eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

Konputazio Zientzia eta Adimen Artifizialaren Saila

Departamento de Ciencia de la Computación e Inteligencia Artificial

Department of Computer Science and Artificial Intelligence

Contributions to the Mathematical Modeling of Estimation of Distribution Algorithms and Pseudo-Boolean Functions

by

Imanol Unanue Gual

Supervised by María Merino Maestre and Jose A. Lozano

Donostia - San Sebastián, July 2023

"If you can't solve a problem,
then there is an easier problem you can solve:
find it."
— George Pólya

Agradecimientos

Primero de todo, quiero agradecer al Ministerio de Educación del Gobierno Vasco por todas las ayudas económicas y administrativas concedidas para la realización de este trabajo. Los códigos de los expedientes que me han financiado son los siguientes: PRE_2018_1_0267, PRE_2019_2_0266, PRE_2020_2_0166 y PRE_2021_2_0244. Agradezco también a mi supervisor y director del BCAM - Basque Center for Applied Mathematics, Jose A. Lozano, por permitirme hacer unas prácticas en el centro. Por último, agradezco a todos los organismos que me han ayudado de forma indirecta en la realización de este trabajo.

La realización de esta tesis ha sido, sin ninguna duda, una de las experiencias que más me han marcado en mi vida, pero afortunadamente, gracias al apoyo de algunos compañeros, amigos y seres queridos, he tenido la fortuna de poder concluir este trabajo. Quiero primero agradecer a todos ellos y especialmente a Borja, al *Fondo Norte*, a Iker, a Ivan, a Jon, a Luken y a Marina.

Además de ellos, quiero dedicar dos párrafos a las tres personas que han sido fundamentales para poder dar fin a este trabajo. Primero, a mi supervisora, tutora y guía, tanto académica como laboral, María Merino Maestre. Me es muy difícil describir todo lo que ha hecho por mí, pero sin duda ha sido la piedra angular del trabajo: se interesó en él, en mi situación familiar y sobretodo en mí; me proponía nuevas investigaciones, nuevos experimentos, mejoras en el trabajo, etc. Por todo ello, *eskerrik asko niregatik eta niretzat egin duzun guztiagatik*.

Para acabar, este último párrafo se lo dedico a mis padres. Ante todas las adversidades y dificultades que se me presentaban y que me empujaban a rendirme, nunca os apartasteis de mí. Estuvisteis en mis momentos emocionales más duros y me ayudasteis a no verlo todo tan negro. Citando una de mis canciones favoritas, "*I owe it to you, (...) for keeping my head up*". Por todo ello, quiero que los lectores de este agradecimiento sepan lo que hicisteis por mí y, por mi parte, acabar con mis palabras más importantes de toda la tesis: siento muchísimo todo lo que os he hecho pasar.

Contents

1

Introduction

1.1 Optimization Problems

Optimization is an area of research which intersects computer science, applied mathematics and operations research. Optimization problems are a set of problems in which the goal is to obtain a feasible solution which maximizes or minimizes a fitness function (also named objective function or utility function). Throughout this thesis, without loss of generality, we will assume maximization problems. Mathematically, a fitness function f is formally described as a function:

$$\begin{aligned} f : \Omega &\to \mathbb{R} \\ x &\rightsquigarrow f(x) \end{aligned}, \tag{1.1}$$

where Ω is the search space (the space of solutions), $x \in \Omega$ is a possible solution and $f(x)$ is the fitness function value of the solution x, which is a real number.

Fitness functions can describe set problems (where the solutions describe sets of objects), graph problems (where, according to a graph, the solutions classify edges or vertexes) or numerical problems (where the solutions are sets of numbers), among others. In addition, even if the solutions of two or more problems are equally described, their meaning might be completely different.

An optimization problem is formally described in the following way:

$$\begin{aligned} &\arg\max_{x \in \Omega} f(x) \\ &\text{subject to} \quad g_i(x) \leq b_i \ , i \in \{1, \ldots, i_{\max}\} \text{ and } b_i \in \mathbb{R}, \end{aligned} \tag{1.2}$$

where the inequalities $g_i(x) \leq b_i$ are *the constraints*, i.e., they define the feasible region. Bear in mind that, without loss of generality, any other inequality or equality (such as $g_i(x) \geq b_i$ or $g_i(x) = b_i$) can be redefined and rewritten as one or two constraints $g_i(x) \leq b_i$ because of the additive inverse property. For any optimization problem, the functions f and g_i could be defined by means of several parameters. An *instance* of the optimization problem is defined by specifying all the parameter values of the functions f and g_i and the values b_i. If an optimization problem is defined without any constraint ($i_{\max} = 0$), then it is called an *unconstrained problem* and all the solutions of the search space are feasible solutions; otherwise, it is called a *constrained problem*. A feasible solution with the highest fitness function value is an *optimal solution*. When

f has only one optimal solution, then the fitness function is unimodal; otherwise, the fitness function is multimodal.

According to the search space, optimization problems can be classified in two types of problems: continuous optimization problems [82] and Combinatorial Optimization Problems (COPs) [3]. The most common search space for continuous optimization problems is $\Omega = \mathbb{R}^{+n}$. On the other hand, COPs are characterized by having infinite numerable or discrete search spaces. In the infinite case, the usual search space is $\Omega \subseteq \mathbb{Z}^{+n}$, whereas in the finite case, the usual search spaces are binary strings of length n and the permutation space of the set $\{1, \ldots, n\}$. In the finite cases, the former group defines *binary-based COPs* and the latter group generates *permutation-based COPs*.

There are many examples of COPs. Some of the most well-known COPs in the fields of computer science, applied mathematics and operations research are Sorting Problems, Integer Programming Problems, the Maximum Satisfiability Problem, the Unconstrained Binary Quadratic Problem, the Chromatic Number Problem, Assignment Problems, the Minimum Spanning Tree Problem, Knapsack Problems, the Traveling Salesman Problem, The Flowshop Scheduling Problem and the Facility Location Problem. Furthermore, in other fields, such as physics and economics, some of the studied problems are equivalent optimization problems described with a different explanation and/or notation. For example, in physics, the simplest and the most known Ising Model is described by a Hamiltonian function, which is equivalent to the study of the Maximum Cut Problem, a well-known problem in computer science and operations research.

Let us briefly explain some examples of optimization problems to show the variety of definitions, representations and fitness functions. In the Integer Linear Programming (the canonical form), the objective is to find an integer vector that maximizes a linear fitness function where the region of feasible solutions is a convex region described by a set of linear inequalities. In this scenario, n is the number of dimensions to describe the region and a solution is defined as $x = (x_1, \ldots, x_n)$ (it being a point of the feasible region). In the Maximum Satisfiability Problem, n is the number of binary variables. The objective is to assign to each binary variable a TRUE-FALSE value in order to satisfy the maximum number of clauses C_k (subsets of binary variables, assertion or negation, combined by logical operators AND and OR) expressed in the fitness function $f = \sum_k C_k$. In the Knapsack Problem, there is a set of objects $\{i_1, \ldots, i_n\}$, determined by their weight $\{w_1, \ldots, w_n\}$ and their utility $\{u_1, \ldots, u_n\}$. The objective is to select a subset of objects $S \subseteq \{i_1, \ldots, i_n\}$ such that the sum of their weights does not exceed the capacity of the knapsack ($\sum_{j \in S} w_j \leq W$) and the sum of their utility values is maximum ($\max \sum_{j \in S} u_j$). The solutions of the problem can also be redescribed by binary strings of length n (the number of different objects), in which each bit x_i describes if the object i is included in the knapsack or not. In the Traveling Salesman Problem, the objective is to give the shortest path (in terms of minimum distance, time or cost) to visit all the given cities once starting and ending at the same point. In this problem, a solution is denoted by a permutation of length n (the number of cities), where the particular ordering of the permutation determines in which order the cities have to be visited. In the Unconstrained Binary Quadratic Problem, the goal is to maximize a quadratic fitness function by a suitable choice of the binary variables. The solutions are described by binary strings of length n. This last particular problem is formally described in Chapter 2.

1.2 How to solve COPs

It is obvious that any COP can be solved with an exhaustive analysis of all the feasible solutions, computing all the fitness function values and comparing them to select the optimal solution ("brute-force search").

Unfortunately, this strategy is not efficient when the size of the problem (or, equivalently, the size of the search space) is large and/or the fitness function is computationally expensive to evaluate. It is well known that the exhaustive analysis is not always viable, see for example the book "Computers and Intractability: A Guide to the Theory of NP-Completeness" by Garey and Johnson [42]. In the first chapter of the book, the authors present several initial definitions about problems, algorithms and time complexity. Considering these initial definitions, the authors show an example of the execution time of polynomial and exponential time complexity algorithms according to the size of the problem. It is clearly proved that, for the first case, when the size of the problem increases, the required time is still tractable, whereas for the second case it is not. Let us present a simple example of the latter case. For any instance of the Traveling Salesman Problem with $n+1$ cities and one of which is the starting point, the size of the space of solutions is $n!$. Let us assume that evaluating a solution requires 1 millisecond. Therefore, when $n = 10$, the required time to evaluate all the solutions is one hour approximately; however, when $n = 15$, the required time to evaluate all the solutions is more than 40 years!

The difference between the required computational time (and/or memory) to solve COPs classifies them in two groups: *P* (polynomial) problems and *NP* (non-deterministic polynomial) problems. In addition, the algorithms that solve optimization problems follow the same classification (evaluating the worst possible scenario of the algorithm in terms of computational cost): polynomial time algorithms and non-deterministic polynomial time algorithms. This classification of the problems still gains more relevance because of the well-known *P vs NP problem*, one of the 7 Millennium Prize Problems selected by the Clay Mathematical Institute and a major unsolved problem in the field of Theoretical Computer Science. Informally speaking, the problem consists of determining whether or not all the optimization problems that can be verified in polynomial time can be solved by a polynomial time algorithm. Currently, there is no known polynomial algorithm to solve *NP* problems (equivalently, a polynomial time algorithm which solves all the instances of a *NP* problem). Because of that, in order to solve optimization problems efficiently, most of the research has been carried out in the design of exact, heuristic and metaheuristic algorithms [2, 62, 63, 69, 89].

Exact algorithms [41] theoretically always achieve an optimal solution of the studied problem, even if the required computational cost (in terms of time or memory) for at least one instance is exponential. The interest of this kind of algorithms is not only to obtain the optimal solution, but they are also useful to understand the problem in essence and how to solve them efficiently. Two very well-known examples of this kind of algorithms to solve optimization problems are the simplex algorithm, which has polynomial time average-case complexity to solve the Integer Linear Programming, and the dynamic programming algorithm, which solves the Knapsack Problem in polynomial time with respect to the number of objects but the required memory is exponential. On the other hand, heuristic and metaheuristic algorithms [44] obtain high quality solutions in a reasonable computational time, but they do not guarantee that the optimal solution will be achieved. The difference between heuristic and metaheuristic algorithms is that, whereas heuristic algorithms are mostly designed with respect to the definition of the studied problem, metaheuristic algorithms are higher-level procedures: they do not depend on the definition of the problems and they guide subordinate heuristics for exploring and exploiting the search space [12, 43, 88, 101]. Besides, metaheuristic algorithms are capable of tackling problems whose fitness function is not a mathematical close expression.

However, a disadvantage about heuristic and metaheuristic algorithms is that, for any instance of a problem, the algorithm can behave differently in two different and independent runs. This case can be obtained when a step of the algorithm depends on a probability $p \in (0, 1)$, for example. So, to evaluate the performance of an algorithm, it is necessary to repeat runs of the algorithm with the same instance (when the algorithm can return different solutions), to select an appropriate set of instances and to compare it with other possible algorithms to solve the selected instances. Mainly, in the literature, researchers compare their proposal with

several state-of-the-art algorithms to solve a particular problem and they use a specific metric to evaluate and compare their performance. One of the main critical steps of this process is to choose an appropriate set of instances of the analyzed problem to make a fair comparison. In other words, the set of instances must be representative of all the possible scenarios that the problem can generate and the algorithms cannot use any information about the instances in advance. There are two types of instances: real-world instances and artificial instances.

Metaheuristic algorithms can be classified in several ways according to specific criteria. From all the possible classifications, we will highlight the following classification in three groups: constructive methods, local-based algorithms and population-based algorithms. In constructive methods, the algorithm generates a solution by the addition and union of the components of a solution. Two well-known constructive methods are Kruskal's algorithm [67] and Prim's algorithm [93]. Secondly, local-based algorithms always consider one solution. At each step, the algorithms study the neighboring solutions and select a solution that improves the considered one. The algorithm ends when all the neighboring solutions are worse than the considered one. A well-known local-based algorithm is Greedy Randomized Adaptive Search Procedure (GRASP) [38]. Lastly, in a population-based algorithm, a population (a set or a multiset of solutions) is used to generate a new population. The objective is to get better solutions (to improve fitness function) when the number of iterations increases. Among population-based algorithms, the most common are Evolutionary Algorithms [5, 28, 33, 47, 57].

