipschitz class, DLC)**.** *Given a connected graph G and a finite set $\mathcal{Y} \subset \mathbb{R}$, the corresponding discrete Lipschitz class with Lipschitz constant K is defined as*

$$\mathcal{L}(G, \mathcal{Y}, K) := \{f : V(G) \to \mathcal{Y} \mid \forall \overline{v_1 v_2} \in E(G), |f(v_1) - f(v_2)| \leq K\} \ .$$

Throughout this thesis, the property of $\mathcal{Y}$ of interest is the ordering, so, without loss of generality, $\mathcal{Y}$ is assumed to be a subset of $\mathbb{N}$ of the form $\{0, 1, \ldots, m\}$ unless specified otherwise. $deg(v)$ and $N(v)$ are used to denote the degree and the neighborhood of a vertex v, respectively.

3.2 DLC as Representation

In this section, we discuss the connection between DLC and real-world optimization problems in order to manifest the broadness and the practicality of DLC.

3.2.1 Real-parameter Optimization Problems

The definition of DLC provides a means to represent the intrinsically real-parameter optimization problems through discretization (for practical computing devices). For instance, if a cube $\mathcal{C} \subset \mathbb{R}^n$ is discretized uniformly into a set of grid points, $V(G) = \{x_1, x_2, \ldots, x_M\}^n \subset \mathbb{R}^n$ with $x_{i+1} - x_i = u > 0$, we can let $E(G) = \{\overline{v_i v_j} \mid v_i, v_j \in V(G) \text{ and } \|v_i - v_j\|_1 = u\}$. $\mathcal{L}(G, \mathcal{Y}, K)$ then forms a class containing all functions, defined on $\mathcal{C}$ with the absolute values of partial derivatives upper bounded by K/u, discretized over $V(G)$. Furthermore, since $\mathcal{Y}$ is bounded, this class contains all functions mapping $V(G)$ to $\mathcal{Y}$ with sufficient large K (e.g., $K = \max \mathcal{Y} - \min \mathcal{Y}$).

The simplest case of DLC is the class of functions defined on $\mathbb{R}$, in which the graph representing the spatial structure is a simple path. Figure 3.1 gives an illustrative example of such functions.

Definition 7 (Pathwise discrete Lipschitz class, PDLC)**.** *Given a finite set $\mathcal{Y} \subset \mathbb{R}$ and a simple path $G = \overline{v_1 v_2 \ldots v_n}$, the pathwise discrete Lipschitz class with Lipschitz constant K is defined as $\mathcal{L}(G, \mathcal{Y}, K)$.*

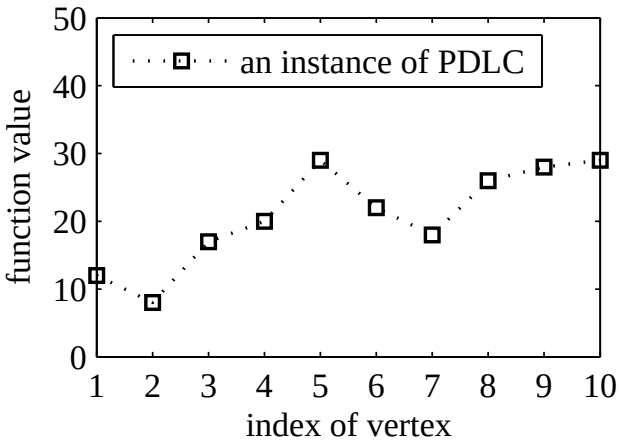


Figure 3.1: A function instance of PDLC with 10 vertexes and $K = 10$.

Higher-dimensional cases can be defined in a similar way and will be addressed later in Chapter 6.

3.2.2 Combinatorial Optimization Problems

Since DLC is discrete in nature, if a combinatorial problem can be represented by a solution-based form, then we can characterize it as DLC by specifying the neighborhood structure and the Lipschitz constant. For instance, the search space of the Traveling Salesman Problem(TSP)[11] consists of all possible tours, i.e. Hamiltonian cycles, and for each tour there is a corresponding edge set containing all edges the tour passes through. Therefore, we can define the neighborhood of a tour as the set of tours which agree on all but two edges with the original path, and the Lipschitz constant can be written in terms of the maximum weight, as shown in the following example:

Example 2 (TSP and DLC)**.** *If the weights of edges are non-negative and upper-bounded by a constant K, then every traveling salesman problem is an instance of $\mathcal{L}(G, \mathcal{Y}, 2K)$, where each vertex of G corresponds to a tour in the original graph and if two tours differ in only two edges, they are considered neighbors.*

Note that G is not the original graph on which the TSP defined.

Also, the minimum spanning tree(MST) problem[11] can be associated to DLC similarly.

Example 3 (MST and DLC)**.** *If the weights of edges are non-negative and upper-bounded by a constant K, then every MST problem is an instance of $\mathcal{L}(G, \mathcal{Y}, 2K)$, where each vertex of G corresponds to a spanning tree in the original graph and if two spanning trees differ in only two edges, they are considered neighbors.*

In addition to graph problems, the optimization version of the set-partition problem[11], which is to partition a set into two equally weighted subsets, can also be connected to DLC. For a set $S \subset \mathbb{N}$, we denote the sum of elements in S as M_S, i.e. $\sum_{x \in S} x = M_S$.

Example 4 (Set-partition problem and DLC)**.** *Given a finite set $S \subset \mathbb{N}$ with $\max S = K$, the solution space of the set-partition problem is the power set of S. For a subset U of S, the corresponding objective value is $f(U) = |M_U - M_S/2|$. Two subsets A and B of S, where $A \subset B$, are considered neighbors if $|B - A| = 1$, and thus $f(A) \leq f(B) = |M_B - M_A + M_A - M_S/2| \leq f(A) + |M_A - M_B| \leq f(A) + K$. Therefore, the set-partition problem on S is an instance of DLC with Lipschitz constant K.*

Generally speaking, if a combinatorial optimization problem involves a weight set, then normally it can be transformed into DLC. In the following section, we will show that mostly the NFL theorem does not hold over DLC, so it is possible to contrive an algorithm outperforming random search on these combinatorial problems.

3.3 DLC and NFL

In this section, we will investigate DLC within the NFL framework and derive a condition under which the the NFL theorem holds. In order to determine whether the NFL theorem holds over a problem class, Schumacher et al. [9] provided a criterion for the NFL theorem based on permutation closure.

Definition 8 (Permutation closure)**.** *If π is a permutation on $\mathcal{X}$, define f_π as $f_\pi(x) := f(\pi(x))$ for all $x \in \mathcal{X}$. $F \subseteq \mathcal{F}_{\mathcal{X},\mathcal{Y}}$ is closed under permutation if for all $f \in F$ and for every permutation π on $\mathcal{X}$, $f_\pi \in F$.*

Lemma 1. *The NFL theorem holds over F if and only if F is closed under permutation.*

Although in [9], this criterion is proposed for deterministic algorithms, since a randomized algorithm is simply a mixed strategy, i.e., a distribution over all possible deterministic strategies [12, 5], this criterion still holds for randomized algorithms in the sense of expectation. Utilizing Lemma 1, a simple criterion for whether or not the NFL result can be applied to a DLC can be obtained.

Theorem 2 (Criterion for NFL on DLC)**.** *Let $\mathcal{L}(G, \mathcal{Y} = \{0, 1, \ldots, m\}, K)$ with $m > K$ be a DLC. NFL holds over $\mathcal{L}(G, \mathcal{Y}, K)$ if and only if G is complete.*

Proof. By Lemma 1, it is sufficient to show that $\mathcal{L}(G, \mathcal{Y}, K)$ is closed under permutation if and only if G is complete.

- If G is complete, for every $f \in \mathcal{L}(G, \mathcal{Y}, K)$, we have $|f(v_i) - f(v_j)| \leq K$ for all v_i and $v_j \in V(G)$. For any permutation π on $\mathcal{X}$ and for all v_i and $v_j \in V(G)$, $|f_\pi(v_i) - f_\pi(v_j)| = |f(\pi(v_i)) - f(\pi(v_j))| \leq K$. Therefore, $f_\pi \in \mathcal{L}(G, \mathcal{Y}, K)$.
- If $\mathcal{L}(G, \mathcal{Y}, K)$ is closed under permutation, suppose for contradiction that G is not complete. The incompleteness and connectivity of G imply that there exist v_i and $v_j \in V(G)$ with $\overline{v_i v_j} \notin E(G)$. Select $v_k \in N(v_i)$, where $N(v_i)$ is the neighborhood of v_i. Obviously, $v_k \neq v_j$. Consider the function $f \in \mathcal{L}(G, \mathcal{Y}, K)$:

$$f(v) = \begin{cases} 0 & \text{if } v = v_i \ ; \\ K+1 & \text{if } v = v_j \ ; \\ K & \text{otherwise.} \end{cases}$$

and the permutation π:

$$\pi(v) = \begin{cases} v_k & \text{if } v = v_j \ ; \\ v_j & \text{if } v = v_k \ ; \\ v & \text{otherwise.} \end{cases}$$

$|f_\pi(v_k) - f_\pi(v_i)| = |f(\pi(v_k)) - f(\pi(v_i))| = |f(v_j) - f(v_i)| = K+1$, so $f_\pi \notin \mathcal{L}(G, \mathcal{Y}, K)$, a contradiction.

□

The completeness of a graph implies that the entire search space is in the same neighborhood. However, such a case is rare in real-world applications, because the degree can be viewed as an indication of dimensions, and the size of search space typically surpasses the number of dimensions significantly. For example, for discretized real-parameter optimization problems, the cardinality of the domain is usually notably larger than the number of dimensions, and hence the corresponding graphs are not complete in most cases. Taking PDLC as an example, when $m > K$, the NFL theorem sustains over a PDLC only if there are merely two vertexes in the problem.

Chapter 4

DLC and Subthreshold-seeker

The subthreshold-seeker (STS), introduced by Whitley and Rowe [8] and proved to outperform random search on uniformly sampled polynomials of one variable, is a metaheuristic that employs the threshold as a switch of local search. In essence, it is a selective local search method as it conducts local search if a given condition is satisfied. In this section, a generalization of subthreshold-seeker is firstly presented, and we will demonstrate that the generalized subthreshold-seeker can outperform random search on DLC.

4.1 Generalized Subthreshold-seeker

In Whitley and Rowe's work, the subthreshold-seeker is an optimization algorithm aiming at functions with a one-dimensional domain, i.e., functions defined on a subset $\mathcal{C} \subseteq \mathbb{R}$. The subthreshold-seeker will successively select a point from the search space uniformly at random (u.a.r.) until a subthreshold point is encountered. Once encountering a subthreshold point, the subthreshold-seeker will search through the quasi-basin where that subthreshold point resides. In Whitley and Rowe's definition, a quasi-basin is a set of contiguous points with objective values below the threshold. In other words, the threshold is used to determine whether the subthreshold-seeker enters the local search phase, and the subthreshold-seeker can be viewed as an optimizer with an exhaustively local search operator.

According to this point of view, we generalize the subthreshold-seeker to the extent that it is applicable to any function of which the domain possesses a neighborhood structure as in Algorithm 1.

Algorithm 1 (Generalized subthreshold-seeker)**.**

procedure SUBTHRESHOLD-SEEKER*($\mathcal{X}$, $\mathcal{Y}$, $N : \mathcal{X} \to 2^{\mathcal{X}}$, $f : \mathcal{X} \to \mathcal{Y}$)*

while *the stopping criterion is not satisfied* ***do***

if *Queue is not empty* ***then***

$x \leftarrow$ *Queue.pop();*

else

Select x from $\mathcal{X}$ u.a.r.

end if

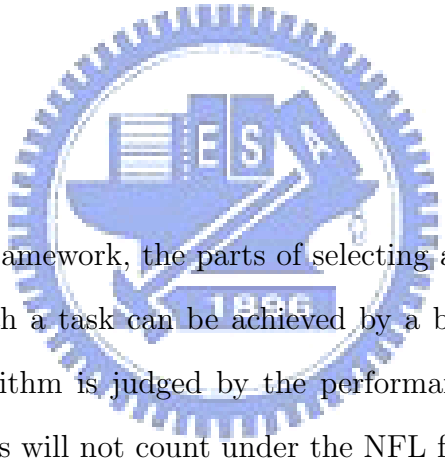
if $f(x) \leq \theta$ ***then***

Queue.push($N(x)$)

end if

end while

end procedure



Following the NFL framework, the parts of selecting and pushing are both restricted to unvisited points. Such a task can be achieved by a bookkeeping manner. Since the performance of an algorithm is judged by the performance vector, all overheads other than function evaluations will not count under the NFL framework.

The only control parameter of the subthreshold-seeker is the threshold. The elegance of the subthreshold-seeker is that it comprises the two fundamental operations of search heuristics, local search and global restart, and yet still stays in a simple form.

4.2 Subthreshold-seeker on DLC

Christensen and Oppacher [7] defined the performance measure as the number of submedian points visited by an algorithm, and Whitley and Rowe [8] generalized this notion to any threshold less than or equal to the median. That is, given a predefined stopping time L and $\alpha \in (0, 1/2]$, the performance measure is the number of points visited in the first L function evaluations with the top $\alpha|\mathcal{X}|$ values in the objective space.

This performance measure may seem odd at the first glance, for typically the performance of an optimizer is measured in terms of the time in which the optimum is located. However, even focusing on functions as simple as unimodal functions that are monotone with respect to the distance from the optimum, the time complexity analysis is still a difficult task. For instance, to the best of our limited knowledge, the time complexity of (1+1)-ES [13] on such functions has not been analyzed until recently [14]. Hence, it seems unlikely to analyze the runtime of an algorithm that is more sophisticated than random search over a broad class of problems. Furthermore, as mentioned in Chapter 2, within the NFL framework, the performance measure can be any function defined on the set containing all performance vectors, and roughly speaking, with more subthreshold points visited, it is more likely to identify a point with a satisfiable objective value. Therefore, Whitley and Rowe's notion appears in between theoretically analyzable and practically meaningful.

For any function f, we define $\beta_\alpha(f)$ be the maximum objective value below the performance threshold, i.e.,

$$\beta_\alpha(f) := \max \left\{ y \in \mathcal{Y} \;\middle|\; \sum_{i=0}^{y} |\{x \in \mathcal{X} \mid f(x) = i\}| \leq \alpha |\mathcal{X}| \right\} .$$

If the set following the "max" notation is empty, then $\beta_\alpha(f)$ is defined to be $-\infty$. Let $\Psi_{\alpha,f}(v)$ be a performance measure that maps a performance vector v to the number of components of v below performance threshold, i.e., $\Psi_{\alpha,f}((v_1, v_2, \ldots, v_L)) = |\{v_i \mid v_i \leq \beta_\alpha(f)\}|$. It is noteworthy that the performance threshold should be distinguished from the algorithmic threshold. The latter should be regarded as a control parameter of the algorithm and hence is not related to the performance measure.

Whitley and Rowe showed that if f is a uniformly sampled polynomials of one variable, and $\beta_\alpha(f)$ is known in advance, setting $\theta = \beta_\alpha(f)$, under certain conditions the subthreshold-seeker outperforms random search on f. In this thesis, we will show that the subthreshold-seeker with θ within some range of codomain, rather than a specific value, will outperform random search in the sense that for all functions in the DLC, the expected number of points below the performance threshold visited by the subthreshold-seeker is greater than or equal to that by random search, and there does exist a function

such that the inequality is strict.

Theorem 3 (Equal or better performance of STS on DLC)**.** *Let* $\mathcal{L}(G, \mathcal{Y} = \{0, 1, \ldots, m\}, K)$ *with* $m > K$ *be a DLC. For all* $f \in \mathcal{L}(G, \mathcal{Y}, K)$ *if the algorithmic threshold* θ *of a subthreshold-seeker satisfies* $\theta \leq \beta_\alpha(f) - K$*, then*

$$E[\Psi_{\alpha,f}(S_y(STS, f, L))] \geq E[\Psi_{\alpha,f}(S_y(RS, f, L))]$$

for all L *with* $1 \leq L \leq |\mathcal{X}|$.

Proof. Let f be any function belonging to $\mathcal{L}(G, \mathcal{Y}, K)$. Suppose $S(STS, f, L) = ((X_{si}, Y_{si}))_{i=1}^{L}$ and $S(RS, f, L) = ((X_{ri}, Y_{ri}))_{i=1}^{L}$. Define the indicator variable I_{si} as $I_{si} = 1$ when $Y_{si} \leq \beta_\alpha(f)$ and $I_{si} = 0$ otherwise, and I_{ri} is defined in a similar way for random search. We can obtain that $\Psi_{\alpha,f}(S_y(STS, f, L)) = \sum_{i=1}^{L} I_{si}$ and $\Psi_{\alpha,f}(S_y(RS, f, L)) = \sum_{i=1}^{L} I_{ri}$.

We prove the theorem by induction on L. Let $U := |\{x \in V(G) \mid f(x) \leq \beta_\alpha(f)\}$ be the total number of points below the performance threshold. When $L = 1$, since both strategies select a point u.a.r. from $\mathcal{X}$ in the first move, clearly $E[I_{s1}] = U/|\mathcal{X}| = E[I_{r1}]$. Suppose $E[\sum_{i=1}^{L} I_{si}] \geq E[\sum_{i=1}^{L} I_{ri}]$ for $1 \leq L < |\mathcal{X}|$. Then,

$$\begin{aligned}
&E[\sum_{i=1}^{L+1} I_{si}] \\
&= E[\sum_{i=1}^{L} I_{si}] + E[I_{sL+1}] \qquad (4.1)\\
&= E[\sum_{i=1}^{L} I_{si}] + \sum_{(x_i)_{i=1}^{L} \in \mathcal{X}^L} E\left[I_{sL+1} \mid (X_{si})_{i=1}^{L} = (x_i)_{i=1}^{L}\right] Prob\{(X_{si})_{i=1}^{L} = (x_i)_{i=1}^{L}\}
\end{aligned}$$

If X_{si} is popped out from the queue, $f(X_{si}) \leq \theta + K \leq \beta_\alpha(f) - K + K = \beta_\alpha(f)$, and hence, $I_{si} = 1$. Otherwise, if X_{si} is selected from $\mathcal{X}$ u.a.r., then $Prob\{I_{si} = 1\} = (U - k)/(|\mathcal{X}| - i + 1)$, where k is the number of points visited in the first $i - 1$ steps with objective values smaller than or equal to $\beta_\alpha(f)$. Let C_L be the set collecting all $(x_i)_{i=1}^{L} \in \mathcal{X}^L$ such that if $(X_{si})_{i=1}^{L} = (x_i)_{i=1}^{L}$, the queue will be nonempty in the $(L+1)$-th

move. Therefore,

$$
\begin{aligned}
&\sum_{(x_i)_{i=1}^L \in \mathcal{X}^L} E\left[I_{sL+1} \mid (X_{si})_{i=1}^L = (x_i)_{i=1}^L\right] Prob\{(X_{si})_{i=1}^L = (x_i)_{i=1}^L\} \\
&= \sum_{(x_i)_{i=1}^L \in C_L} E[I_{sL+1} \mid (X_{si})_{i=1}^L \in C_L] Prob\{(X_{si})_{i=1}^L = (x_i)_{i=1}^L\} + \\
&\quad \sum_{(x_i)_{i=1}^L \notin C_L} E[I_{sL+1} \mid (X_{si})_{i=1}^L \notin C_L] Prob\{(X_{si})_{i=1}^L = (x_i)_{i=1}^L\} \\
&= \sum_{(x_i)_{i=1}^L \in C_L} 1 \cdot Prob\{(X_{si})_{i=1}^L = (x_i)_{i=1}^L\} + \qquad (4.2) \\
&\quad \sum_{(x_i)_{i=1}^L \notin C_L} \frac{U - \left|\{x_i \in (x_i)_{i=1}^L \mid f(x_i) \le \beta_\alpha(f)\}\right|}{|\mathcal{X}| - L} Prob\{(X_{si})_{i=1}^L = (x_i)_{i=1}^L\} \\
&\ge \sum_{(x_i)_{i=1}^L \in \mathcal{X}^L} \frac{U - \left|\{x_i \in (x_i)_{i=1}^L \mid f(x_i) \le \beta_\alpha(f)\}\right|}{|\mathcal{X}| - L} Prob\{(X_{si})_{i=1}^L = (x_i)_{i=1}^L\} \\
&= \sum_{k=0}^{L} \frac{U-k}{|\mathcal{X}| - L} Prob\{\sum_{i=1}^{L} I_{si} = k\}
\end{aligned}
$$