1.3 Evolutionary Algorithms

Evolutionary Algorithms are a set of algorithms based on Darwin's theory of evolution. The theory describes the evolution and adaptation of the species to the environment according to the principle of natural selection, favoring the best adapted species. This phenomenon is summarized as "survival of the fittest", by Herbert Spencer. In addition, another factor is the occurrence of small, apparently random and undirected variations between the manner of response and physical embodiment of parents and their children (denoted as "mutation"). Through these variations, new combinations of characteristics occur and are evaluated. The best individuals survive and reproduce new individuals with their best features with respect to the environment (denoted as "crossover"), whereas the worst individuals perish. The same theory can be translated to programming and creating new algorithms to solve COPs. In practice, it has been shown that this kind of algorithms are widely applicable and they perform efficiently. Some of the subareas of Evolutionary Algorithms are Evolutionary Programming [40], Evolution Strategies [6, 10], Genetic Programming [64] and, the most popular subarea, Genetic Algorithms [57, 110].

Genetic Algorithms were initially designed to solve COPs, and later they were redesigned for continuous problems. Let us explain an iteration of a generic Genetic Algorithm. Starting from a population, first the algorithm selects a subset of solutions from the population. The selection procedure is defined by a selection operator. Secondly, the algorithm takes two or more selected solutions (*the parents*) and they are combined to generate new solutions (*the children*) which define a new population. The crossover operator defines how the children are generated. The goal is to keep the best features of the parents. Finally, the mutation operator can modify the children randomly (depending on a probability). The operator is defined by primitive functions such as conditional logical operators and/or mathematical functions. In Algorithm 1 the general pseudocode of a generic Genetic Algorithm is introduced.

Algorithm 1 General pseudocode of a generic Genetic Algorithm

```
Obtain an initial population D_0
while Stop criteria = FALSE do
  Select a subset of individuals from the population D_i: D_i^S
  Apply the crossover operator to D_i^S: D_{i+1/3}
  Apply the mutation operator to D_{i+1/3}: D_{i+2/3}
  Generate a new population D_{i+1} with D_i, D_{i+1/3} and D_{i+2/3}
  i = i + 1
end while
Return Best individual of the final population
```

Some well-known selection operators are n-tournament selection, proportional selection and truncation selection [11, 119]. The n-tournament selection considers n solutions and takes the best solution to form the subset of individuals of the population. The proportional selection generates a probability distribution based on the fitness function value of each individual of the population, and that distribution generates the selected population. The truncation selection generates a number of solutions greater than the population size N and chooses the best N solutions. On the other hand, the most used crossover operator is the m-point crossover, in which a solution is created by combining $m + 1$ disjoint parts of its parents.

1.4 Estimation of Distribution Algorithms

From all the subareas of Evolutionary Algorithms, there is an intriguing group which has gained relevance in the last few years: Estimation of Distribution Algorithms (EDAs) [70]. EDAs, also named Probabilistic Model-Building Genetic Algorithms, were introduced for the first time in the field of Evolutionary Computation by Mühlenbein and Paaβ [85]. The main characteristic of EDAs with respect to generic Evolutionary Algorithms is the use of probability distributions instead of the usual natural evolution operators, such as recombination and mutation. In this way, EDAs start with a population D, in most cases by means of sampling a uniform probability distribution over the search space. There are several ways to study EDAs, depending on how an iteration of the algorithm is described. The most common explanation of a step of an EDA is the following one. From the population D_i (where i indicates the iteration of the algorithm), EDAs use a selection operator and obtain a subset of solutions D_i^S which is used to learn a probability distribution P_i^L. This distribution can be learnt from scratch or by modifying the probability distribution used to sample the population at the previous iteration (such as in Compact Genetic Algorithm (cGA) [51]). The ideal goals of the learned probability distribution are to summarize the main features of the selected solutions and to highlight the best solutions. Finally, the learned probability distribution is sampled to obtain a new set of solutions and to generate a new population D_{i+1}, which is used at the next iteration of the algorithm. In Algorithm 2 the general pseudocode of an EDA which learns a probability distribution from scratch at each iteration is introduced.

EDAs have been and are being designed, applied and analyzed in the solution of COPs. They have been mostly designed and studied for binary-based COPs. Some examples of the designed EDAs for binary-based COPs are Univariate Marginal Distribution Algorithm (UMDA) [85], Population-based Incremental Learning (PBIL) [7] and Factorized Distribution Algorithm (FDA) [84]. Moreover, they have also been complemented

Algorithm 2 General pseudocode of an EDA

Obtain an initial population D_0
while Stop criteria = FALSE **do**
  Select a subset of individuals from the population D_i: D_i^S
  Learn a probability distribution from D_i^S: P_i^L
  Sample a new set of individuals using P_i^L: $D_{i+\frac{1}{2}}$
  Generate a new population D_{i+1} with D_i and $D_{i+\frac{1}{2}}$
  $i = i + 1$
end while
Return Best individual of the final population

with a theoretical study with the purpose of understanding and improving these algorithms [48, 49, 84]. The first theoretical studies focused on the convergence behavior of the algorithm and the first studied algorithms were UMDA [77, 118, 119] and PBIL [48, 49, 56]. Nonetheless, several works have been presented recently in the literature with the aim of attaining new results regarding the runtime [26, 27, 65, 71, 72, 111, 112, 115], the population sizing [83, 91, 117] or the model accuracy of EDAs [30]. We highly recommend the work [66] for a state-of-the-art on binary EDAs.

From [66], we want to highlight three inspiring works that have been considered to present our results. In [48], the authors prove that when the fitness function is unimodal, PBIL converges to the global optimum. In [49], it is proved that any discrete EDA generates a population with an optimal solution if any solution of the search space can be generated at any iteration of the algorithm. In addition, in the same work, the authors review a dynamical system used in the literature to study UMDA and PBIL. In the present work, we have considered the idea of studying EDAs as dynamical systems. Last but not least, in [84], the authors study the convergence behavior of the FDA using Boltzmann and truncation selection and by analyzing finite and infinite populations, which shows the influence of the assumption of infinite populations and the differences in the obtained results.

Current theoretical research of binary EDAs is often based on runtime analysis. The goal is to find bounds on the number of generations to sample a high quality or optimal solution for the first time. This goal has a close connection with the practical use of the algorithms, where we would like to sample an optimal solution as soon as possible. Notice that an optimal solution can be reached for an algorithm without requiring convergence to it [113].

1.5 Which heuristic is the most appropriate to solve a COP?

It is observed that there exist many exact and (meta)heuristic algorithms to solve COPs. But the questions are: 1) is there an algorithm that outperforms any other algorithm at any COP problem/instance? and 2) why researchers develop new designs of exact and (meta)heuristic algorithms? There exists an article whose results answer both questions: “No Free Lunch Theorems for Optimization” [114]. The main result states that any two algorithms designed to solve COPs will perform, on average, equally well over all optimization problem instances. In addition, any algorithm performs, on average, as well as a random search. Therefore, the performance of an algorithm and the comparison among algorithms is completely dependent on the evaluated problem instance. This observation leads to a new intriguing question: having a particular COP,

which is the algorithm that will perform better? The idilic goal is to present an association that, given a COP (instance), the "oracle" function returns the most efficient algorithm to solve it [97]. This scenario is known in the literature as the Algorithm Selection Problem [96, 97] and the resolution of it would significantly reduce the cost of solving problems. To do so, the association considers the definition of the COP, the difficulty of the problem instances and the design of the algorithms. The study of the relations between COPs and algorithms is the primary study in the area of the algorithm selection (which is closely related to the field of computational complexity).

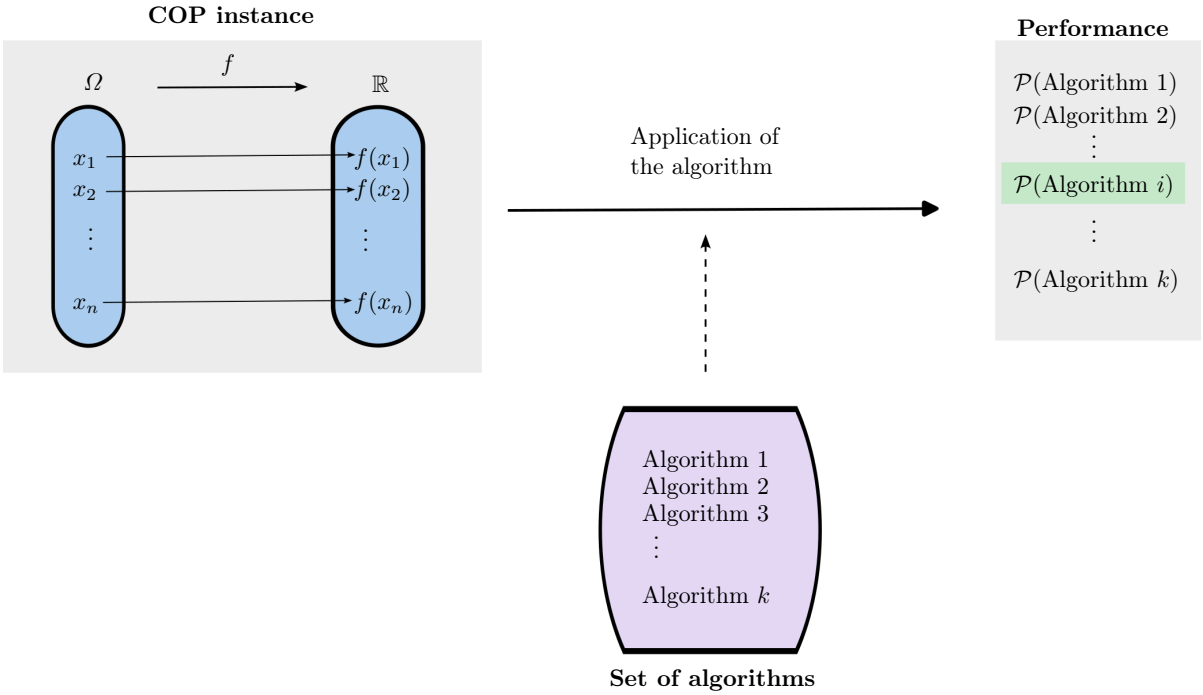


Fig. 1.1: Schema of the algorithm selection to solve a COP instance.

This objective can be addressed by two main theoretical research lines, depending on the focus, and a final step of joining the obtained results. The two main research lines are: to study the design and characteristics of the algorithms and to study the definition of the COP and the range of possible instances generated by the problem. The first line focuses on analyzing the algorithms and obtaining results which can be used for its practical application (as previously said, theoretical studies of the algorithms). The second line focuses on the definition, characteristics and equivalences among problems and problem instances. However, grouping problems or instances is a difficult task as they come defined in a variety of different forms. For example, the definition of the Unconstrained Binary Quadratic Problem is not apparently related with the definition of the well-known Knapsack Problem (even if the solutions of both problems are described by binary strings). Several mathematical tools presented in the literature to confront this problem are: equivalences and taxonomization of COPs (and/or COP instances) [35, 75], generation of surrogate models [73, 106, 116], study of linkage among the variables [21, 52], decomposition of the fitness functions [34, 35] and classification of the problem

instances [1]. From all the mentioned research lines, we are especially intrigued by surrogated models and the decomposition of fitness functions. Both research lines share a crucial property of the study: analyze a very "related" and different fitness function. This can be done by a (re)definition, a generation and/or a simplification of the fitness function. For example, in [35], the authors use the Fourier transform and calculate the Fourier coefficients of several permutation-based combinatorial optimization problems. Therefore, their analysis diverges from the particular definition of each problem and presents a new "framework" in which a result can be extended to many problems. In this framework, the authors are able to see which instances of different problems are equivalent and which possible rankings can be generated from a specific combinatorial problem.

In this thesis, we will focus on the Walsh decomposition. However, for a better comprehension of it, first we present briefly the most well-known decomposition of continuous functions in mathematics: the Fourier analysis. The Fourier analysis allows any continuous function to be studied as a sum of simple trigonometric functions. Particularly, the most basic representation of a continuous function is to approximate it by a weighted sum of sine and cosine functions:

$$f(x) \sim a_0 + \sum_{j=1}^{\infty} a_j \cos\left(\frac{2\pi jx}{P}\right) + b_j \sin\left(\frac{2\pi jx}{P}\right), \tag{1.3}$$

where P is the periodicity of the function and a_j, b_j are the Fourier coefficients. When j tends to infinity (i.e., the approximation is described with a higher number of sine and cosine functions), the approximated function converges pointwise to f. The Fourier transform is used to simplify complex mathematical expressions, to transform differential equations or to process signals, for example.

1.6 The Walsh decomposition

The Walsh functions is a complete set of orthogonal functions introduced by Walsh in [108]. Although this set of functions was originally described for functions defined over the interval $(0, 1)$, it has been extended to decompose any discrete function, similar to the Fourier transform over the continuous functions. This decomposition process is known as the Walsh transform, Walsh-Hadamard transform or Walsh-Fourier transform.

The Walsh transform has been recently used in the field of evolutionary computation in the solution of many binary and real-world COPs. For example, it has been used to create accurate surrogate models for black-box optimization [73, 106, 116], to study the linkage discovery problem [21, 52] or to recover polynomials in which the monomials with small degrees are the most significant [68, 86], among others. In all the previous mentioned works, the authors present algorithms and methodologies to build proxies based on the Walsh functions which are accurate approximations of an original function or to use the steps of the Walsh decomposition to approximate fitness functions as polynomials in which non-null monomials have low order. For example, in the black-box optimization, the generated approximations of the fitness function have low computational cost; and in the linkage discovery problem, Walsh coefficients show the relation among the variables in a very direct way which is crucial to design new methodologies to solve the problem. In addition, each Walsh function associated to at least one variable presents a partition of the solutions of the search space. So, the analysis of the relations among the variables can be extended and can observe the meaning of the values of the Walsh coefficients. Particularly, the most intriguing Walsh coefficients are the ones with a null value.