Substituting into (4.1),

$$
\begin{aligned}
E[\sum_{i=1}^{L+1} I_{si}] &\ge \sum_{k=0}^{L} k Prob\{\sum_{i=1}^{L} I_{si} = k\} + \sum_{k=0}^{L} \frac{U-k}{|\mathcal{X}| - L} Prob\{\sum_{i=1}^{L} I_{si} = k\} \\
&= \frac{U}{|\mathcal{X}| - L} + \frac{|\mathcal{X}| - L - 1}{|\mathcal{X}| - L} \sum_{k=0}^{L} k Prob\{\sum_{i=1}^{L} I_{si} = k\} \\
&= \frac{U}{|\mathcal{X}| - L} + \frac{|\mathcal{X}| - L - 1}{|\mathcal{X}| - L} E[\sum_{i=1}^{L} I_{si}] \\
&\ge \frac{U}{|\mathcal{X}| - L} + \frac{|\mathcal{X}| - L - 1}{|\mathcal{X}| - L} E[\sum_{i=1}^{L} I_{ri}] \qquad (4.3) \\
&= \sum_{k=0}^{L} k Prob\{\sum_{i=1}^{L} I_{ri} = k\} + \sum_{k=0}^{L} \frac{U-k}{|\mathcal{X}| - L} Prob\{\sum_{i=1}^{L} I_{ri} = k\} \\
&= E[\sum_{i=1}^{L} I_{ri}] + \sum_{k=0}^{L} E[I_{rL+1} \mid \sum_{i=1}^{L} I_{ri} = k] Prob\{\sum_{i=1}^{L} I_{ri} = k\} \\
&= E[\sum_{i=1}^{L+1} I_{ri}]
\end{aligned}
$$

Inequality (4.3) follows from the induction hypothesis. □

Furthermore, next theorem guarantees that for any $f \in \mathcal{L}(G, \mathcal{Y}, K)$, if there exists a point above performance threshold and the subthreshold-seeker ever enters the local

search phase, the subthreshold-seeker will outperform random search strictly in expectation according to the performance measure $\Psi_{\alpha,f}$.

Theorem 4 (Strictly better performance of STS on DLC)**.** *Let $\mathcal{L}(G, \mathcal{Y} = \{0, 1, \ldots, m\}, K)$ with $m > K$ be a DLC. For all $f \in \mathcal{L}(G, \mathcal{Y}, K)$ and for every subthreshold-seeker STS with $\theta \leq \beta_\alpha(f) - K$ satisfy:*

1. *$\exists v \in V(G)$ with $f(v) > \beta_\alpha(f)$, and*
2. *$\exists v \in V(G)$ with $f(v) \leq \theta$,*

$E[\Psi_{\alpha,f}(S_y(STS, f, L))] > E[\Psi_{\alpha,f}(S_y(RS, f, L))]$ for all $L \in [2, |\mathcal{X}| - 1]$.

Proof. If there are no such functions in $\mathcal{L}(G, \mathcal{Y}, K)$, the theorem holds vacuously. Otherwise, let f be any function satisfying the two conditions and define $((X_{si}, Y_{si}))_{i=1}^{L}$, $((X_{ri}, Y_{ri}))_{i=1}^{L}$, I_{si}, I_{ri}, U, and C_L in the same way as in Theorem 3. We prove by induction on L. When $L = 2$, since in the second step, the queue is nonempty if and only if $f(X_{si}) \leq \theta$, $C_1 = \{v \in V(G) \mid f(v) \leq \theta\} \neq \emptyset$ by Condition (2). Therefore,

$$
\begin{aligned}
&E[I_{s1} + I_{s2}] \\
=&E[I_{s1}] + \sum_{x \in \mathcal{X}} E[I_{s2} \mid X_{s1} = x] Prob\{X_{s1} = x\} \\
=&E[I_{s1}] + \sum_{x: f(x) \leq \theta} 1 \cdot Prob\{X_{s1} = x\} + \sum_{x: \theta < f(x) \leq \beta_\alpha(f)} \frac{U-1}{|\mathcal{X}|-1} Prob\{X_{s1} = x\} \\
&+ \sum_{x: f(x) > \beta_\alpha(f)} \frac{U}{|\mathcal{X}|-1} Prob\{X_{s1} = x\} \\
>&E[I_{s1}] + \sum_{x: f(x) \leq \beta_\alpha(f)} \frac{U-1}{|\mathcal{X}|-1} Prob\{X_{s1} = x\} + \sum_{x: f(x) > \beta_\alpha(f)} \frac{U}{|\mathcal{X}|-1} Prob\{X_{s1} = x\} \\
=&E[I_{s1}] + \sum_{k \in \{0,1\}} \frac{U-k}{|\mathcal{X}|-1} Prob\{I_{s1} = k\} \\
=&E[I_{r1}] + \sum_{k \in \{0,1\}} \frac{U-k}{|\mathcal{X}|-1} Prob\{I_{r1} = k\} \\
=&E[I_{r1} + I_{r2}]
\end{aligned}
$$

The inequality follows from $C_1 \neq \emptyset$ and $(U-1)/(|\mathcal{X}|-1) < 1$, for Condition (1) implies $U < |\mathcal{X}|$. For induction hypothesis, suppose $E[\sum_{i=1}^{L} I_{si}] > E[\sum_{i=1}^{L} I_{ri}]$ for L with $2 \leq$

$L < |\mathcal{X}| - 1$. In the $(L+1)$-th step, from the proof of Theorem 3, we always have

$$\begin{aligned} E[\sum_{i=1}^{L+1} I_{si}] &\geq \frac{U}{|\mathcal{X}| - L} + \frac{|\mathcal{X}| - L - 1}{|\mathcal{X}| - L} E[\sum_{i=1}^{L} I_{si}] \\ &> \frac{U}{|\mathcal{X}| - L} + \frac{|\mathcal{X}| - L - 1}{|\mathcal{X}| - L} E[\sum_{i=1}^{L} I_{ri}] \\ &= E[\sum_{i=1}^{L+1} I_{ri}] \end{aligned} \tag{4.4}$$

Since $(|\mathcal{X}| - L - 1)/(|\mathcal{X}| - L) > 0$ when $L < |\mathcal{X}| - 1$, and $E[\sum_{i=1}^{L} I_{si}] > E[\sum_{i=1}^{L} I_{ri}]$ from the induction hypothesis, Inequality (4.4) is strict. □

Let $d := \max\{deg(v) \mid v \in V(G)\}$ be the maximum degree of the graph and $dis(u, v)$ be the length of the shortest path from u to v. For any subthreshold-seeker, if we are able to set its θ within some interval, the following corollary gives a sufficient condition of the existence of functions on which the subthreshold-seeker strictly outperforms random search.

Corollary 1. *Let $\mathcal{L}(G, \mathcal{Y} = \{0, 1, \ldots, m\}, K)$ be a DLC. Given $\alpha \in (0, 1/2]$ and an integer $C > 1$ with $CK + 1 \leq m$, if*

$$\alpha|V(G)| > \frac{d(d-1)^C - 2}{d - 2},$$

then there exists a function $f \in \mathcal{L}(G, \mathcal{Y}, K)$ such that

$$E[\Psi_{\alpha,f}(S_y(STS, f, L))] > E[\Psi_{\alpha,f}(S_y(RS, f, L))]$$

for all L with $2 \leq L \leq |\mathcal{X}| - 1$, where STS is a subthreshold-seeker with $\theta \in \beta_\alpha(f) - [K, CK]$.

Proof. We prove this corollary constructively. Select a vertex v_0 from $V(G)$ arbitrarily. Consider the function f defined as

$$f(v) = \begin{cases} 0 & \text{if } v = v_0 \text{ ;} \\ dis(v, v_0)K & \text{if } 1 \leq dis(v, v_0) \leq C \text{ ;} \\ CK + 1 & \text{otherwise.} \end{cases}$$

Since

$$
\begin{aligned}
&|v_o| + |\{v \in V(G) \mid 1 \leq dis(v, v_0) \leq C\}| \\
&\leq 1 + \left(d + d(d-1) + d(d-1)^2 + \ldots + d(d-1)^{C-1}\right) \\
&= 1 + \frac{d\left[(d-1)^C - 1\right]}{(d-1) - 1} \\
&= \frac{d(d-1)^C - 2}{d-2} < \alpha|V(G)| \ ,
\end{aligned}
$$

from the definition of $\beta_\alpha(f)$, $\beta_\alpha(f) = CK$. Furthermore, there must exist $v_1 \in V(G)$ that $f(v_1) = CK + 1$, for $|v_o| + |\{v \in V(G) \mid 1 \leq dis(v, v_0) \leq C\}| < |V(G)|$. Therefore, we have $f(v_0) \leq \theta$ and $f(v_1) > \beta_\alpha(f)$. Thereby Theorem 4 can be applied. □

Combining Theorem 3 and Theorem 4, if we manage to set $\theta \leq \beta_\alpha(f) - K$, the subthreshold-seeker will perform at least as good as random search on a DLC. If the subthreshold-seeker has a chance to conduct local search, it will strictly outperform random search. Estimating a θ within some range should be more practical than gauging a specific value such as $\beta_\alpha(f)$. In next section, we will explore this possibility and empirically confirm the theoretical results obtained in this section by proposing and adopting a sampling-test scheme.

Chapter 5

Sampling-test Scheme for PDLC

Conventionally, the effectiveness of an optimizer is examined via experiments on a suite of test functions that serves as a benchmark. These test functions are selected according to some prior knowledge of the importance thereof. Here we propose and adopt a different approach in order to confirm the theoretical results obtained in the previous section from an empirical aspect. We draw a sample of functions randomly from PDLC in a manner similar to select respondents in a campaign survey and conduct experiments on these sampled functions. There is no bias in favor of which functions should be selected. We expect the arbitrariness delivers information about the composition of the problem class.

A uniform sampler for PDLC is firstly given in Section 5.1. Experiments are then presented to summarize this section and demonstrate how the Lipschitz condition facilitates the search process in a practical standpoint.

5.1 A Uniform Sampler for PDLC

In order to conduct the sampling test, we need a uniform sampler in the first place. The following algorithm generates problem instances of PDLC with Lipschitz constant K uniformly at random (u.a.r.)

Algorithm 2 (Uniform PDLC Sampler)**.**

procedure UNIFORM PDLC SAMPLER$(\overline{v_1 v_2 \dots v_n}, \mathcal{Y} = \{0, 1, \dots, m\}, K)$

$f(v_1) \leftarrow Uniform([0, m])$

$i \leftarrow 2$

while $i \leq n$ ***do***

$f(v_i) \leftarrow f(v_{i-1}) + Uniform([-K, K])$

$i \leftarrow i + 1$

if $f(v_i) > m$ *or* $f(v_i) < 0$ ***then***

$f(v_1) \leftarrow Uniform([0, m])$

$i \leftarrow 2$;

end if

end while

return f

end procedure

Remark 1. *The sequence of objective values,* $f(v_i)$*, forms a martingale.*

Here $Uniform([a, b])$ denotes the function that selects an integer u.a.r. from the closed interval $[a, b]$. Such a sampler belongs to the category of accept-reject algorithms [15]. It generates a problem instance with bounded difference between any two successive vertexes u.a.r., and if the instance at hand exceeds the range of the codomain, the sampler rejects the instance. The accept-reject mechanism guarantees the uniformity. Once the sampler halts, the output is always an instance of the PDLC.

Since this sampler is Las Vegas, we need to address its time complexity for the practicality. For each candidate instance, the sampler will go through at most $|\mathcal{X}|$ steps to assign all the vertex objective values, so it remains to show how many candidate instances it takes to generate a legit instance successfully. The accept-reject process is geometrically distributed, and therefore the expected number of instances consumed is the inverse of the acceptance probability. The following theorem provides an upper bound for the rejection probability.

Lemma 2. *Suppose* $|\mathcal{Y}| = 2m + 1$*, where* m *is an integer, and* $|\mathcal{X}| = n$*. If*

$$m > \sqrt{\frac{(n-1)(K^2+K)}{3}} \geq 2\,,$$

then the rejection probability is less than

$$\frac{4\sqrt{(n-1)(K^2+K)}}{\sqrt{3}|\mathcal{Y}|} - \frac{4(n-1)(K^2+K)}{3|\mathcal{Y}|^2} + \frac{5}{|\mathcal{Y}|}\,.$$

Proof. Without loss of generality, suppose $\mathcal{Y} = \{-m, -m+1, \ldots, m\}$. Let (K_i) be a sequence of i.i.d. random variables that $K_i = j$ with probability $1/(2K+1)$ for $j \in \{-K, -K+1, \ldots, K\}$ and $S_j := \sum_{i=1}^{j} K_i$. When $f(v_1) = i$, the instance is rejected if and only if $i + S_j \geq m+1$ or $i + S_j \leq -m-1$ for some $1 \leq j \leq n-1$, so the occurrence of rejection always implies $\max_{1\leq j\leq n-1} |S_j| \geq \min\{|m+1-i|, |-m-1-i|\}$. Moreover, the symmetry indicates that $Prob\{\text{rejection} \mid f(v_1) = i\} = Prob\{\text{rejection} \mid f(v_1) = -i\}$ for $|i| \leq m$. Therefore,

$$
\begin{aligned}
&Prob\{\text{rejection}\} \\
&= \sum_{i=-m}^{m} Prob\{\text{rejection} \mid f(v_1) = i\} Prob\{f(v_1) = i\} \\
&= \frac{\sum_{i=-m}^{m} Prob\{\text{rejection} \mid f(v_1) = i\}}{2m+1} \\
&\leq \frac{Prob\{\max_{1\leq j\leq n-1} |S_j| \geq m+1\} + 2\sum_{i=1}^{m} Prob\{\max_{1\leq j\leq n-1} |S_j| \geq m+1-i\}}{2m+1} \\
&= \frac{Prob\{\max_{1\leq j\leq n-1} |S_j| \geq m+1\} + 2\sum_{i=1}^{m} Prob\{\max_{1\leq j\leq n-1} |S_j| \geq i\}}{2m+1} .
\end{aligned}
$$

Using Kolmogorov's inequality [16], we can get

$$
Prob\{\max_{1\leq j\leq n-1} |S_j| \geq i\} \leq \min\left\{\frac{Var[S_{n-1}]}{i^2}, 1\right\} .
$$

Since $Var[K_i] = 2(1^2 + 2^2 + \ldots + K^2)/(2K+1) = (K^2+K)/3$, $Var[S_{n-1}] = (n-1)Var[K_i] = (n-1)(K^2+K)/3$. Moreover, $Var[S_{n-1}]/i^2 \leq 1$ if and only if $i \geq \sqrt{Var[S_{n-1}]}$, we have

$$
\begin{aligned}
&Prob\{\text{rejection}\} \\
&\leq \frac{\frac{Var[S_{n-1}]}{(m+1)^2} + 2\left(\sum_{i=1}^{\left\lceil\sqrt{Var[S_{n-1}]}\right\rceil - 1} 1 + \sum_{i=\left\lceil\sqrt{Var[S_{n-1}]}\right\rceil}^{m} \frac{Var[S_{n-1}]}{i^2}\right)}{2m+1} \\
&\leq \frac{\frac{Var[S_{n-1}]}{(m+1)^2} + 2\left(\left\lceil\sqrt{Var[S_{n-1}]}\right\rceil - 1 + Var[S_{n-1}] \int_{x=\left\lceil\sqrt{Var[S_{n-1}]}\right\rceil - 1}^{m} x^{-2} dx\right)}{2m+1} \\
&\leq \frac{\frac{Var[S_{n-1}]}{(m+1)^2} + 2\left(\sqrt{Var[S_{n-1}]} - \frac{Var[S_{n-1}]}{m} + \frac{Var[S_{n-1}]}{\left\lceil\sqrt{Var[S_{n-1}]}\right\rceil - 1}\right)}{2m+1} .
\end{aligned}
$$

Since $x/(x-1)$ decreases when $x > 1$, we obtain

$$\begin{aligned}\frac{Var[S_{n-1}]}{\left\lceil\sqrt{Var[S_{n-1}]}\right\rceil - 1} &\le \frac{Var[S_{n-1}]}{\sqrt{Var[S_{n-1}]} - 1} \\ &= \sqrt{Var[S_{n-1}]} + \frac{\sqrt{Var[S_{n-1}]}}{\sqrt{Var[S_{n-1}]} - 1} \\ &\le \sqrt{Var[S_{n-1}]} + 2 \ .\end{aligned}$$

According to the hypothesis that $Var[S_{n-1}]/(m+1)^2 < 1$,

$$\begin{aligned}Prob\{\text{rejection}\} &< \frac{4\sqrt{Var[S_{n-1}]} - \frac{2Var[S_{n-1}]}{m} + 5}{2m+1} \\ &= \frac{4\sqrt{(n-1)(K^2+K)}}{\sqrt{3}(2m+1)} - \frac{2(n-1)(K^2+K)}{3m(2m+1)} + \frac{5}{2m+1} \\ &< \frac{4\sqrt{(n-1)(K^2+K)}}{\sqrt{3}|\mathcal{Y}|} - \frac{4(n-1)(K^2+K)}{3|\mathcal{Y}|^2} + \frac{5}{|\mathcal{Y}|} \ .\end{aligned}$$

□

Theorem 5 (Upper bound for the rejection probability)**.** *Define* $m := \lfloor(|\mathcal{Y}|-1)/2\rfloor$*. If* $m > \sqrt{(n-1)(K^2+K)/3} \ge 2$ *, then the rejection probability is less than*

$$\frac{4\sqrt{(n-1)(K^2+K)}}{\sqrt{3}|\mathcal{Y}|} - \frac{4(n-1)(K^2+K)}{3|\mathcal{Y}|^2} + O\left(|\mathcal{Y}|^{-1}\right) \ .$$

Proof. If $|\mathcal{Y}| = 2m+1$, then we are done by the previous lemma. Otherwise, if $|\mathcal{Y}| = 2m+2$, without loss of generality, suppose that $\mathcal{Y} = \{-m, -m+1, \ldots, m+1\}$ and let $\mathcal{Y}' = \{-m, -m+1, \ldots, m\}$. Therefore,

$$\begin{aligned}&Prob\{\text{rejection}\} \\ =&Prob\{f(v_1) \in \mathcal{Y}'\}Prob\{\text{rejection} \mid f(v_1) \in \mathcal{Y}'\} \\ &+ Prob\{f(v_1) \notin \mathcal{Y}'\}Prob\{\text{rejection} \mid f(v_1) \notin \mathcal{Y}'\} \\ =&\left(\frac{2m+1}{2m+2}\right) Prob\{\text{rejection} \mid f(v_1) \in \mathcal{Y}'\} \\ &+ \left(\frac{1}{2m+2}\right) Prob\{\text{rejection} \mid f(v_1) = m+1\} \ .\end{aligned}$$