1.7 Fitness function as ranking of solutions

The Walsh decomposition presents a bijection between sets of Walsh coefficients and pseudo-Boolean functions, i.e., functions whose solutions are codified as 0-1 vectors. Moreover, in the particular case of binary-based COPs, a very relevant result states that any fitness function of binary-based COPs can be redefined as a pseudo-Boolean function. In fact, any pseudo-Boolean function can be written uniquely as a multi-linear polynomial of degree $m \leq n$ [15, 50]. Consequently, it is intractable to analyze all the sets of Walsh coefficients.

One way to approach this diversity is to consider each fitness function as a *ranking generator* or, equivalently, a *ranking of solutions* of the search space. A ranking of solutions is an ordered list of all the solutions of the search space Ω. Throughout this thesis, we denote a ranking of solutions with the letter r. Hence, a ranking of solutions defined by a a fitness function f is an ordered list of all the solutions according to their fitness function values: the first solution is the solution with the highest fitness function value, the second solution is the second highest fitness function value, and so on. Therefore, a COP can be interpreted as the set of rankings generated by its definition. The most important feature of the rankings is the relative order of the solutions according to the fitness function, not their exact fitness function values. This makes sense as most algorithms, such as local search or evolutionary algorithms with tournament or ranking selection (to name a few), only consider the ranking of solutions in their machinery instead of the specific fitness function value of a solution. This avenue has been previously followed in works such as [54]. In the mentioned article, the authors show that the studied permutation-based COPs cannot generate all the possible rankings of solutions and they present "the intersection of COPs", i.e., instances that can be generated by several COPs. Our desired goal is to present a characterization of the rankings according to their features which allows us to select the most "appropriate" algorithm (in terms of efficiency) to solve (an instance of) a problem.

1.8 Instance generation

The study of the rankings of solutions allows us to recognize the scenarios that can and cannot be generated by the studied COP. Still, among all the possible instances of a problem, there are significant differences. So, even for a specific problem, it is necessary to evaluate the proposed algorithms in a wide range of instances for a correct evaluation. Because of the lack of real-world instances of the studied problem, researchers study and generate artificial instances to validate their proposed algorithms.

In order to obtain random instances of a problem, researchers determine the specific values of a set of parameters to define a case of the problem. In general, those values can be conducted in two ways: selecting values to define problem instances that satisfy some properties (to study "easy" and "hard" scenarios, for example), or generating random values from uniform distributions. Nevertheless, generating artificial instances knowing very little or nothing about them and evaluating the performance of the algorithms using them can induce some wrong assumptions, ideas and/or results.

1.9 Main motivations of the thesis

Our main motivation is to go a step further in the study of the relations between COPs and optimization algorithms. The new knowledge would allow us to present several results that can be considered to improve

any "oracle" proposed in the future: that is to say, the obtained results in these areas facilitate us to propose an algorithm that, due to its design and the definition of the COP (instance), we know in advance is the most efficient algorithm from a particular set of algorithms to solve the problem. However, in this thesis, we do not focus on proposing a new "oracle", but on: 1) the study of COPs avoiding the classical scope of analyzing each problem independently and centering on their exact definition, and on 2) the theoretical studies of algorithms with a lack of studies in the literature. Depending on the scope, we divide the motivations of this thesis in two groups.

In the study of COPs, we strongly believe that the Walsh transform can give us more information about binary-based COPs and their instances. One of the ideal objectives that we believe that the Walsh decomposition is able to achieve, is to present "a common framework" to study any binary-based COP. The first motivation of this thesis is to present a framework based on the Walsh decomposition for binary-based COPs where we can study the problems, compare the existing results in the literature with the proposed framework, establish equivalences and relations among the problems, highlight characteristics of each problem and present classifications of problem instances in "an abstract way".

With the presented framework in mind, our second motivation is to better understand the meaning of the Walsh decomposition and Walsh coefficients in order to extend any result obtained from a particular problem to new problems and scenarios. Particularly, we are interested in the study of the rankings of solutions based on the Walsh decomposition. Because of the bijection that exists between Walsh decomposition and pseudo-Boolean functions, the intriguing research question is if the definition of the Walsh coefficients allows pseudo-Boolean functions to be classified and then extended to the study of the rankings of solutions.

In the study of the algorithms, we will focus on the subfield of EDAs. In this field, particular attention has been paid to the solution of binary-based COPs, where theoretical results at different levels have been provided for different implementations of EDAs. However, this development has not been extended to other non-binary-based COPs, such as permutation-based COPs. In order to bridge this gap, in this thesis our motivation is to extend part of those results to the area of permutation-based COPs.

While several EDAs have been designed for permutation-based COPs which use probabilistic models specifically designed for codifying probability distributions over permutation spaces (for example, [103]), we concentrate on those that use the Mallows model, as it is the one that has received the highest attention in the literature. The Mallows model [78] is considered as the analogous distribution of the Gaussian distribution over the permutation space and it can be included in a more general class of probability models: distance-based exponential models. The Mallows model has been used for designing EDAs in the solution of the Permutation Flowshop Scheduling Problem [17, 18] and the Vehicle Routing Problem with Time Windows [92]. In the mentioned articles, the authors design EDAs in which a Mallows model is learnt from the selected population at each iteration of the algorithm. In [18], the authors named this algorithm Mallows-EDA, whereas in [17, 92] the authors generalize and expand Mallows-EDA. However, even if the mentioned articles have presented competitive results in practice, it is still not clear which mechanisms allow them to obtain those results and there are no studies that analyze the behavior of the applied algorithms. All in all, our motivation behind this second part of the thesis is to present, for the first time, a theoretical analysis of EDAs designed for permutation-based COPs, and a mathematical modeling to study their behavior in several scenarios of increasing complexity. To the best of our knowledge, there are no theoretical studies on permutation-based EDAs. Therefore, we seek general knowledge for a better comprehension of the algorithms designed over the permutation space.

1.10 Outlook of the dissertation

This thesis presents new advances by expanding the analysis in the following subjects: study of the Walsh decomposition, characterization of pseudo-Boolean functions, generation of rankings of solutions and mathematical modeling of algorithms.

In the first part of this thesis, divided in three chapters, we study pseudo-Boolean functions and the instances that each function can generate. In Chapter 2, we consider the Walsh decomposition and calculate the Walsh coefficients of several binary-based COPs: the Unconstrained Binary Quadratic Problem, the Max-Cut Problem and the Number Partitioning Problem. We observe that most of the Walsh coefficients of the studied problems are null values and the non-null Walsh coefficients follow several patterns. Then, in Chapter 3, we study pseudo-Boolean functions. We define the partitions of the solutions based on the Walsh functions and we show the equations that any pseudo-Boolean function of degree $m < n$ fulfill. In addition, the induced partitions are considered to define *the words of a ranking* and *Dyck Words* and to present the necessary conditions of a ranking of solutions to be generated by a pseudo-Boolean function of degree $m < n$. Finally, to finish the first part of the thesis, in Chapter 4, we present several experiments about the rankings of solutions generated by sampling coefficients (parameters) uniformly at random. We observe that there exist rankings more usual (in terms of frequency) than the rest, which might lead to wrong conclusions.

In the second part, we focus on EDAs designed for permutation-based COPs to present a first mathematical modeling to study algorithms. Particularly, in Chapter 5, we present a mathematical framework to study a Mallows-EDA and focus on the convergence behavior of the algorithm for several fitness functions. Considering the ideas presented in previous works in the literature, such as [49, 84, 119], we study the sequence of the expected probability distributions obtained at each iteration of the algorithm (or, equivalently, we study the behavior of the algorithm when the population size tends to infinity). In this way, the randomness is removed and the algorithm is modeled as a dynamical system. Finally, our proposed mathematical framework is used to calculate the convergence behavior of the algorithm for several fitness functions. The studied functions are the constant function, the *needle in a haystack* and a function defined by means of a Mallows model centered at different permutations.

To conclude the thesis, in Chapters 6 and 7, we summarize our work and the main contributions and we present several future works.

Part I

Analysis of COP instances and pseudo-Boolean functions

2

A general framework based on Walsh decomposition

2.1 Introduction

A binary-based COP which has grown in importance in the last years is the Unconstrained Binary Quadratic Problem (UBQP) [15, 62]. The UBQP is the NP-hard problem with the lowest possible degree polynomial function and similar to the study of 2-degree pseudo-Boolean functions. Besides the fact that the basic definition of the UBQP has been studied in a direct way, it has also been used to reformulate other combinatorial problems as UBQP instances. For example, the Linear Ordering Problem, several constrained binary problems and pseudo-Boolean functions of order 3 or more can be redefined as UBQP instances [62]. Moreover, the Ising Problem, which is equivalent to the UBQP, is currently being used for the recent research in Quantum Annealing [58, 105]. Many metaheuristic algorithms have been proposed in the literature not only to solve the UBQP, but to solve its particular cases and generalizations, such as the Maximum Independent Set [76] and the multi-objective UBQP [74] and to develop theoretical analyses of them [20, 81, 98].

Considering all, in this study we overview the main definition and properties of the Walsh decomposition and we calculate the Walsh coefficients of the UBQP. By the Walsh decomposition, not only the main characteristics of a particular problem can be observed, but the relation, common properties and differences of several binary-based COPs can be studied as well. The analysis of "the common framework" would imply that the study of just one specific problem would be enough to present new results for any similar binary problem. Moreover, this framework would be capable of comparing and classifying different problem instances and to study the complexity and characteristics of any particular scenario. This analysis would present the possibility of being able to choose the most appropriate algorithm for a particular instance of a binary-based COP.

The main goals of this study are to overview the main definition and properties of the Walsh coefficients and to calculate the Walsh coefficients of several popular unconstrained binary-based COPs to observe how the common properties among the problems are shown in the Walsh coefficients. In addition, the opposite question will be demonstrated: given a set of Walsh coefficients, is there an instance of a specific problem that produces that set of coefficients? Which constraints must fulfill the Walsh coefficients to define an instance of a problem? With these results, a first example of a representation of "the common framework" of binary problems is shown.

The rest of this chapter is organized as follows. In Section 2.2, the definition of Walsh decomposition and some basic properties are shown. In Section 2.3, the computation of the Walsh coefficients of the UBQP

are calculated. In Section 2.4, the Walsh coefficients of the Max-Cut Problem and the NPP are studied. In Section 2.5, we elaborate about the relevance of the presented framework in order to taxonomize problems and algorithms, pointing out to several relevant research questions. Finally, Section 2.6 concludes the chapter.

2.2 Walsh functions

Let us start with the definition of pseudo-Boolean functions.

Definition 1. ***Pseudo-Boolean functions.*** *Let $\Omega = \{0,1\}^n$ be the search space (binary strings of length n) and $x = x_n x_{n-1} \dots x_1 \in \Omega$ a solution. Then, a function $f : \Omega \longrightarrow \mathbb{R}$ is a pseudo-Boolean function.*

$$\begin{array}{rll} f: & \Omega = \{0,1\}^n & \longrightarrow \mathbb{R} \\ & x = x_n x_{n-1} \cdots x_1 & \longmapsto f(x) = f(x_n x_{n-1} \cdots x_1) \end{array}. \tag{2.1}$$

By using this notation, the solutions can be ordered as binary numbers. Moreover, each character (bit) of the solutions is considered as a binary variable. Let us denote by $X_n, X_{n-1}, \dots, X_1$ the binary variables.

Definition 2. ***Additively Decomposable Function (ADF).*** *Let f be a pseudo-Boolean function. Then, f is an additively decomposable function (ADF) if f can be rewritten in the following way:*

$$f(X_1, \dots, X_n) = f_1(s_1) + \cdots + f_k(s_k), \tag{2.2}$$

where $k \geq 2$; $s_i \subset \{X_1, \dots, X_n\}$; $s_i \not\subset s_j, \forall i \neq j$; and

$$\bigcup_{i=1}^{k} s_i = \{X_1, \dots, X_n\}. \tag{2.3}$$

We say that the decomposition of an ADF is ***minimal*** *if for any i value $f_i(s_i)$ is not an ADF. Bear in mind that even if the subsets s_i are unique in a minimal decomposition, the subfunctions f_i might not be unique.*

Example 1. Let f be the following pseudo-Boolean function:

$$f(x_1, x_2, x_3) = x_1 + x_1 x_2 - x_1 x_3. \tag{2.4}$$

As it can be observed, f is an additively decomposable function:

$$f(x_1, x_2, x_3) = f_1(x_1, x_2) + f_2(x_1, x_3). \tag{2.5}$$

Still, there are more than one option to define f_1 and f_2 subfunctions. One example would be to define f_1 and f_2 in the following way:

$$f_1(x_1, x_2) = cx_1 + x_1 x_2 \text{ and } f_2(x_1, x_3) = dx_1 - x_1 x_3, \text{ such that } c, d \in \mathbb{R} \text{ and } c + d = 1. \tag{2.6}$$

For example, if $c = 1$,

$$f_1(x_1, x_2) = x_1 + x_1 x_2 \text{ and } f_2(x_1, x_3) = -x_1 x_3, \tag{2.7}$$

whereas if $c = 0$,

$$f_1(x_1, x_2) = x_1 x_2 \text{ and } f_2(x_1, x_3) = x_1 - x_1 x_3. \tag{2.8}$$

Definition 3. ***Additively Separable Function (ASF).*** *Let f be a pseudo-Boolean function. Then, f is an additively separable function (ASF) if it is an additively decomposable function and $s_i \cap s_j = \emptyset$, for all pairs of sets (s_i, s_j), for all $i \neq j$.*

When the intersection of the sets of variables is empty, the decomposition is unique. If it is known that the fitness function analyzed in a specific problem is an ASF, then the algorithm used to solve that problem can work independently over each subfunction f_i and finally combine those results.