When $f(v_1) \in \mathcal{Y}'$, if f exceeds the range of $\mathcal{Y}$, then f also exceeds the range of $\mathcal{Y}'$, so from the previous lemma we have

$$Prob\{\text{rejection} \mid f(v_1) \in \mathcal{Y}'\} < \frac{4\sqrt{(n-1)(K^2+K)}}{\sqrt{3}(2m+1)} - \frac{4(n-1)(K^2+K)}{3(2m+1)^2} + \frac{5}{2m+1} \ .$$

As a result,

$$
\begin{aligned}
&Prob\{\text{rejection}\} \\
&\leq \left(\frac{2m+1}{2m+2}\right) Prob\{\text{rejection} \mid f(v_1) \in \mathcal{Y}'\} + \left(\frac{1}{2m+2}\right) \\
&< \frac{4\sqrt{(n-1)(K^2+K)}}{\sqrt{3}(2m+2)} - \frac{4(n-1)(K^2+K)}{3(2m+1)(2m+2)} + \frac{6}{2m+2} \\
&< \frac{4\sqrt{(n-1)(K^2+K)}}{\sqrt{3}|\mathcal{Y}|} - \frac{4(n-1)(K^2+K)}{3|\mathcal{Y}|^2} + O\left(|\mathcal{Y}|^{-1}\right)
\end{aligned}
$$

□

Corollary 2. *If* $|\mathcal{Y}| = C\sqrt{(n-1)(K^2+K)} > C \cdot 2\sqrt{3}$ *for some constant* $C \geq \sqrt{3}$*, then the rejection probability is less than*

$$\frac{4\sqrt{3}C - 4}{3C^2} + O\left(|\mathcal{Y}|^{-1}\right) .$$

Proof. If $C \geq \sqrt{3}$,

$$
\begin{aligned}
m = \left\lfloor \frac{|\mathcal{Y}|-1}{2} \right\rfloor &\geq \frac{|\mathcal{Y}|}{2} - 1 \\
&\geq \frac{\sqrt{3(n-1)(K^2+K)}}{2} - 1 \\
&= \sqrt{\frac{(n-1)(K^2+K)}{3}} + \frac{\sqrt{(n-1)(K^2+K)}}{2\sqrt{3}} - 1 \\
&> \sqrt{\frac{(n-1)(K^2+K)}{3}} .
\end{aligned}
$$

Substituting $\sqrt{(n-1)(K^2+K)}/|\mathcal{Y}|$ by $1/C$ and applying Theorem 5, the corollary is proved. □

For instance, if $C = \sqrt{3}$ and $|\mathcal{Y}|$ is so large that $O(|\mathcal{Y}|^{-1})$ is negligible, the expected number of instances consumed is no more than 9. Multiplying the time to assign all vertexes values, the expected runtime, in terms of the number of assignments, is no more than $9|\mathcal{X}|$. In other words, asymptotically speaking, if $|\mathcal{X}|$ and $|\mathcal{Y}|$ are about equal and $|\mathcal{Y}|$ is larger than K^2 to some extent, then the expected runtime is approximately linear.

5.2 Experimental Settings and Results

As demonstrated in Chapter 4, the virtues of the subthreshold-seeker rely on a proper algorithmic threshold. Although the main results in Chapter 4 hold when $\theta \leq \beta_\alpha(f) - K$,

because we do not set a performance threshold literally to scrutinize how many subthreshold points are visited in real-world applications, in an experimental setting, we can examine the subthreshold-seeker more practically in terms of the time to identify the optimum. Therefore, the algorithmic threshold should be utilized for optimization, or more specifically, to minimize the objective function in this case.

We will compare the subthreshold-seeker with random search. Here we present three different subthreshold-seekers. For the theoretical purpose, the first one uses the actual median of all objective values, in the form of exterior knowledge, as θ. The second one firstly selects a 100 points u.a.r. and then employs the calculated median as θ. The third one also starts with obtaining 100 points u.a.r., but it computes the mean and the standard deviation of these points and sets θ to the mean minus the standard deviation. Moreover, the three subthreshold-seekers and random search obey the NFL framework and hence are non-repeating.

In advance of experiments, we need to determine the size of the set PDLC problems to be sampled. Suppose we want to estimate a population proportion $q \in [0, 1]$. We draw a sequence of samples uniformly and independently from the population with replacement. For each sample, we observe if it belongs to the variety of interest. With a large sample size, we expect the proportion in the sample approximates the real proportion. The following theorem depicts the relationship between the sample size and the error bound.

Theorem 6 (Sample size and error bound)**.** *Let (Z_i) be a sequence of i.i.d. indicator variables with $E[Z_i] = q$. For all $\delta, \epsilon \in (0, 1)$, if*

$$n \geq -\frac{\ln(\delta/2)}{2\epsilon^2} ,$$

then

$$Prob\left\{\left|\frac{\sum_{i=1}^{n} Z_i}{n} - q\right| > \epsilon\right\} \leq \delta .$$

Proof. Let $\overline{Z} = (\sum_{i=1}^{n} Z_i)/n$. Applying Hoeffding's inequality [17], for $0 < \epsilon < 1 - q$, we have

$$Prob\{\overline{Z} - q > \epsilon\} \leq e^{-2n\epsilon^2} ,$$

and for $0 < \epsilon < q$,

$$Prob\{\overline{Z} - q < -\epsilon\} \leq e^{-2n\epsilon^2} .$$

Moreover, if $\epsilon \geq 1 - q$,

$$Prob\{\overline{Z} - q > \epsilon\} \leq Prob\{\overline{Z} > 1\} = 0 \leq e^{-2n\epsilon^2} .$$

Similarly, if $\epsilon \geq q$,

$$Prob\{\overline{Z} - q < -\epsilon\} \leq Prob\{\overline{Z} < 0\} = 0 \leq e^{-2n\epsilon^2} .$$

Hence, we conclude that for all $\epsilon \in (0, 1)$,

$$Prob\{|\overline{Z} - q| > \epsilon\} \leq 2e^{-2n\epsilon^2} .$$

Finally,

$$n \geq -\frac{\ln(\delta/2)}{2\epsilon^2}$$

implies $2e^{-2n\epsilon^2} \leq \delta$, and we complete the proof. □

In particular, with the conventional setting of $(\epsilon, \delta) = (0.03, 0.05)$, a sample of size $2,050$ suffices. In other words, if we draw a sample of size $2,050$, $[\overline{Z} - 0.03, \overline{Z} + 0.03]$ forms a confidence interval for q with confidence level at least 95%.

The sampler generates 2,050 instances of PDLC with $(|\mathcal{X}|, |\mathcal{Y}|) = (10^4, 10^4)$, $(10^5, 10^5)$, and $(10^6, 10^6)$, respectively. The Lipschitz constant K is set to 100 for the concern of execution time, as previously discussed. For each problem instance, we test each algorithm for 50 independent runs. If the average time of a subthreshold-seeker to find the optimum is less than that of random search, the instance is counted as a success. We also count the number of instances that a subthreshold-seeker outperforms random search by a 20% margin, i.e., the instance where the average optimization time of a subthreshold-seeker is less than 80% of that of random search. Table 5.1 displays the empirical results.

All three subthreshold-seekers outperform random search in most of the sampled problem instances. Furthermore, the subthreshold-seeker with $\theta = \hat{\mu} - \hat{\sigma}$ outperforms random search in all 2,050 instances sampled, even with the requirement of a 20% margin. The statistical significance of such results is obvious to see: Suppose the population proportion

Table 5.1: Successful rate of STS

θ	category	$(\lvert\mathcal{X}\rvert, \lvert\mathcal{Y}\rvert)$ $(10^4, 10^4)$	$(10^5, 10^5)$	$(10^6, 10^6)$
γ	$>$	0.9995 (2,049)	0.9985 (2,047)	0.9976 (2,045)
	$> .2$	0.9951 (2,040)	0.9620 (1,972)	0.9624 (1,973)
$\hat{\gamma}$	$>$	0.9995 (2,049)	0.9990 (2,048)	0.9985 (2,047)
	$> .2$	0.9937 (2,037)	0.9620 (1,972)	0.9732 (1,995)
$\hat{\mu} - \hat{\sigma}$	$>$	1.0000 (2,050)	1.0000 (2,050)	1.0000 (2,050)
	$> .2$	1.0000 (2,050)	1.0000 (2,050)	1.0000 (2,050)

γ: median. $\hat{\gamma}$: estimated median. $\hat{\mu}$: estimated mean. $\hat{\sigma}$: estimated standard deviation. "$>$": proportion of instances where the subthreshold-seeker outperforms random search. "$> .2$": proportion of instances where the subthreshold-seeker outperforms random search by a 20% margin.

Table 5.2: Mean time steps to locate the minimum

algorithm	$(\lvert\mathcal{X}\rvert, \lvert\mathcal{Y}\rvert)$ $(10^4, 10^4)$	$(10^5, 10^5)$	$(10^6, 10^6)$
STS, $\theta = \gamma$	2037.58	22913.23	229232.26
STS, $\theta = \hat{\gamma}$	2221.44	23170.58	229532.04
STS, $\theta = \hat{\mu} - \hat{\sigma}$	918.29	8095.78	80322.92
random search	4972.50	49724.74	496912.49

γ: median. $\hat{\gamma}$: estimated median. $\hat{\mu}$: estimated mean. $\hat{\sigma}$: estimated standard deviation.

that the subthreshold-seeker with $\theta = \hat{\mu} - \hat{\sigma}$ outperforms random search is q. To obtain the result that random search is outperformed in all instances, the probability is q^{2050}. Even if q is as high as 0.995, the above probability is just 0.000034. To more formally rephrase, if the null hypothesis is "$q \leq 0.995$", the p-value is merely 0.000034.

Table 5.2 displays the averaged optimization time over the 2,050 sampled problem instances. The subthreshold-seeker with $\theta = \hat{\mu} - \hat{\sigma}$ outperforms others by a significant margin. Random search averages approximately $|\mathcal{X}|/2$ to find the minimum, which is expected. The subthreshold-seeker using the actual median and the one using the sample median both take about half time steps of that needed by random search to optimize the function.

The subthreshold-seekers with $\theta = \hat{\mu} - \hat{\sigma}$ and $\theta = \hat{\gamma}$ are indeed black-box algorithms, for there is no exterior knowledge exerted and the only information they can use are function evaluations, but they outperform random search by a remarkable difference.

5.3 The Estimation of Median

In this section, we address the issue of the estimation of median as an echo to Section 5.2 section and Corollary 1 in Chapter 4.

The performance difference between $\theta = \hat{\gamma}$ and $\theta = \gamma$ is insignificant, suggesting that in this case, an estimation of median may be adequate. Suppose that P with $|P| = N$ is a subset of real numbers, and for all $i \in P$, $R(i)$ is defined to be the rank (i.e., ordering) of i in P. For instance, $R(\min P) = 1$ and $R(\max P) = N$. For simplicity, we assume that N is odd and hence the median of P is the element i with $R(i) = \lceil N/2 \rceil$. Now we want to estimate the median of P. If a point sample S of size n, where n is assumed odd, is drawn by successively selecting an element u.a.r. from P with replacement, the estimated median, γ, is presumed to be the sampled median, and we want the error is bounded by $\epsilon > 0$, i.e., $|R(\gamma) - \lceil N/2 \rceil| \leq \epsilon N$.

If $R(\gamma) < \lceil N/2 \rceil - \epsilon N$, there are at least $\lceil n/2 \rceil$ selections with ranks less than $\lceil N/2 \rceil - \epsilon N$. Let X_i be the indicator variable that indicates if the i-th selection is less than $\lceil N/2 \rceil - \epsilon N$, $X_i = 1$ with probability $p := (\lceil N/2 \rceil - \lfloor \epsilon N \rfloor - 1)/N$. $R(\gamma) < \lceil N/2 \rceil - \epsilon N$ if and only if $\sum_{i=1}^{n} X_i \geq \lceil n/2 \rceil$. Since $E[\sum_{i=1}^{n} X_i] = np$, applying another form of Hoeffding's inequality [17], we have

$$
\begin{aligned}
Prob\left\{ R(\gamma) < \left\lceil \frac{N}{2} \right\rceil - \epsilon N \right\} &= Prob\left\{ \sum_{i=1}^{n} X_i \geq \left\lceil \frac{n}{2} \right\rceil \right\} \\
&\leq Prob\left\{ \sum_{i=1}^{n} X_i \geq \frac{n}{2} \right\} \\
&= Prob\left\{ \frac{1}{n} \sum_{i=1}^{n} X_i \geq p + (\frac{1}{2} - p) \right\} \\
&\leq \left[\left(\frac{p}{p + \frac{1}{2} - p} \right)^{p + \frac{1}{2} - p} \left(\frac{1-p}{1 - p - (\frac{1}{2} - p)} \right)^{1 - p - (\frac{1}{2} - p)} \right]^{n} \\
&= [4p(1-p)]^{\frac{n}{2}} \ .
\end{aligned}
$$

Moreover, the symmetry implies that

$$
Prob\left\{ R(\gamma) > \left\lceil \frac{N}{2} \right\rceil + \epsilon N \right\} \leq [4p(1-p)]^{\frac{n}{2}} \ .
$$

Therefore,

$$Prob\left\{\left|R(\gamma)-\left\lceil\frac{N}{2}\right\rceil\right|>\epsilon N\right\}\leq 2\left[4p(1-p)\right]^{\frac{n}{2}}\ .$$

Now the only quantity left is p. By definition,

$$p=\frac{\left\lceil\frac{N}{2}\right\rceil-\lfloor\epsilon N\rfloor-1}{N}\approx\frac{1}{2}-\epsilon\ .$$

For instance, if we set $\epsilon=0.1$ and $n=100$, the probability of exceeding the error bound is less than 0.26. If the sample size n increases to $2,000$, even with a small $\epsilon=0.03$, the probability reduces to just 0.054. It is noteworthy that the effect of the population size N is negligible. Therefore, the required number of samples remains the same, even if the search space is immense. Although in real-world applications P is usually a multiset, if the multiplicities of P are not too large, such a gauge should not diverge significantly.

Chapter 6

Toward Higher Dimensions

In this chapter, we attempt to extend the sampling-test scheme proposed in Chapter 5 to higher dimensions and examine a few methods to generate planar DLC, i.e. two-dimensional DLC.

Definition 9 (Planar DLC)**.** *Suppose the graph G is defined as $V(G) = [n]^2$ and $E(G) = \{\overline{v_1v_2} \mid \|v_1 - v_2\|_1 = 1\}$ and the codomain is a finite set $\mathcal{Y} \subset \mathbb{R}$. The planar discrete Lipschitz class with Lipschitz constant K is defined as $\mathcal{L}(G, \mathcal{Y}, K)$.*

Here the notation $[n]$ signifies the set $\{1, 2, \ldots, n\}$.

Figure 6.2 illustrates an instance of planar DLC.

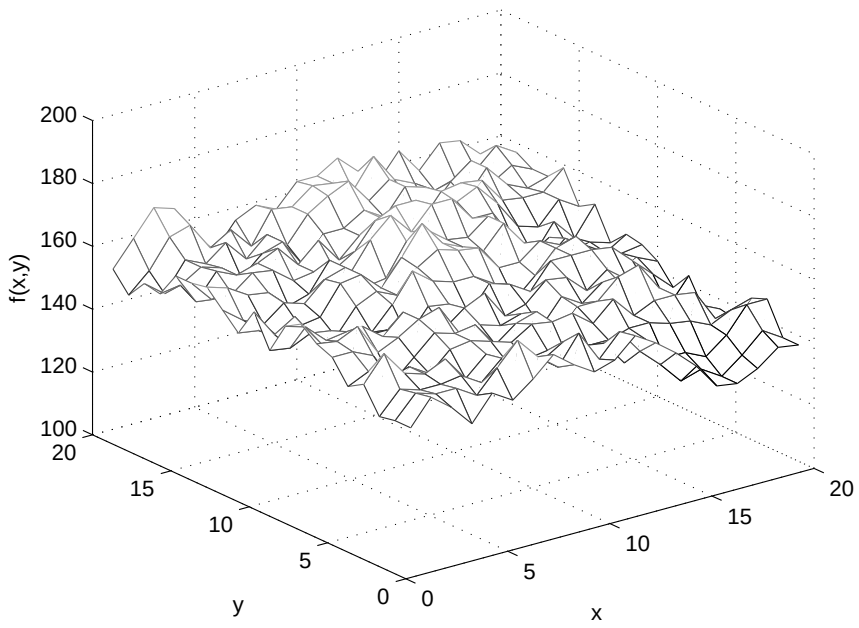


Figure 6.1: An instance of planar DLC with 20^2 vertexes and $K = 10$

In this chapter, the graph G is defined as $V(G) = [n]^2$ and $E(G) = \{\overline{v_1 v_2} \mid \|v_1 - v_2\|_1 = 1\}$ unless specified otherwise.

6.1 Accept-reject Sampler for Planar DLC

In this section, we adopt the accept-reject algorithm proposed in Chapter 5 and modify it into an uniform sampler for planar DLC.

Algorithm 3 (Uniform Planar DLC Sampler)**.**

procedure UNIFORM PLANAR DLC SAMPLER*(G, $\mathcal{Y} = \{0, 1, \ldots, m\}, K$)*

Fix a rooted spanning tree T of G

$f(root) \leftarrow Uniform([0, m])$

for *each node v* ***do***

$f(v) \leftarrow f(parent(v)) + Uniform([-K, K])$

end for

if $\exists v_i, v_j \in V(G)$ *such that* $\overline{v_i v_j} \in E(G)$ *and* $|f(v_i) - f(v_j)| > K$ ***then***

Reject and restart

end if

if $\exists v \in V(G)$ *such that* $f(v) \notin \mathcal{Y}$ ***then***

Reject and restart

end if

Return f

end procedure

This sampler uniformly generate instances of DLC defined on T, and if the instance at hand is in $\mathcal{L}(G, \mathcal{Y}, K)$, the sampler will halt and output the instance. In fact, Algorithm 3 can be applied to generate not only planar DLC but also all DLC in general, since the requirement of connectivity in Definition 6 guarantees the existence of a spanning tree.

However, in addition to rejecting out-of-codomain instances, for cyclic graphs, this sampler needs to check additionally that for any edge not in the spanning tree the difference between its two endpoints is less than or equal to K. Therefore, the acceptance probability decreases exponentially with respect to $|E(G)| - |E(T)|$, which implies that

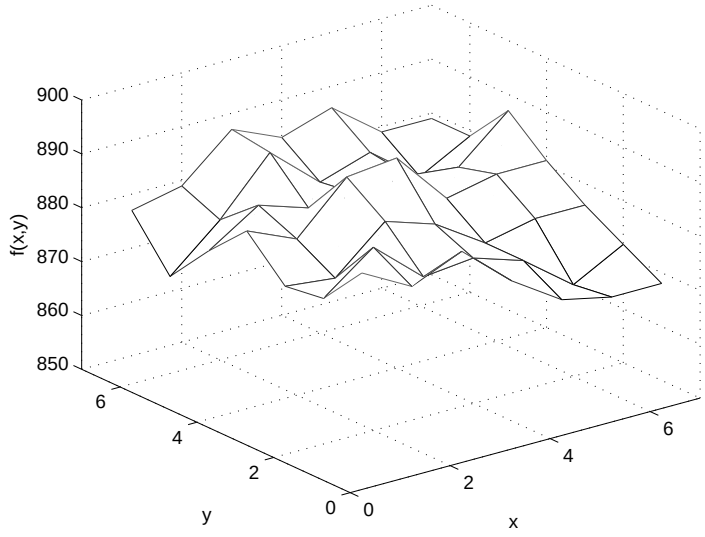


Figure 6.2: an instance of planar DLC generated by Algorithm 3

the expected runtime is exponential of $|E(G)| - |E(T)|$. For planar DLC, since there are $(n-1)^2 = \Theta(|V(G)|)$ non-spanning-tree edges in total, this sampler is impractical for its intractability. Empirically, this sampler even fails to generate an instance of planar DLC with mere 10^2 vertexes.