Next, we briefly summarize the Walsh functions and Walsh decomposition, which is enough for the comprehension of our work. For the reader interested in the formal mathematical definition and properties, we recommend the following works: [22, 45, 46, 53, 108].

Definition 4. ***Walsh decomposition.*** *The Walsh decomposition is an additive decomposition of a pseudo-Boolean function using Walsh functions. Any pseudo-Boolean function f can be written as a Walsh polynomial:*

$$f(x) = \sum_{i=1}^{2^n} \alpha_{s_i} W_{s_i}(x), \tag{2.9}$$

where $s_i \subseteq \{X_1, \ldots, X_n\}$, $\alpha_{s_i} \in \mathbb{R}$ is the Walsh coefficient of f associated to the set s_i and

$$W_{s_i}(x) := \prod_{X_j \in s_i} \begin{cases} +1, \, x_j = 1 \\ -1, \, x_j = 0. \end{cases} \tag{2.10}$$

is the Walsh function associated to the set s_i, for all non-empty subsets s_i of $\{X_1, \ldots, X_n\}$. We define $W_\emptyset(x) = 1$ for any solution x. The functions W_{s_i} form an orthogonal basis for the space of all pseudo-Boolean functions. For any solution x, the calculus of $W_{s_i}(x)$ can be interpreted in the following way:

$$W_{s_i}(x) = \begin{cases} +1, \, \textit{if } |\{X_j \in s_i : x_j = 0\}| \equiv 0 \ (mod 2) \\ -1, \, \textit{otherwise.} \end{cases} \tag{2.11}$$

This second notation shows that for a solution x the parity of the number of 0s in the subset s_i is enough to know if $W_{s_i}(x) = 1$ or not.

To simplify the notation, let us denote the variables of each Walsh coefficient with subscripts: for example, $\alpha_{\{X_i, X_j\}} = \alpha_{\{i,j\}}$. Talking of variables, the Walsh coefficients are ordered based on binary numbers: for any variable X_i, the k-th Walsh coefficient relates the variable X_i if the number $k-1$ in binary form fulfills $x_i = 1$, for any $k = 1, \ldots, 2^n$. On the other hand, a superscript α^f is used if it is required to express the Walsh coefficient of a particular function f.

The Walsh decomposition of a pseudo-Boolean function is unique, i.e., there is only one Walsh decomposition for each pseudo-Boolean function. In order to calculate the Walsh coefficients of a pseudo-Boolean function, we will use the Walsh-Hadamard transform. First, let us calculate $2^n \times 2^n$ Hadamard matrix by Sylvester's construction [100].

$$H_0 = [1] \ ; \ H_n = \begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{bmatrix}, \forall n \geq 1. \tag{2.12}$$

Once we have H_n, the Walsh coefficients of a function f are calculated in the following way:

$$\alpha := \begin{bmatrix} \alpha_\emptyset \\ \alpha_{\{1\}} \\ \alpha_{\{2\}} \\ \alpha_{\{1,2\}} \\ \cdots \\ \alpha_{\{1,\ldots,n\}} \end{bmatrix} = \frac{1}{2^n} H_n \cdot \begin{bmatrix} f(1\ldots11) \\ f(1\ldots10) \\ f(1\ldots01) \\ f(1\ldots00) \\ \cdots \\ f(0\cdots00) \end{bmatrix}. \tag{2.13}$$

As it can be observed, in general, it is necessary to know all the fitness function values to determine the Walsh coefficients. Note that $\alpha_\emptyset$ is the average fitness function value. Let us denote by F the matrix of all the fitness function values ordered decreasingly according to their binary number:

$$F = \begin{bmatrix} f(1\ldots11) \\ f(1\ldots10) \\ f(1\ldots01) \\ \cdots \\ f(0\cdots00) \end{bmatrix}. \tag{2.14}$$

Once we know how the Walsh coefficients of a function can be calculated, the next step is to observe several basic properties. These properties are some of the most used properties in the literature in practice and their proofs are trivial. Let us start with the addition property and the scalar multiplication.

Lemma 1. *Let $t(x) = c_1 \cdot f(x) + c_2 \cdot g(x)$, where $f(x)$ and $g(x)$ are two pseudo-Boolean functions, and $c_1, c_2 \in \mathbb{R}$. Let us denote α^t, α^f and α^g the Walsh coefficients of $t(x)$, $f(x)$ and $g(x)$, respectively. Then, $\alpha^t = c_1 \cdot \alpha^f + c_2 \cdot \alpha^g$.*

Secondly, let us show how the Walsh coefficients of a function are altered when a function is extended to a bigger domain.

Lemma 2. *Let f be a pseudo-Boolean function defined over the set $\{X'_1, \ldots, X'_k\}$ and f^* the extension of f defined over the set $\{X_1, \ldots, X_n\}$, where $\{X'_1, \ldots, X'_k\} \subset \{X_1, \ldots, X_n\}$: that is to say,*

$$f^*(x_n x_{n-1} \ldots x_1) = f(x'_k x'_{k-1} \ldots x'_1), \tag{2.15}$$

where $x'_k x'_{k-1} \ldots x'_1$ is the binary substring of $x_n x_{n-1} \ldots x_1$. Let us denote α^f and α^{f^*} the Walsh coefficients of f and f^*, respectively. Then,

$$\alpha_s^{f^*} = \begin{cases} \alpha_s^f, & \text{if } s \subseteq \{X'_1, \ldots, X'_k\} \\ 0, & \text{otherwise.} \end{cases} \tag{2.16}$$

Therefore, combining both results, if a fitness function f can be decomposed as a sum of j subfunctions f_i, then the Walsh coefficients of the function f is the sum of Walsh coefficients of all the subfunctions f_i after extending its domain to the set $\{X_1, \ldots, X_n\}$. That is to say, if

$$f(x_n x_{n-1} \ldots x_1) = f_1(s_1) + f_2(s_2) + \cdots + f_j(s_j) \tag{2.17}$$

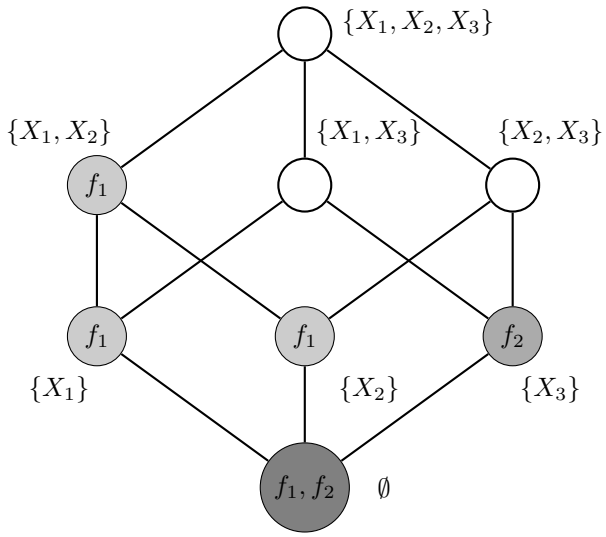


Fig. 2.1: Graphical representation of the power set of $\{X_1, X_2, X_3\}$ and example of the Walsh coefficients equivalence with the subfunctions f_1 and f_2.

such that $s_i \subset \{X_1, \ldots, X_n\}$ and f_i^* is the extension of the subfunction f_i to the domain $\{X_1, \ldots, X_n\}$ for any $i = 1, \ldots, j$ value, then

$$\alpha^f = \alpha^{f_1^*} + \alpha^{f_2^*} + \cdots + \alpha^{f_j^*}. \tag{2.18}$$

This interpretation can be displayed in a graphic. If we draw the power set of $\{X_1, \ldots, X_n\}$ with respect to the inclusion, each node represents the Walsh coefficients to the associated set of variables. If the node has at least one dependent subfunction f_i, it means that the Walsh coefficient is the sum of all the Walsh coefficients of the subfunctions associated to that set of variables. If there are no subfunctions f_i, then the Walsh coefficient associated to that subset is 0.

Figure 2.1 shows an example of how the coefficients of a Walsh decomposition are dependent on the Walsh coefficients of its subfunctions. The function f displayed for the figure is $f(X_1, X_2, X_3) = f_1(X_1, X_2) + f_2(X_3)$. In this example, the nodes with the label f_1 are part of the power set of $\{X_1, X_2\}$, the nodes with the label f_2 are part of the power set of X_3; and finally the nodes with no labels are neither part of the power set of $\{X_1, X_2\}$ nor the power set of $\{X_3\}$. Hence, it is easy to check which Walsh coefficients are dependent on its subfunctions or not. Moreover, we can summarize it with the following expression:

$$\alpha_s^f = \begin{cases} \alpha_s^{f_1} + \alpha_s^{f_2}, & \text{if } s = \emptyset \\ \alpha_s^{f_1}, & \text{if } s \subseteq \{X_1, X_2\} \text{ and } s \neq \emptyset \\ \alpha_s^{f_2}, & \text{if } s = \{X_3\} \\ 0, & \text{otherwise.} \end{cases} \tag{2.19}$$

In addition, bearing this idea in mind, two observations known in the literature are obtained in a direct way. The first observation is presented as the following lemma.

Lemma 3. *f is an ADF if and only if $\alpha^{(f)}_{\{1,\ldots,n\}} = 0$.*

The proof of the lemma is trivial. The second observation is that any nk-landscape function (see [60, 73]) has at most $n \cdot 2^{k+1}$ non-null Walsh coefficients. To get the maximum number of non-null Walsh coefficients, the defined nk-landscape function must be an ASF.

2.3 Walsh coefficients of the UBQP

In the next two sections, some known unconstrained binary-based COPs will be considered. For each problem, their Walsh coefficients, the number of required parameters to define each problem and the equivalences among them have been studied. The results presented in this section are stated with their respective proof in a simplified version and the results from Section 2.4 are stated as corollaries. The complete proofs are based on the definition of the Walsh coefficients and the uniqueness of Walsh polynomials to describe pseudo-Boolean functions. These results can be directly calculated from the Walsh transform.

Our first studied problem is the UBQP. As previously mentioned, UBQP (which is equivalent to the Ising Problem) is one of the most used ADF studied in the literature because of the simplicity of its definition and its application in real-world problems. In Section 2.4, two particular cases of the UBQP are studied: the Max-Cut Problem and the NPP.

Definition 5. ***Unconstrained Binary Quadratic Problem (UBQP).*** *The goal of this problem is to maximize a quadratic fitness function by a suitable choice of binary variables. Let n be the size of the problem, $M = [a_{ij}]_{i,j=1}^{n}$ a matrix of real values of size $n \times n$, and $x_n x_{n-1} \dots x_1$ an n length binary string. Then the objective of the problem is to find a solution $x_n x_{n-1} \dots x_1$ that maximizes the following sum:*

$$f(x_n x_{n-1} \dots x_1) = \sum_{i,j=1}^{n} a_{ij} x_i x_j. \tag{2.20}$$

It is common to assume that M is upper triangular or symmetric, without loss of generality. Let us consider the former structure.

Let us calculate the Walsh coefficients of an UBQP.

Lemma 4. *For $n = 1$, the Walsh coefficients of the UBQP are:*

$$\alpha_\emptyset = \alpha_{\{1\}} = \frac{a_{11}}{2}. \tag{2.21}$$

Proof. When $n = 1$, the matrix of values is $M = [a_{11}]$. So, $f(1) = a_{11}$ and $f(0) = 0$. Therefore,

$$\begin{bmatrix} \alpha_\emptyset \\ \alpha_{\{1\}} \end{bmatrix} = \frac{1}{2} H_1 \cdot F = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} a_{11} \\ 0 \end{bmatrix} = \begin{bmatrix} a_{11}/2 \\ a_{11}/2 \end{bmatrix}. \tag{2.22}$$

□

Lemma 5. *For $n \geq 2$, the Walsh coefficients of the UBQP are as follows:*

$$\begin{aligned}
\alpha_{\emptyset} &= \frac{1}{4}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} a_{ij} + \frac{1}{2}\sum_{i=1}^{n} a_{ii};\\
\alpha_{\{i\}} &= \frac{1}{4}\left(\sum_{j=1}^{i} a_{ji} + \sum_{j=i}^{n} a_{ij}\right), 1 \leq i \leq n;\\
\alpha_{\{i,j\}} &= \frac{a_{ij}}{4}, 1 \leq i < j \leq n;\\
\alpha_{s} &= 0, \forall s \subseteq \{X_1, \ldots, X_n\} \textit{ such that } |s| > 2.
\end{aligned} \tag{2.23}$$

Proof. Let us prove it by induction. Before starting with the proof, let us explain some notation used throughout the proof. Let us denote by $F^{(i)}$ the $2^i \times 1$ objective function values matrix F and $\alpha^{(i)}$ as the $2^i \times 1$ Walsh coefficients matrix α.