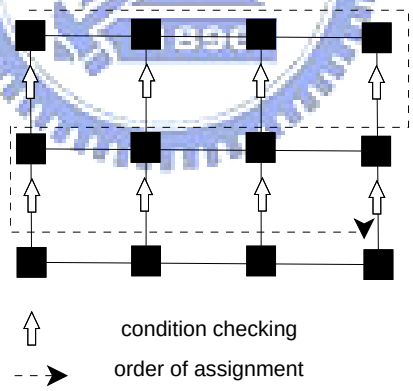


Figure 6.3: the order of assignment and condition checking of a planar DLC sampler

Furthermore, for planar DLC, there exists a Hamiltonian path in G. Therefore, We can arrange the order of vertexes such that the assignment of objective values and condition checking can be done in the same pass, and each vertex only needs to be checked once, as shown in Figure 6.3. That is to say, the generation of planar DLC can be transformed into the generation of PDLC on a Hamiltonian path provided with some overheads of inspecting. Such a result suggests that the number of problems in planar DLC is signifi-

cantly less than that of PDLC. In consequence, to generate planar DLC via the sampler of PDLC appended with further sieving mechanism is infeasible.

Similar to Section 5.2, we examine the performances of subthreshold-seekers and random search, as shown in Table 6.1 and Table 6.2. The sample size for estimating the algorithmic threshold is 6 for $|\mathcal{X}| = 6^2$ and 10 for $|\mathcal{X}| = 7^2$. In spite of the diminutive search space, the two subthreshold-seekers still outperform random search in above 80% instances sampled.

Table 6.1: Experimental results on planar DLC with $|\mathcal{X}| = 6^2$,$K = 10$

algorithm	category		
	$>$	$> .2$	time
STS, $\theta = \hat{\gamma}$	0.8190(1,679)	0.3346(686)	14.70
STS, $\theta = \hat{\mu} - \hat{\sigma}$	0.8015(1,643)	0.3312(679)	14.77
random search			17.02

$\hat{\gamma}$: estimated median. $\hat{\mu}$: estimated mean. $\hat{\sigma}$: estimated standard deviation.

Table 6.2: Experimental results on planar DLC with $|\mathcal{X}| = 7^2$,$K = 10$

algorithm	category		
	$>$	$> .2$	time
STS, $\theta = \hat{\gamma}$	0.8478(1,738)	0.3746(768)	19.74
STS, $\theta = \hat{\mu} - \hat{\sigma}$	0.8415(1,725)	0.3956(811)	19.57
random search			23.25

$\hat{\gamma}$: estimated median. $\hat{\mu}$: estimated mean. $\hat{\sigma}$: estimated standard deviation.

6.2 MCMC Sampler

In this section, we turn to the MCMC(Markov chain Monte Carlo) method for sampling. The MCMC method is to design a Markov chain with its stationary distribution equal to the desired distribution [18][15], e.g. uniform distribution in our case.

Algorithm 4 (MCMC Planar DLC Sampler)**.**

procedure MCMC Planar DLC Sampler*(G, $\mathcal{Y} = \{0, 1, \ldots, m\}, K$)*

Initialize f as an arbitrary function $f_0 \in \mathcal{L}(G, \mathcal{Y}, K)$

while** the stopping criterion is not attained **do

u.a.r. select $v \in V(G)$

$Increment \leftarrow Uniform([-K, K])$

if $f(v) + Increment \in \mathcal{Y}$ ***then***

if $\forall v' \in N(v)$, $|f(v) + Increment - f(v')| \leq K$ ***then***

$f(v) \leftarrow f(v) + Increment$

end if

end if

end while

Return f

end procedure

Similar to Algorithm 3, this sampler is applicable to all DLC. This algorithm is conceived as a Markov chain whose each state is an instance of planar DLC. In other words, it defines a random walk on $\mathcal{L}(G, \mathcal{Y}, K)$. As time goes to infinity, each state is equally likely to occur, and hence the uniformity is achieved. The following theorem shows that the stationary distribution of this chain is uniform.

Theorem 7. *The stationary distribution of the Markov chain defined by Algorithm 4 is uniform.*

Proof. For all $f_i \in \mathcal{L}(G, \mathcal{Y}, K)$, we define

$$N(f_i) := \{f \in \mathcal{L}(G, \mathcal{Y}, K) \,|\, \exists v \in V(G) \text{ such that } |f(v) - f_i(v)| \leq K \text{ and } \forall v' \neq v \ f(v') = f_i(v')\}.$$

For $f_i, f_j \in \mathcal{L}(G, \mathcal{Y}, K)$, the transition probability P_{f_i, f_j} is

$$P_{f_i,f_j} = \begin{cases} 1/(2Kn^2) & \text{if } f_j \in N(f_i)\ ; \\ 0 & \text{if } f_j \notin N(f_i)\ ; \\ 1 - |N(f_i)|/(2Kn^2) & \text{if } f_i = f_j\ . \end{cases}$$

Since this chain is irreducible and aperiodic, from Lemma 10.7 of [18], this theorem is proved. □

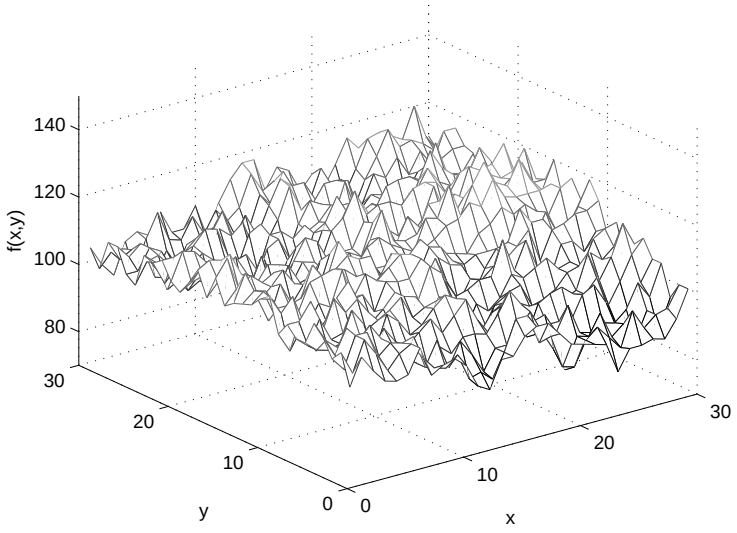


Figure 6.4: An instance of planar DLC generated by Algorithm 4

Nevertheless, Theorem 7 only guarantees that this Markov chain will eventually converge to uniform distribution, yet it is doubtful that this chain is rapidly mixing when K is much smaller than $|V(G)|$ and $|\mathcal{Y}|$. Figure 6.4 demonstrates an instance generated by Algorithm 4 with $|V(G)| = 30^2$, $K = 10$ and the maximum number of iterations 10^9, and it can be observed that the landscape of the solution space is quite level.

6.3 Semi-planar DLC

As demonstrated previously, the practicability of both Algorithm 3 and 4 is questionable for large $|V(G)|$. Also, the arguments in Section 6.1 suggest that no matter how a sampler assigns values to vertexes, it inevitably fails if the number of edges needing to be checked is enormous. Hence, in order to generate instances with sufficient vertexes, we will loose the restriction of Lipschitz condition slightly to avoid edge checking, so the DLC instances generated here are semi-planar in the sense that they belong to the DLC defined on a spanning tree of G, rather than G itself. More specifically, we will discard the edge-checking mechanism in Algorithm 3 and prefix to it a uniform spanning-tree generater. In other words, the sampler firstly generates a spanning tree u.a.r. and thereafter assign values to vertexes according to it.

The following question will be what the targeted population is. It is obvious to see

that the modified sampler uniformly generates the tuple (T, f), where T is a spanning tree of G and $f \in \mathcal{L}(T, \mathcal{Y}, K)$. However, this sampler is not uniform with respect to f because of its two-phase nature.

As to the issue of generating random spanning trees, Broder[19] proposed the following algorithm and proved that it is a uniform spanning-tree generater.

Algorithm 5 (Uniform Spanning-Tree Sampler)**.**

procedure UNIFORM SPANNING-TREE SAMPLER*(A connected graph G)*

Initialize T as $E(T) = \emptyset$ and $V(T) = V(G)$

Select v_0 u.a.r. from $V(G)$

Start a random walk from v_0 on G

while** there exists a vertex has not been visited **do

Continue the random walk

if** $v \in V(G)$ is firstly visited via the edge $\overline{uv}$ **then

Add $\overline{uv}$ to $E(T)$

end if

end while

Return T

end procedure

Clearly, the expected runtime of this algorithm is equal to the expected cover time of simple random walks on G, the time which is $O(|V(G)||E(G)|)$ [20], and this complexity can be further reduced in most cases([21],[22],[23]). For planar DLC, G is a sparse graph, so the expected runtime of Algorithm 5 is $O(|V(G)|^2)$. Based on this algorithm, the following generater is presented.

Algorithm 6 (Semi-Planar DLC Generater)**.**

procedure SEMI-PLANAR DLC GENERATER*(G, $\mathcal{Y} = \{0, 1, \ldots, m\}$,$K$)*

Select v_0 u.a.r. from $V(G)$

$f(v_0) \leftarrow Uniform([0, m])$

Start a random walk from v_0 on G

while** there exists a vertex has not been visited **do

Continue the random walk

if $v \in V(G)$ *is firstly visited via the edge* $\overline{uv}$ ***then***

$f(v) \leftarrow f(u) + Uniform([-K, K])$

if $f(v) \notin \mathcal{Y}$ ***then***

Reject and restart

end if

end if

end while

Return f

end procedure

Figure 6.5 displays an instance generated by Algorithm 6. As shown in the figure, there exist some steep slopes in the search space, for the values of the neighbors corresponding to those non-spanning-tree edges are not necessarily bounded by K.

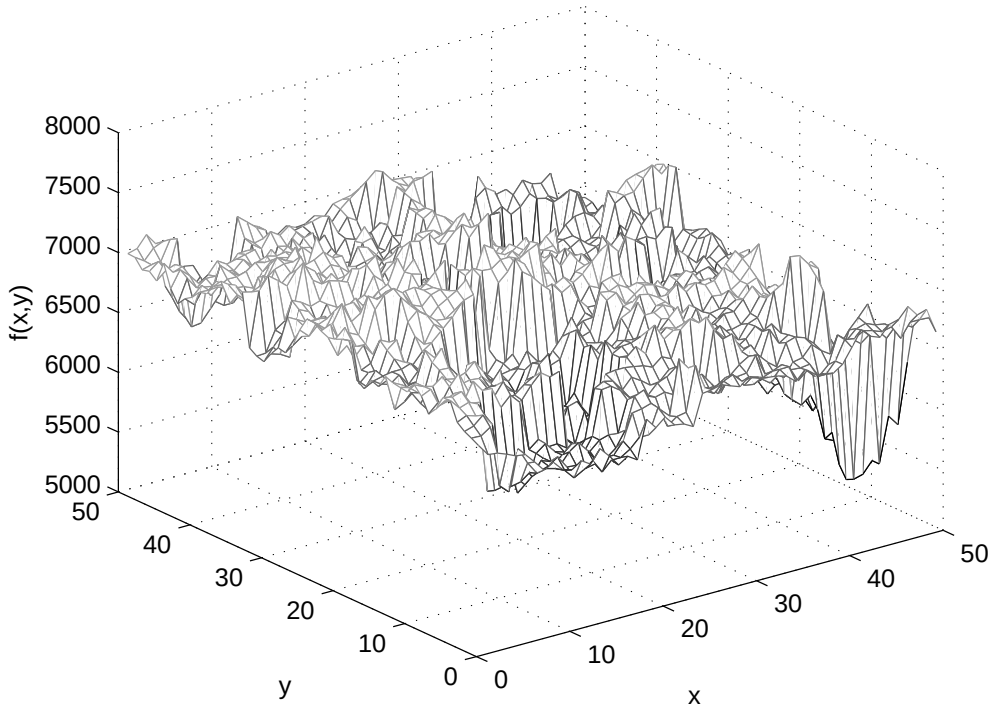


Figure 6.5: An instance generated by Algorithm 6

Although not belonging to genuine planar DLC, these problem instances may still render us some insights into DLC. Therefore, a corresponding experiment on a suite of

Table 6.3: Experimental results on semi-planar DLC with $|\mathcal{X}| = 100^2$,$K = 100$

algorithm	category		
	$>$	$> .2$	time
STS, $\theta = \hat{\gamma}$	0.9932(2,036)	0.9405(1,928)	2670.82
STS, $\theta = \hat{\mu} - \hat{\sigma}$	1.0000(2,050)	1.0000(2,050)	962.56
random search			4973.94

$\hat{\gamma}$: estimated median. $\hat{\mu}$: estimated mean. $\hat{\sigma}$: estimated standard deviation.

2,050 problems is conducted likewise, and the result is shown in Table 6.3. It can be seen that the result coincide with the case of PDLC.

Even though the limitation of sampling techniques prohibits us from examining planar DLC thoroughly, the results in both Section 6.1 and 6.3 suggest that the advantage of subthreshold-seeker over random search on DLC could be carried over into higher dimensions.

Chapter 7

Conclusions

7.1 Summary

In this study, we introduced and investigated the properties of the discrete Lipschitz class. A generalized subthreshold-seeker was then proposed and shown to outperform random search on this broad function class. Finally, we proposed a tractable sampling-test scheme to empirically demonstrate the performance of the generalized subthreshold-seeker under practical configurations. We showed that optimization algorithms outperforming random search on the discrete Lipschitz class do exist from both theoretical and practical aspects.

7.2 Main Conclusions

As controversial as it may be, the NFL theorem provides an alternative standpoint to review the position of optimization algorithms and search heuristics. The NFL theorem expels the false hope to conquer all possible functions with only limited information available, as it points out the expectation to find a universally black-box optimizer is definitely over-optimistic. However, the NFL theorem does not imply the utter infertility in the land of search heuristics by any means, if our goals are appropriately placed. In this thesis, the discrete Lipschitz class, as a simulation of continuous functions in a discrete space, is shown to be a class of problems on which black-box optimizers have performance advantages in both theory and practice. The only constraint imposed on the search space is bounded differences within a neighborhood. Under such a minor condition, black-box optimizers can still be effective over a broad, meaningful, and practical problem class as suggested by this study.

7.3 Future Work

The issue of sampling is a major field that needs further researching. To generate PDLC instances with a moderate Lipschitz constant, i.e. $O(|\mathcal{Y}|^{1/2})$, the sampling algorithm proposed in Chapter 5 renders us a simple and effective means. As for extremely large K, we can generate unbounded instances at first and accept those which are legitimate. Nevertheless, those in between seem unable to be generated by simple accept-reject algorithms. Therefore, the search for competent samplers for various sizes of K is an intriguing area worth investigating. This situation also holds with regard to planar DLC, for the rejection rate of sampling planar DLC is significantly higher than that of PDLC. The relaxation of absolute uniformity, that is to seek a fully polynomial almost uniform sampler(FPAUS)[24], might be a possible route to bypass the difficulties. Once this goal had been achieved, the effect of the magnitude of K could be subsequently analyzed via simulations.

Another possible objective, which may be more attainable, is to explore the relationships between DLC and those optimization algorithms in practice in real life. Without the restriction of the NFL framework, the sampling-test scheme of DLC can serve as a benchmark for testing and comparing optimization algorithms, so it might supply an alternative point of view and, hopefully, a paradigm of examination into the applicability of search heuristics.

Bibliography

[1] D. H. Wolpert, W. G. Macready, No free lunch theorems for search, Tech. Rep. SFI-TR-95-02-010, Santa Fe Institute (1995).

[2] D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 4 (1997) 67–82.

[3] J. C. Culberson, On the futility of blind search: An algorithmic view of "no free lunch", Evolutionary Computation 6 (1998) 109–127.

[4] S. Droste, T. Jansen, I. Wegener, Perhaps not a free lunch but at least a free appetizer, Tech. Rep. ISSN 1433-3325, Department of Computer Science, University of Dortmund (1998).

[5] S. Droste, T. Jansen, I. Wegener, Optimization with randomized search heuristics – the (a)nfl theorem, realistic scenarios, and difficult functions, Theoretical Computer Science 287 (2002) 131–144.

[6] M. J. Streeter, Two broad classes of functions for which a no free lunch result does not hold, in: Proceedings of the Genetic and Evolutionary Computation Conference 2003, 2003, pp. 1418–1430.

[7] S. Christensen, F. Oppacher, What can we learn from no free lunch? a first attempt to characterize the concept of a searchable function, in: Proceedings of the Genetic and Evolutionary Computation Conference 2001, 2001, pp. 1219–1226.

[8] D. Whitley, J. Rowe, Subthreshold-seeking local search, Theoretical Computer Science 361 (2006) 2–17.

[9] C. Schumacher, M. D. Vose, L. D. Whitley, The no free lunch and problem description length, in: Proceedings of the Genetic and Evolutionary Computation Conference 2001, 2001, pp. 565–570.

[10] R. Courant, F. John, Introduction to Calculus and Analysis, Vol. 1, Vol. 1, Springer-Verlag, 1989.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to Algorithms, 2nd Edition, The MIT Press, 2001.

[12] R. Motwani, P. Raghavan, Randomized Algorithms, Cambridge University Press, 1995.

[13] I. Rechenberg, Evolutionsstrategie '94, Frommann Holzboog, 1994.

[14] J. Jägersküpper, Algorithmic analysis of a basic evolutionary algorithm for continuous optimization, Theoretical Computer Science 379 (3) (2007) 329–347.

[15] C. Robert, G. Casella, Monte Carlo Statistical Methods, Springer-Verlag, 1999.

[16] Y. S. Chow, H. Teicher, Probability theory: independence, interchangeability, martingales, 3rd Edition, Springer, 1997.

[17] W. Hoeffding, Probability inequalities for sums of bounded random variables, Journal of the American Statistical Association 58 (1963) 13–30.

[18] M. Mitzenmacher, E. Upfal, Probability and Computing: Randomized Algorithms and Probabilistic Analysis, Cambridge University Press, 2005.

[19] A. Broder, Generating random spanning trees, in: Foundations of Computer Science 1989, 1989, pp. 442–447.

[20] R. Aleliunas, R. Karp, R. Lipton, L. Lovasz, C. Rackoff, Random walks, universal traversal sequences, and the complexity of maze problems, in: Foundations of Computer Science 1979, 1979, pp. 218–233.

[21] A. Broder, E. Shamir, On the second eigenvalue of random regular graphs, in: Foundations of Computer Science 1987, 1987, pp. 286–294.

[22] A. Broder, A. Karlin, Bounds on cover times, Journal of Theoretical Probability 2 (1989) 101–120.

[23] Z. Füredi, J. Komlós, The eigenvalues of random symmetric matrices, Combinatorica 1 (1981) 233–241.

[24] R. Y. Rubinstein, D. P. Kroese, Simulation and the Monte Carlo Method, 2nd Edition, Wiley, 2007.