For $n = 2$,

$$F^{(2)} = \begin{bmatrix} f(11) \\ f(10) \\ f(01) \\ f(00) \end{bmatrix} = \begin{bmatrix} a_{11} + a_{12} + a_{22} \\ a_{22} \\ a_{11} \\ 0 \end{bmatrix}. \tag{2.24}$$

Therefore,

$$\alpha^{(2)} = \begin{bmatrix} \alpha_{\emptyset} \\ \alpha_{\{1\}} \\ \alpha_{\{2\}} \\ \alpha_{\{1,2\}} \end{bmatrix} = \frac{1}{2^2} H_2 \cdot F^{(2)} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_{11} + a_{12} + a_{22} \\ a_{22} \\ a_{11} \\ 0 \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 2a_{11} + a_{12} + 2a_{22} \\ 2a_{11} + a_{12} \\ a_{12} + 2a_{22} \\ a_{12} \end{bmatrix}, \tag{2.25}$$

obtaining the same solutions of the statement.

Now, let us assume that for $n-1$ the result is obtained. Let us calculate for n.

$$\begin{aligned}
\alpha^{(n)} &= \frac{1}{2^n} H_n \cdot F^{(n)}\\
&= \frac{1}{2^n}\begin{bmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{bmatrix} \cdot \begin{bmatrix} A^{(n-1)} + F^{(n-1)} \\ F^{(n-1)} \end{bmatrix}\\
&= \frac{1}{2^n}\begin{bmatrix} H_{n-1} \cdot A^{(n-1)} + 2H_{n-1} \cdot F^{(n-1)} \\ H_{n-1} \cdot A^{(n-1)} \end{bmatrix},
\end{aligned} \tag{2.26}$$

where $A^{(n-1)}$ is a $2^{n-1} \times 1$ auxiliary matrix where each row is $f(1x_{n-1} \ldots x_1) - f(0x_{n-1} \ldots x_1)$, ordered as binary numbers. So,

$$A^{(n-1)} = \begin{bmatrix} \sum_{i=1}^{n} a_{in} \\ \sum_{i=2}^{n} a_{in} \\ \sum_{\substack{i=1 \\ i\neq 2}}^{n} a_{in} \\ \sum_{i=3}^{n} a_{in} \\ \cdots \\ a_{nn} \end{bmatrix}. \tag{2.27}$$

By the definition of the inductive process of the Hadamard matrix, the expression $\alpha^{(n)}$ can be simplified according to the Walsh coefficients of $\{X_1, \ldots, X_{n-1}\}$ variables:

$$\alpha^{(n)} = \begin{bmatrix} \alpha^{(n-1)} \\ \mathbf{0} \end{bmatrix} + \frac{1}{2^n} \begin{bmatrix} H_{n-1} \cdot A^{(n-1)} \\ H_{n-1} \cdot A^{(n-1)} \end{bmatrix}, \tag{2.28}$$

where $\mathbf{0}$ is a $2^{n-1} \times 1$ null matrix and

$$H_{n-1} \cdot A^{(n-1)} = \begin{bmatrix} 2^{n-2} \sum_{j=1}^{n-1} a_{jn} + 2^{n-2} \sum_{j=1}^{n-1} a_{nj} + 2^{n-1} a_{nn} \\ 2^{n-2}(a_{1n} + a_{n1}) \\ 2^{n-2}(a_{2n} + a_{n2}) \\ 0 \\ 2^{n-2}(a_{3n} + a_{n3}) \\ 0 \\ 0 \\ 0 \\ 2^{n-2}(a_{4n} + a_{n4}) \\ \cdots \end{bmatrix}. \tag{2.29}$$

Consequently, by the induction hypothesis and expanding the equations, the following Walsh coefficients are obtained:

$$\begin{aligned}
\alpha_{\emptyset}^{(n)} &= \alpha_{\emptyset}^{(n-1)} + \frac{1}{2^n} \left(2^{n-2} \sum_{j=1}^{n-1} a_{jn} + 2^{n-1} a_{nn} \right) = \frac{1}{4} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} a_{ij} + \frac{1}{2} \sum_{i=1}^{n} a_{ii}; \\
\alpha_{\{i\}}^{(n)} &= \alpha_{\{i\}}^{(n-1)} + \frac{1}{2^n} \left(2^{n-2}(a_{in}) \right) = \frac{1}{4} \left(\sum_{j=1}^{i} a_{ji} + \sum_{j=i}^{n} a_{ij} \right), 1 \leq i \leq n-1; \\
\alpha_{\{n\}}^{(n)} &= 0 + \frac{1}{2^n} \left(2^{n-2} \sum_{j=1}^{n-1} a_{jn} + 2^{n-1} a_{nn} \right) = \frac{1}{4} \left(\sum_{j=1}^{n} a_{jn} + a_{nn} \right); \\
\alpha_{\{i,j\}}^{(n)} &= \alpha_{\{i,j\}}^{(n-1)} + 0 = \frac{a_{ij}}{4}, 1 \leq i < j \leq n-1; \\
\alpha_{\{i,n\}}^{(n)} &= 0 + \frac{1}{2^n} \left(2^{n-2} a_{in} \right) = \frac{a_{in}}{4}, 1 \leq i \leq n-1; \\
\alpha_{\{i,j,k\}}^{(n)} &= \alpha_{\{i,j,k\}}^{(n-1)} + 0 = 0, 1 \leq i < j < k \leq n,
\end{aligned} \tag{2.30}$$

and this last argument can be used for any Walsh coefficient associated to more than 2 variables. Therefore, the lemma is proved. □

This lemma works for any matrix M, without any condition about the coefficients a_{ij}. In addition, this lemma helps us to understand the opposite problem and our next step: given a set of Walsh coefficients, is there an UBQP instance that produces that set of coefficients? In that case, how does the matrix M look like? The following lemma answers both questions.

Lemma 6. *Given α Walsh coefficients, they have been produced by an UBQP instance if they fulfill the following two conditions:*

1. $\alpha_s = 0$, *if* $|s| > 2$.
2. $$\alpha_\emptyset = \sum_{i=1}^{n} \alpha_{\{i\}} - \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \alpha_{\{i,j\}}. \tag{2.31}$$

Moreover, the UBQP matrix defined by the given α coefficients is the matrix $M = [a_{ij}]_{i,j=1}^{n}$ such that:

$$\begin{aligned} a_{ij} &= 4\alpha_{\{i,j\}},\ 1 \leq i < j \leq n; \\ a_{ii} &= 2\left(\alpha_{\{i\}} - \sum_{j=1}^{i-1} \alpha_{\{j,i\}} - \sum_{j=i+1}^{n} \alpha_{\{i,j\}}\right), 1 \leq i \leq n. \end{aligned} \tag{2.32}$$

Proof. The first constraint is obtained from the fact that the fitness function of an UBQP can only be described as a polynomial of maximum order 2. The second constraint is due to the fact that there are $1 + n + \binom{n}{2}$ Walsh coefficients and $\binom{n+1}{2}$ parameters on the matrix M to define an UBQP instance. Consequently, there exists one Walsh coefficient which is dependent. The easiest way to calculate that dependency is to consider the fitness function value of the solution $0 \ldots 0$:

$$f(0 \ldots 0) = 0 = \alpha_\emptyset - \sum_{i=1}^{n} \alpha_{\{i\}} + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \alpha_{\{i,j\}} \tag{2.33}$$

and the second constraint is obtained. To obtain the description of the parameters of M, the equation system described in the proof of Lemma 5 must be solved.

$$\begin{aligned} \alpha_{\{i,j\}} = \frac{a_{ij}}{4} \Longrightarrow a_{ij} = 4\alpha_{\{i,j\}},\ 1 \leq i < j \leq n; \\ \alpha_{\{i\}} = \frac{1}{2}a_{ii} + \frac{1}{4}\sum_{\substack{j=1 \\ i \neq j}}^{n} a_{ij} = \frac{1}{2}a_{ii} + \sum_{\substack{j=1 \\ i \neq j}}^{n} \alpha_{\{i,j\}} \Longrightarrow a_{ii} = 2\left(\alpha_{\{i\}} - \sum_{\substack{j=1 \\ i \neq j}}^{n} \alpha_{\{i,j\}}\right),\ 1 \leq i \leq n. \end{aligned} \tag{2.34}$$

□

This result can be interpreted geometrically. Let us consider the Euclidean space of Walsh coefficients associated to less than 3 variables whose dimension is $d = 1 + n + \binom{n}{2}$. All the Walsh coefficients of an UBQP except one ($\alpha_\emptyset$, for example) are linearly independent. Consequently, UBQP can be represented as a $d-1$ dimensional hyperplane of $\mathbb{R}^d$. Moreover, the second constraint of Lemma 6 specifies which exact hyperplane is considered. In addition, it must be mentioned that if a real additive term would be added to the definition of the UBQP, then any Walsh decomposition of order 2 or less can be represented as a particular UBQP instance.

2.4 Particular cases of UBQP

Definition 6. ***Max-Cut Problem.*** *Let $G(V, E)$ be an undirected graph ($|V| = n$) in which every edge $\{v_i, v_j\} \in E$ has an assigned interaction weight $C_{ij} \in \mathbb{R}$. The objective of this problem is to find a subset of vertexes $W \subseteq V$ such that maximizes*

$$\sum_{\{v_i,v_j\}\in\delta(W)} C_{ij}, \tag{2.35}$$

where $\delta(W)$ is the set of edges with just one vertex in the subset W: that is to say, $v_i \in W$ and $v_j \in V \backslash W$, or viceversa.

Any solution of the Max-Cut Problem can be described with a binary string of length n. Each x_i determines if the vertex v_i is in W or not. Let us denote $x_i = 1$ if $v_i \in W$, and $x_i = 0$ otherwise. So, if $x_i = x_j$, then $\{v_i, v_j\} \notin \delta(W)$. Considering this interpretation, it is possible to rewrite the objective function in the following way:

$$\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} C_{ij}\left(x_i + x_j - 2x_ix_j\right) = \sum_{i=1}^{n}\left(\sum_{j=1}^{i-1} C_{ji} + \sum_{j=i+1}^{n} C_{ij}\right)x_i - 2\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} C_{ij}x_ix_j. \tag{2.36}$$

The Max-Cut Problem is a particular case of the UBQP: that is to say, any Max-Cut Problem of n vertexes can be described as an UBQP of $n \times n$ dimensional matrix of real values. We can rewrite it as an UBQP with the following transformation:

$$a_{ii} = \sum_{j=1}^{i-1} C_{ji} + \sum_{j=i+1}^{n} C_{ij} \text{ and } a_{ij} = -2C_{ij}. \tag{2.37}$$

The particularity of the case can be easily identified calculating its Walsh coefficients as well.

Corollary 1. *For $n \geq 2$, the Walsh coefficients of the Max-Cut Problem are as follows:*

$$\begin{aligned}\alpha_{\emptyset} &= \frac{1}{2}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} C_{ij};\\ \alpha_{\{i,j\}} &= -\frac{C_{ij}}{2}, 1 \leq i < j \leq n;\\ \alpha_s &= 0, \forall s \subseteq \{X_1, \ldots, X_n\} \text{ such that } |s| \neq 0, 2.\end{aligned} \tag{2.38}$$

However, it must be mentioned that any UBQP of size n can be described as a Max-Cut Problem of $n+1$ variables and fixing the value of one variable [8].

Now let us consider the opposite problem: given a set of Walsh coefficients, is there a Max-Cut instance that produces that set of coefficients?

Corollary 2. *Given α Walsh coefficients, they describe an instance of the Max-Cut Problem if they fulfill the following two conditions:*

1. $\alpha_s = 0$, *if* $|s| \neq 0, 2$.
2. $$\alpha_{\emptyset} = -\sum_{i=1}^{n}\sum_{j=i+1}^{n} \alpha_{\{i,j\}}. \tag{2.39}$$

Moreover, the edges of the Max-Cut Problem defined by the given α Walsh coefficients are the following ones:

$$C_{ij} = -2\alpha_{\{i,j\}}.$$

Consequently, for a particular set of Walsh coefficients, we can ensure if there exists a Max-Cut instance which produces those Walsh coefficients. Bearing in mind all the observations about the Max-Cut problem and its similarities with the UBQP, the Walsh coefficients of the Max-Cut problem can be geometrically interpreted as a subspace of the hyperplane defined by the Walsh coefficients of the UBQP in $\mathbb{R}^d$. The number of linearly independent Walsh coefficients for the Max-Cut Problem is $\binom{n}{2}$ and every instance of the Max-Cut Problem can be interpreted as an UBQP instance with a complete symmetry property for all the variables, without distinctions between the solutions $x_n \ldots x_1$ and $1 \ldots 1 - x_n \ldots x_1$. The subspace of the Max-Cut Problem is described by the second constraint of Corollary 2.

Definition 7. ***Number Partitioning Problem (NPP).*** *Let $Z = \{z_1, \ldots, z_n\}$ be a set of non-negative integer numbers. The objective of the problem is to find a subset P of Z such that the difference between the sum of the values of P and $Z \backslash P$ is minimized:*

$$\left| \sum_{z_i \in P} z_i - \sum_{z_i \in Z \backslash P} z_i \right|. \tag{2.40}$$

That is to say, for any binary solution $x_n \ldots x_1$, if we denote $x_i = 1$ if $z_i \in P$ and $x_i = 0$ if $z_i \in Z \backslash P$, we want to minimize the following difference:

$$f(x_n \ldots x_1) = \left| \sum_{x_i=1} z_i - \sum_{x_i=0} z_i \right| = \left| \sum_{i=1}^{n} z_i - 2\sum_{i=1}^{n} z_i x_i \right|. \tag{2.41}$$

If there exists a solution x' such that $f(x') = 0$, then x' is the optimal solution and Z has a perfect partition. If there exists a solution x' such that $f(x') = 1$, then x' is the optimal solution.

In order to avoid several trivial situations, let us assume that $z_i \neq 0$, for any i value. NPP can be modeled as an instance of an UBQP. To do so, f^2 fitness function is calculated, instead of f. This variation does not affect on the relative comparisons among the solutions: for any two solutions x and y, $f(x) > f(y) \Longleftrightarrow f^2(x) > f^2(y)$ due to the non-negativity of the numbers. Hence, they produce the same ranking of solutions. For that reason, any algorithm based on the ranking of solutions will behave similarly for f and f^2 fitness functions. In order to simplify the notation, let us denote $c = \sum_{i=1}^{n} z_i$. So,

$$\begin{aligned} f^2(x_n \dots x_1) = & \left(c - 2\sum_{i=1}^{n} z_i x_i \right)^2 \\ & = c^2 - 4c\left(\sum_{i=1}^{n} z_i x_i\right) + 4\left(\sum_{i=1}^{n} z_i x_i\right)^2 \\ & = c^2 + 4\sum_{i=1}^{n} z_i(z_i - c)x_i + 8\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} z_i z_j x_i x_j. \end{aligned} \tag{2.42}$$

Consequently, we can model this problem as an UBQP. Dropping the additive constant c^2 and defining

$$a_{ii} = 4z_i(z_i - c) \text{ and } a_{ij} = 8z_i z_j \ (i < j) \tag{2.43}$$

an equivalent UBQP is obtained. If the constant term c^2 is kept, then the coefficient $\alpha_\emptyset$ will increase, but the ranking of solutions will be the same.

Furthermore, it can be observed that any set of Walsh coefficients which describes a NPP instance also describes an instance of the Max-Cut Problem. If we define $C_{ij} = -4z_i z_j$, then the Max-Cut Problem with the defined C_{ij} values generates the same objective function values (and consequently the same Walsh coefficients). Nevertheless, bear in mind that Max-Cut Problem is a maximization problem, whereas NPP is a minimization problem. To generate the opposite ranking of solutions, it is enough to use the definition of the coefficients of the Max-Cut Problem multiplied by -1: $C_{ij} = 4z_i z_j$.

Corollary 3. *Let f be the function generated by a NPP and $c^2 = \left(\sum_{i=1}^{n} z_i\right)^2$. Then, for $n \geq 2$, the Walsh coefficients of the function $f^2 - c^2$ can be written as follows:*

$$\begin{aligned} \alpha_\emptyset = & -2\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} z_i z_j; \\ \alpha_{\{i,j\}} = & \ 2z_i z_j, 1 \leq i < j \leq n; \\ \alpha_s = & \ 0, \forall s \subseteq \{X_1, \dots, X_n\} \text{ such that } |s| \neq 0, 2. \end{aligned} \tag{2.44}$$

This result is quite surprising because if we calculate the Walsh coefficients directly from the definition of the NPP (with the fitness function f instead of f^2), then the number of non-null Walsh coefficients is quite

larger. This is due to the symmetry property for all the variables, without distinctions between the solutions $x_n \dots x_1$ and $1 \dots 1 - x_n \dots x_1$ (analogous to the Max Cut Problem). Specifically, all the Walsh coefficients associated to an even number of variables are non-null, whereas for f^2 there are $n(n-1)/2+1$ non-null Walsh coefficients at most. Therefore, this example shows that different definitions of equivalent problems in terms of the ranking of solutions they produce can generate different Walsh decompositions, which increases the interest of studying this framework.

Considering the non-null Walsh coefficients, let us calculate the main constraints to know if a set of Walsh coefficients can be produced by a NPP instance. Because f^2 can be described as a Max-Cut Problem, the Walsh coefficients associated to zero or two variables are the only non-null Walsh coefficients. It remains to observe if the non-null coefficients must fulfill more specific constraints.

Bear in mind that NPP is a combinatorial problem which each instance is defined by n non-negative integer numbers, and when f^2 is calculated, the generated coefficients are dependent on those n numbers. On the other hand, the number of Walsh coefficients associated to two variables are $n(n-1)/2$. Consequently, the number of possible combinations of Walsh coefficients produced by NPP instances is much lower than the ones produced by Max-Cut instances.

Corollary 4. *Given α Walsh coefficients, they describe an NPP instance if they fulfill the following conditions:*

1. $\alpha_s = 0$, *if* $|s| \neq 0, 2$.
2. $$\alpha_\emptyset = -\sum_{i=1}^{n}\sum_{j=i+1}^{n} \alpha_{\{i,j\}}. \tag{2.45}$$
3. *When $n \geq 4$, for all $1 \leq i < j < k < l \leq n$,*
$$\alpha_{\{i,j\}}\alpha_{\{k,l\}} = \alpha_{\{i,k\}}\alpha_{\{j,l\}} = \alpha_{\{i,l\}}\alpha_{\{j,k\}}. \tag{2.46}$$
4. $\alpha_s \equiv 0 \ (mod\ 2)$ *($\alpha_s \in \mathbb{N}$).*
5. *For all $1 \leq i < j < k \leq n$, $\frac{\alpha_{\{i,k\}}\alpha_{\{j,k\}}}{2\alpha_{\{i,j\}}}$ is a perfect square.*

Moreover, the numbers z_i of the NPP defined by the given α Walsh coefficients are the following ones: for all $i \neq j \neq k \neq i$,
$$z_i = \sqrt{\frac{\alpha_{\{i,j\}}\alpha_{\{i,k\}}}{2\alpha_{\{j,k\}}}}. \tag{2.47}$$

Several observations can be deduced from the previous corollary. Firstly, the first two constraints are the same ones obtained for the Max-Cut Problem. Secondly, the third constraint requires to observe all the equalities in groups of 4 indexes. However, when $n \geq 5$, some equations can be deduced from other equalities, so it is not necessary to check all of them. In the following example this idea is shown.

Example 2. When $n = 5$, if $\alpha_{\{1,2\}}\alpha_{\{3,4\}} = \alpha_{\{1,3\}}\alpha_{\{2,4\}}$ and $\alpha_{\{1,2\}}\alpha_{\{3,5\}} = \alpha_{\{1,3\}}\alpha_{\{2,5\}}$ are satisfied, then it follows $\alpha_{\{2,5\}}\alpha_{\{3,4\}} = \alpha_{\{2,4\}}\alpha_{\{3,5\}}$. Hence, two equalities deduce a third one.

The last detail is about the forth and fifth constraints. Both constraints are associated to the fact that NPP is defined over a set of non-negative integer numbers. Because of that, these observations make us think about a generalization of the NPP over a set of non-negative real numbers.

Geometrically, the Euclidean space defined by the Walsh coefficients of f^2 is a subspace of the Max-Cut Problem. Indeed, the dimension of the subspace of $\mathbb{R}^d$ for the NPP is n. In this comparison, the main difference between the space of Walsh coefficients of the Max-Cut Problem and the NPP is the domain of the problems: the NPP parameters are defined over natural numbers, whereas the Max-Cut Problem parameters are real values.

2.5 Discussion

A remarkable fact of our previous results is that there exist functions that produce the same ranking of solutions, but however they have completely different set of Walsh coefficients. Given that an algorithm which only considers the ranking of solutions will behave the same in those functions, a relevant question is what the smallest non-null set of Walsh coefficients for a specific function is. This is equivalent to ask for the minimal structure of Walsh coefficients for a specific ranking of solutions of the search space. The research about the connection between Walsh coefficients and rankings can open interesting avenues. A first question is to know the set of rankings (functions) that can be generated with some non-null Walsh coefficients, or what the smallest non-null set of Walsh coefficients to make a problem NP-hard. Furthermore it would be possible to think in algorithms that are efficient for some kind of rankings and associate them with Walsh coefficients.

2.6 Conclusions

In this chapter, the Walsh coefficients have been obtained for several unconstrained binary-based COPs. In Section 2.2, some basic properties of Walsh decomposition have been revised to show the interest of this orthogonal basis. In Sections 2.3 and 2.4, the Walsh coefficient of some known binary-based COPs have been calculated. Besides calculating the Walsh polynomial of a problem instance, we have also studied the opposite direction: given a Walsh polynomial, in which cases they define a problem instance. From these results, the similarities and differences known in the literature have been checked. Moreover, several comments about the geometrical interpretation of the problems and the Walsh coefficients have been added. Finally, in Section 2.5, possible future research questions are briefly suggested and commented.

3

Characterization of rankings generated by pseudo-Boolean functions

3.1 Introduction

In Chapter 2, it is shown that the Walsh decomposition indicates the relation among the variables. Moreover, it is observed that, in several binary-based COPs, the majority of the Walsh coefficients are zeros. Based on that, our study continues studying the meaning of the null Walsh coefficients and analyzes its implications. Let us present an example about a null Walsh coefficient.

Example 3. Let f be an UBQP instance and s a subset of variables such that $n \geq |s| > 2$. By definition of the Walsh functions,

$$W_s(x) := \prod_{X_j \in s} \begin{cases} +1, x_j = 1 \\ -1, x_j = 0 \end{cases} = \begin{cases} +1, \text{ if } |\{X_j \in s : x_j = 0\}| \equiv 0 \ (mod\ 2) \\ -1, \text{ otherwise.} \end{cases} \tag{3.1}$$

On the other hand, by Lemma 5, $\alpha_s = 0$. Therefore, considering that the Walsh functions is a complete set of orthogonal functions, the following equality is obtained.

$$\alpha_s = \sum_{x \in \{0,1\}^n} f(x) \cdot W_s(x) = 0 \tag{3.2}$$

Let us define

$$\mathcal{E}_s = \{x \in \{0,1\}^n : |\{X_j \in s : x_j = 0\}| \equiv 0 \ (mod\ 2)\} \tag{3.3}$$

and

$$\mathcal{O}_s = \{x \in \{0,1\}^n : |\{X_j \in s : x_j = 0\}| \equiv 1 \ (mod\ 2)\}\,. \tag{3.4}$$

(The formal definitions of the sets $\mathcal{E}_s$ and $\mathcal{O}_s$ are introduced in Section 3.2).

Hence,

$$\sum_{x\in\{0,1\}^n} f(x)\cdot W_s(x)=0 \Longleftrightarrow \sum_{x\in\{0,1\}^n} f(x)\cdot \begin{cases} +1\ , & x\in\mathcal{E}_s \\ -1\ , & x\in\mathcal{O}_s \end{cases} =0$$
$$\Longleftrightarrow \sum_{x\in\mathcal{E}_s} f(x)-\sum_{x\in\mathcal{O}_s} f(x)=0$$
$$\Longleftrightarrow \sum_{x\in\mathcal{E}_s} f(x)=\sum_{x\in\mathcal{O}_s} f(x). \tag{3.5}$$

Example 3 shows an equality (deduced from the set of variables s) that any UBQP instance must fulfill. Consequently, for each set of variables s such that $|s|>2$, an equality is defined which all the UBQP instances must fulfill.

Following the idea of Example 3 and inspired by the works of [19, 35, 54, 55], in this chapter, we analyze for the first time the rankings generated by pseudo-Boolean functions of degree $m\leq n$, being n the size of the search space. Our main contributions are the following. First, we prove that there exist rankings that cannot be generated by a pseudo-Boolean function of degree $m<n$. Moreover, we exactly present the necessary conditions for a ranking to be generated by an m-degree pseudo-Boolean function. We provide a novel and easy-to-compute procedure to check when a ranking cannot be generated by an m-degree pseudo-Boolean function. Secondly, we study if the obtained necessary conditions are sufficient conditions to prove when a ranking can be generated by m-degree pseudo-Boolean functions. When $m=n-1$, we conjecture that the answer is affirmative and we calculate the exact number of rankings generated by $(n-1)$-degree pseudo-Boolean functions; whereas when $m<n-1$, the presented procedure is not sufficient to check if a ranking can be generated by an m-degree pseudo-Boolean function. Throughout this chapter, we present several examples for the particular case of $m=2$ (analogous to the UBQP).

The rest of this chapter is organized as follows. In Section 3.2, the required mathematical concepts are defined. In Section 3.3, the main results are shown: the analysis of the rankings of solutions generated by an m-degree pseudo-Boolean function. Finally, in Section 3.4, conclusions are presented.

3.2 Preliminaries

In this chapter, we focus on pseudo-Boolean functions considered as ranking generators. As mentioned in Chapter 1, any pseudo-Boolean function can be written uniquely as a multi-linear polynomial of degree $m\leq n$ (notice that for any bit x_i, if the rest of the bit values are fixed, then the function f is linear with respect to x_i):

$$f(x)=a_0+\sum_{1\leq i_1\leq n} a_{i_1}x_{i_1}+\sum_{1\leq i_1<i_2\leq n} a_{i_1i_2}x_{i_1}x_{i_2}+\cdots+\sum_{1\leq i_1<\cdots<i_m\leq n} a_{i_1\ldots i_m}x_{i_1}\ldots x_{i_m}. \tag{3.6}$$

Definition 8. ***Degree of a pseudo-Boolean function.*** *A pseudo-Boolean function is of degree $m\leq n$ if the degree of its polynomial representation is m.*

We highly recommend [15] for a deep introduction to pseudo-Boolean functions and their main properties.

Bear in mind that the same ranking of solutions can represent several functions, see Example 4. We denote a ranking generated by a pseudo-Boolean function f with the letter r_f.

Example 4. Let $\Omega = \{0,1\}^2$. The two different 1-degree pseudo-Boolean functions $f(x) = 3x_1 - 2x_2$ and $g(x) = -4 + 6x_1 - 2x_2$ generate the same ranking of solutions.

$$\begin{array}{c|cccc} x & 11 & 10 & 01 & 00 \\ \hline f(x) & 1 & -2 & 3 & 0 \\ g(x) & 0 & -6 & 2 & -4 \end{array} \Longrightarrow \begin{cases} f(01) > f(11) > f(00) > f(10) \\ g(01) > g(11) > g(00) > g(10) \end{cases} \Longrightarrow r_f = r_g = \begin{bmatrix} 01 \\ 11 \\ 00 \\ 10 \end{bmatrix}. \tag{3.7}$$

Moreover, for any fitness function f, real constant c and positive real constant c', the rankings generated by f, $f + c$ and $c' \cdot f$ are the same: r_f.

To simplify, let us assume that the studied pseudo-Boolean functions are injective. Even the presented analysis can be replicated for non-injective pseudo-Boolean functions, the notation needs to be much more tedious. With this simplification in mind, even though there are infinite n-dimensional pseudo-Boolean functions, the number of possible rankings that can be generated by them is $2^n!$, which is also the number of permutations of the group Σ_{2^n}. Consequently, we can group pseudo-Boolean functions that generate the same ranking of solutions and study COPs as the sets of all the rankings that can be generated by all the instances of the problems. Note that all the results we could obtain for a set of rankings can be extended to all the COPs that generate those rankings regardless of how they have been defined. For instance, the set of rankings that can be generated by both the UBQP and the NPP could be solved in the same way.

Notice that, given a pseudo-Boolean function f as in Equation (3.6), the value of the coefficient a_0 does not change the ranking. Because of that, we assume that $a_0 = 0$ for the rest of the manuscript.

Next, let us define a partition of Ω based on the parity of zeros of the solutions. Definition 9 is analogous to the one presented in [102] or the *Hamming weight* [14, 94].

Definition 9. ***Even (odd) solutions.*** *Let $x \in \Omega$ be an even (odd) solution, labeled as E (O), if it contains an even (odd) number of 0 values. Let us denote by $\mathcal{E}$ ($\mathcal{O}$) the set of all even (odd) solutions.*

By definition, $\{\mathcal{E}, \mathcal{O}\}$ is a partition of Ω such that $|\mathcal{E}| = |\mathcal{O}| = 2^{n-1}$. For the presented results in this study, there is no difference if we define even and odd solutions according to the number of ones in a solution. The definition of the set of even (odd) solutions can be extended and defines a partition according to a non-empty set of variables $s \subseteq \{1, \ldots, n\}$.

Definition 10. ***Even (odd) solutions defined by s.*** *Let $s \subseteq \{1, \ldots, n\}$ be a non-empty set of variables and $x \in \Omega$. Then, x is an even (odd) solution defined by s, labeled as E_s (O_s), if it contains an even (odd) number of 0 values from the set of s. Moreover, let us denote by $\mathcal{E}_s$ ($\mathcal{O}_s$) the set of all even (odd) solutions defined by s. By definition, $\{\mathcal{E}_s, \mathcal{O}_s\}$ is a partition of Ω such that $|\mathcal{E}_s| = |\mathcal{O}_s|$.*

When the subset s is clear from the context, we simplify the notation and remove the subscript s from E and O.

3.3 Studying the rankings generated by pseudo-Boolean functions

The main result of this section is to show and prove the existence of rankings of solutions that cannot be generated by any m-degree pseudo-Boolean function, where $m < n$. In addition, the necessary conditions for a ranking to be generated by an m-degree pseudo-Boolean function are presented.

3.3.1 Characterization of pseudo-Boolean functions of degree $m < n$

Let us introduce a characterization of pseudo-Boolean functions according to the partitions of even and odd solutions. To present the characterization of pseudo-Boolean functions, we start with the following lemma.

Lemma 7. *Let $j, n \in \mathbb{N}$, $1 \leq j < n$, a set of variables $\{i_1, \ldots, i_j\} \subset \{1, \ldots, n\}$ and a subset of variables $s \subseteq \{1, \ldots, n\}$ such that $|s| > j$. Then, given a value to the variables with indices in $\{i_1, \ldots, i_j\}$, the number of even and odd solutions defined by s is the same. In other words, for any two j-tuples $(c_1, \ldots, c_j)$, $(d_1, \ldots, d_j) \in \{0,1\}^j$, the following equality holds:*

$$|\{x \in \mathcal{E}_s : x_{i_1} = c_1 \wedge \cdots \wedge x_{i_j} = c_j\}| = |\{x \in \mathcal{O}_s : x_{i_1} = d_1 \wedge \cdots \wedge x_{i_j} = d_j\}|. \tag{3.8}$$

Proof. The lemma is deduced from the definition of the partition $\{\mathcal{E}_s, \mathcal{O}_s\}$. In terms of the relation between the sets $\{i_1, \ldots, i_j\}$ and s, there are three types of possible scenarios: (a) $\{i_1, \ldots, i_j\} \subset s$; (b) $\{i_1, \ldots, i_j\} \not\subset s$ and $s \cap \{i_1, \ldots, i_j\} \neq \emptyset$; and (c) $s \cap \{i_1, \ldots, i_j\} = \emptyset$.

Without loss of generality, let us consider the case (a) and that s has $1 \leq t < n - j$ additional elements apart from $\{i_1, \ldots, i_j\}$. Then, there are 2^{t-1} solutions of $\mathcal{E}_s$ of length $j + t$ and 2^{n-j-t} options for the rest of terms in $\{1, \ldots, n\}$. Therefore, there are in total 2^{n-j-1} solutions of $\mathcal{E}_s$ and 2^{n-j-1} solutions of $\mathcal{O}_s$, where the bit values in the positions $\{i_1, \ldots, i_j\}$ are determined as in Equation (3.8). The cases (b) and (c) are proved analogously. □

Lemma 7 proves our main result. It is stated in Theorem 1.

Theorem 1. *Let $\Omega = \{0,1\}^n$ and $f : \Omega \longrightarrow \mathbb{R}$ a pseudo-Boolean function. Then, f is a pseudo-Boolean function of degree $m < n$ if and only if*

$$\begin{cases} \forall s \subseteq \{1, \ldots, n\} \text{ such that } |s| > m \ , \ \sum_{x \in \mathcal{E}_s} f(x) = \sum_{x \in \mathcal{O}_s} f(x) \\ \exists s \subseteq \{1, \ldots, n\} \text{ such that } |s| = m \text{ and } \sum_{x \in \mathcal{E}_s} f(x) \neq \sum_{x \in \mathcal{O}_s} f(x). \end{cases} \tag{3.9}$$

Furthermore, when the equality holds, the sum $\sum_{x \in \mathcal{E}_s} f(x)$ is half of the sum of the function value of all the solutions of the search space:

$$2^{n-1} \cdot \bar{f} = 2^{n-2} \sum_{1 \leq i_1 \leq n} a_{i_1} + 2^{n-3} \sum_{1 \leq i_1 < i_2 \leq n} a_{i_1 i_2} + \cdots + 2^{n-m-1} \sum_{1 \leq i_1 < i_2 < \cdots < i_m \leq n} a_{i_1 \ldots i_m}, \tag{3.10}$$

where $\bar{f}$ is the average fitness function value of f.

Proof. $\Longrightarrow$ Let f be an m-degree polynomial defined over $\{0,1\}^n$. Considering Equality (3.8) of Lemma 7, for any set s such that $|s| > m$, there are the same number of solutions with $x_{i_1} = 1$, with $x_{i_1} = x_{i_2} = 1$, ... and with $x_{i_1} = \cdots = x_{i_m} = 1$ in $\mathcal{E}_s$ and $\mathcal{O}_s$. Therefore, each coefficient a_{i_1}, $a_{i_1 i_2}$, ..., $a_{i_1 \ldots i_m}$ appears the same number of times in $\sum_{x\in\mathcal{E}_s} f(x)$ and in $\sum_{x\in\mathcal{O}_s} f(x)$ and, consequently, $\sum_{x\in\mathcal{E}_s} f(x) = \sum_{x\in\mathcal{O}_s} f(x)$.

On the other hand, because f is an m-degree polynomial, there exists, at least, one non-null coefficient $a_{i_1 \ldots i_m}$. Consequently, when $s = \{i_1, \ldots, i_m\}$, the solutions such that $x_{i_1} = \cdots = x_{i_m} = 1$ only appear in $\mathcal{E}_s$ whereas the rest of coefficients appear the same number of times in $\sum_{x\in\mathcal{E}_s} f(x)$ and $\sum_{x\in\mathcal{O}_s} f(x)$. So, it implies that $\sum_{x\in\mathcal{E}_s} f(x) \neq \sum_{x\in\mathcal{O}_s} f(x)$.

$\Longleftarrow$ Let $f : \Omega \longrightarrow \mathbb{R}$ be a function that fulfills Equation (3.9). Let us consider a set of binary variables s such that $|s| = m$ and $\sum_{x\in\mathcal{E}_s} f(x) \neq \sum_{x\in\mathcal{O}_s} f(x)$. Because of Equality (3.8) of Lemma 7, for any non-empty subset of indexes $\{i_1, \ldots, i_j\} \subset s$, the coefficient $a_{i_1 \ldots i_j}$ appears the same number of times in the sums $\sum_{x\in\mathcal{E}_s} f(x)$ and $\sum_{x\in\mathcal{O}_s} f(x)$. Therefore, because the solutions such that $x_{i_1} = \cdots = x_{i_m} = 1$ only appear in $\mathcal{E}_s$, the only coefficient which causes $\sum_{x\in\mathcal{E}_s} f(x) \neq \sum_{x\in\mathcal{O}_s} f(x)$ is the coefficient $a_{i_1 \ldots i_m}$, which implies that $a_{i_1 \ldots i_m} \neq 0$ and consequently the function f is at least an m-degree pseudo-Boolean function.

In addition, by hypothesis, for any set of binary variables s such that $|s| > m$, the equality $\sum_{x\in\mathcal{E}_s} f(x) = \sum_{x\in\mathcal{O}_s} f(x)$ holds. Then, the solutions such that $x_{i_1} = \cdots = x_{i_{|s|}} = 1$ have no relevance in the sums and therefore $a_{i_1 \ldots i_{|s|}}$ must be a null coefficient. Consequently, f is an m-degree polynomial.

Finally, let us calculate the exact value of the sum $\sum_{x\in\mathcal{E}_s} f(x) = 2^{n-1}\bar{f}$. For a set of indexes $\{i_1, \ldots, i_j\}$, $1 \leq j \leq n$, the number of solutions such that $x_{i_1} = \cdots = x_{i_j} = 1$ is 2^{n-j}. Because f is an m-degree pseudo-Boolean function, for any subset of indexes $\{i_1, \ldots, i_j\}$ such that $m < j \leq n$, then $a_{i_i \ldots i_j} = 0$, which implies that Equation (3.10) is fulfilled. □

Notice that Equation 3.9 depends on the cardinality of s, not on the indexes of s. Theorem 1 shows all the conditions that any m-degree pseudo-Boolean function must fulfill. In addition, from Theorem 1, the following corollary is obtained.

Corollary 5. *Let f be an m-degree pseudo-Boolean function defined over $\{0,1\}^n$ $(m < n)$. For any subsets $s, s' \subseteq \{1, \ldots, n\}$ such that $|s|, |s'| \geq m + 1$, then the following holds,*

$$\sum_{x\in\mathcal{E}_s\cap\mathcal{O}_{s'}} f(x) = \sum_{x\in\mathcal{E}_{s'}\cap\mathcal{O}_s} f(x) \tag{3.11}$$

and

$$\sum_{x\in\mathcal{E}_s\cap\mathcal{E}_{s'}} f(x) = \sum_{x\in\mathcal{O}_s\cap\mathcal{O}_{s'}} f(x). \tag{3.12}$$

In addition, if $s \subset s'$, Equalities (3.11) *and* (3.12) *are rewritten respectively as*

$$\sum_{x\in\mathcal{E}_s\cap\mathcal{O}_{s'\setminus s}} f(x) = \sum_{x\in\mathcal{O}_s\cap\mathcal{O}_{s'\setminus s}} f(x) \tag{3.13}$$

and

$$\sum_{x\in\mathcal{E}_s\cap\mathcal{E}_{s'\setminus s}} f(x) = \sum_{x\in\mathcal{O}_s\cap\mathcal{E}_{s'\setminus s}} f(x). \tag{3.14}$$

Proof. By Theorem 1, for any subsets s, s' such that $|s|, |s'| \geq m+1$,

$$\sum_{x\in\mathcal{E}_s} f(x) = \sum_{x\in\mathcal{E}_{s'}} f(x) = \sum_{x\in\mathcal{O}_s} f(x) = \sum_{x\in\mathcal{O}_{s'}} f(x). \tag{3.15}$$

On the other hand, since for any subset s $\{\mathcal{E}_s, \mathcal{O}_s\}$ is a partition of Ω, we can decompose each summation:

$$\sum_{x\in\mathcal{E}_s} f(x) = \sum_{x\in\mathcal{E}_s\cap\mathcal{E}_{s'}} f(x) + \sum_{x\in\mathcal{E}_s\cap\mathcal{O}_{s'}} f(x). \tag{3.16}$$

Consequently,

$$\begin{aligned}\sum_{x\in\mathcal{E}_s} f(x) = \sum_{x\in\mathcal{E}_{s'}} f(x) &\iff \sum_{x\in\mathcal{E}_s\cap\mathcal{E}_{s'}} f(x) + \sum_{x\in\mathcal{E}_s\cap\mathcal{O}_{s'}} f(x) = \sum_{x\in\mathcal{E}_{s'}\cap\mathcal{E}_s} f(x) + \sum_{x\in\mathcal{E}_{s'}\cap\mathcal{O}_s} f(x)\\ &\iff \sum_{x\in\mathcal{E}_s\cap\mathcal{O}_{s'}} f(x) = \sum_{x\in\mathcal{E}_{s'}\cap\mathcal{O}_s} f(x).\end{aligned} \tag{3.17}$$

Equality (3.12) is analogously obtained:

$$\sum_{x\in\mathcal{E}_s} f(x) = \sum_{x\in\mathcal{O}_{s'}} f(x) \iff \sum_{x\in\mathcal{E}_s\cap\mathcal{E}_{s'}} f(x) = \sum_{x\in\mathcal{O}_s\cap\mathcal{O}_{s'}} f(x). \tag{3.18}$$

Finally, when $s \subset s'$:

- If $x \in \mathcal{E}_s \cap \mathcal{E}_{s'}$, then $x \in \mathcal{E}_s \cap \mathcal{E}_{s'\setminus s}$.
- If $x \in \mathcal{E}_s \cap \mathcal{O}_{s'}$, then $x \in \mathcal{E}_s \cap \mathcal{O}_{s'\setminus s}$.
- If $x \in \mathcal{O}_s \cap \mathcal{O}_{s'}$, then $x \in \mathcal{O}_s \cap \mathcal{E}_{s'\setminus s}$.
- If $x \in \mathcal{O}_s \cap \mathcal{E}_{s'}$, then $x \in \mathcal{O}_s \cap \mathcal{O}_{s'\setminus s}$.

Consequently,

$$\sum_{x\in\mathcal{E}_s\cap\mathcal{O}_{s'}} f(x) = \sum_{x\in\mathcal{E}_{s'}\cap\mathcal{O}_s} f(x) \iff \sum_{x\in\mathcal{E}_s\cap\mathcal{O}_{s'\setminus s}} f(x) = \sum_{x\in\mathcal{O}_s\cap\mathcal{O}_{s'\setminus s}} f(x) \tag{3.19}$$

and

$$\sum_{x\in\mathcal{E}_s\cap\mathcal{E}_{s'}} f(x) = \sum_{x\in\mathcal{O}_{s'}\cap\mathcal{O}_s} f(x) \iff \sum_{x\in\mathcal{E}_s\cap\mathcal{E}_{s'\setminus s}} f(x) = \sum_{x\in\mathcal{O}_s\cap\mathcal{E}_{s'\setminus s}} f(x). \tag{3.20}$$

□

Once Theorem 1 and Corollary 5 are presented, our next goal is to show that there exist rankings of solutions that cannot be generated by pseudo-Boolean functions of degree $m < n$.

3.3.2 Study of pseudo-Boolean functions of degree $m = n - 1$

Based on Theorem 1, several new results are obtained. The first result will prove that some rankings of solutions follow a pattern which implies that they do not fulfill the equalities of Equation (3.9) of Theorem 1 (and consequently cannot be generated by a $(n-1)$-degree pseudo-Boolean function or, equivalently, the ranking can only be generated by an n-degree pseudo-Boolean function). This specific result is enough to prove that, when $m < n$, m-degree pseudo-Boolean functions cannot generate all the possible rankings from the space of solutions[1].

To show that the pseudo-Boolean functions of degree $m < n$ cannot generate all the rankings of solutions, new definitions are required.

Definition 11. ***Word of a ranking.*** *Let f be a pseudo-Boolean function defined over $\{0,1\}^n$ and r_f the ranking generated by f. Let us denote by $r_f(i)$ the i-th solution of the ranking r_f. Then, we define the word of the ranking r_f, denoted by W_f, as the ordered list of length 2^n with the alphabet $\{E, O\}$ in the following way:*

$$W_f = \begin{bmatrix} w_1 \\ \vdots \\ w_{2^n} \end{bmatrix} \; s.t. \; w_i = \begin{cases} E, & \text{if } r_f(i) \text{ is an even solution} \\ O, & \text{if } r_f(i) \text{ is an odd solution.} \end{cases} \tag{3.21}$$

When a word is considered without a function f, we simplify the notation and remove the subindex f from W.

Example 5. Let us consider the fitness function $f(x) = x_1 - 3x_2 + 3x_3 - 2x_1x_2 + 7x_1x_3 - x_2x_3 + 11x_1x_2x_3$ and calculate the word of its ranking.

$$\begin{array}{c|cccccccc} x & 111 & 110 & 101 & 100 & 011 & 010 & 001 & 000 \\ \hline f(x) & 16 & -1 & 11 & 3 & -4 & -3 & 1 & 0 \end{array} \Longrightarrow r_f = \begin{bmatrix} 111 \\ 101 \\ 100 \\ 001 \\ 000 \\ 110 \\ 010 \\ 011 \end{bmatrix} \Longrightarrow W_f = \begin{bmatrix} E \\ O \\ E \\ E \\ O \\ O \\ E \\ O \end{bmatrix}. \tag{3.22}$$

Moreover, we extend the definition of the words of a ranking and present two new definitions.

Definition 12. ***Word of a ranking defined by s.*** *Let f be a pseudo-Boolean function defined over $\{0,1\}^n$, r_f the ranking generated by f and s a subset of binary variables. Let us denote by $r_f(i)$ the i-th solution of the ranking r_f. Then, we define* the word of the ranking r_f defined by *s, denoted by W_f^s, as the ordered list of length 2^n with the alphabet $\{E, O\}$ in the following way:*

[1] Note that any ranking of solutions generated by an m-degree pseudo-Boolean function can be generated by a pseudo-Boolean function of degree $m+1$ and, by induction, by a pseudo-Boolean function of degree $n \geq m$.

$$W_f^s = \begin{bmatrix} w_1^s \\ \vdots \\ w_{2^n}^s \end{bmatrix} \; s.t. \; w_i^s = \begin{cases} E, & \text{if } r_f(i) \text{ is an even solution defined by } s \\ O, & \text{if } r_f(i) \text{ is an odd solution defined by } s. \end{cases} \tag{3.23}$$

When a word defined by s is considered without a function f, we simplify the notation and remove the subindex f from W^s.

Definition 13. ***Word of a ranking with constraints*** C**.** *Let f be a pseudo-Boolean function defined over $\{0,1\}^n$ and C a set of constraints (specific bit values) defined over k bit values, $1 \leq k \leq n-1$. Let us denote by $f|_C$ the reduction of the function f to all the solutions that fulfill the constraints of C, and $r_{f|_C}$ the ranking generated by $f|_C$. Then, we define* the word of the ranking $r_{f|_C}$*, denoted by $W_{f|_C}$, as the ordered list of length 2^{n-k} with the alphabet $\{E, O\}$ in the following way:*

$$W_{f|_C} = \begin{bmatrix} w_1 \\ \vdots \\ w_{2^{n-k}} \end{bmatrix} \; s.t. \; w_i = \begin{cases} E, & \text{if } r_{f|_C}(i) \text{ is an even solution} \\ O, & \text{if } r_{f|_C}(i) \text{ is an odd solution.} \end{cases} \tag{3.24}$$

Example 6. Let us consider the fitness function f of Example 5 and $s = \{2, 3\}$. Then, the word of the ranking defined by s is:

$$r_f = \begin{bmatrix} 111 \\ 101 \\ 100 \\ 001 \\ 000 \\ 110 \\ 010 \\ 011 \end{bmatrix} \Longrightarrow W_f^s = \begin{bmatrix} E \\ O \\ O \\ E \\ E \\ E \\ O \\ O \end{bmatrix}. \tag{3.25}$$

On the other hand, if we are considering the ranking of solutions that satisfies the constraint C: $x_1 = 0$ (or, equivalently, the function $f|_{x_1=0}$), then the word of the ranking is:

$$r_{f|_C} = \begin{bmatrix} 100 \\ 000 \\ 110 \\ 010 \end{bmatrix} \Longrightarrow W_{f|_C} = \begin{bmatrix} E \\ O \\ O \\ E \end{bmatrix}. \tag{3.26}$$

Once we have defined the word of a ranking, we present a specific type of word: Dyck Words [24].

Definition 14. ***Dyck Word.*** *Let W be a word of length 2^n and Δ_i the difference between the number of E and O letters for the first i letters in a word W, $1 \leq i \leq 2^n$. Then, a word is a Dyck Word, with E (O) as dominant letter, if for any i, $\Delta_i \geq 0$ ($\Delta_i \leq 0$).*

The Catalan number $Cat_{2^{n-1}}$ is the number of possible Dyck Words of length 2^n with a fixed dominant letter, where $Cat_n = \binom{2n}{n}/(n+1)$.

In the literature, there exist a large number of articles about Dyck Words and equivalent definitions (such as Dyck paths) are also analyzed [29, 79].

Example 7. The word W_f from Example 5 is a Dyck Word with E as dominant letter.

$$r_f = \begin{bmatrix} 111 \\ 101 \\ 100 \\ 001 \\ 000 \\ 110 \\ 010 \\ 011 \end{bmatrix} \Longrightarrow W_f = \begin{bmatrix} E \\ O \\ E \\ E \\ O \\ O \\ E \\ O \end{bmatrix} \Longrightarrow \Delta = \begin{bmatrix} \Delta_1 \\ \Delta_2 \\ \Delta_3 \\ \Delta_4 \\ \Delta_5 \\ \Delta_6 \\ \Delta_7 \\ \Delta_8 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 2 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}. \tag{3.27}$$

Once we have defined Dyck Words, we present Proposition 1 which shows rankings that cannot be generated by a $(n-1)$-degree pseudo-Boolean function.

Proposition 1. *Let $n \geq 2$ and $\Omega = \{0,1\}^n$. Let r be a ranking of solutions from Ω and W the word generated by r. If W is a Dyck Word, then r cannot be generated by a $(n-1)$-degree pseudo-Boolean function.*

Proof. By reduction ad absurdum. Let r be a ranking generated by a $(n-1)$-degree pseudo-Boolean function f ($r_f = r$) and W a Dyck Word generated by r with E as dominant letter. By definition of the ranking of solutions and Dyck Words, we can group the solutions by 2^{n-1} different pairs of even-odd solutions, (x_e, x_o), such that $f(x_e) > f(x_o)$ for all pairs. Therefore, $\sum_{x\in\mathcal{E}} f(x) > \sum_{x\in\mathcal{O}} f(x)$ is deduced, which goes against Theorem 1 with $s = \{1,\ldots,n\}$. For Dyck Words with O as dominant letter, we obtain the opposite inequality.

□

Remark 3.1 *Due to Proposition 1 and Note 1, there exist rankings that cannot be generated by pseudo-Boolean functions of degree $m < n$.*

Example 8. Let $n = 3$ and f be the following fitness function:

$$f(x) = -16 + 33x_1 + 34x_2 + 36x_3 - 64x_1x_2 - 64x_1x_3 - 64x_2x_3 + 128x_1x_2x_3. \tag{3.28}$$

The function is similar to the well-known function BINVAL. By definition of f, it is obvious that $\sum_{x\in\mathcal{E}} f(x) > \sum_{x\in\mathcal{O}} f(x)$. So, by Proposition 1, the function f cannot be rewritten as a pseudo-Boolean function of degree 2 whose ranking of solutions is r_f.

Proposition 1 shows a necessary condition that any ranking must fulfill to have the possibility of being generated by a $(n-1)$-degree pseudo-Boolean function. Our next step is to check the "opposite direction" of Proposition 1: if the word W of a ranking r is not a Dyck Word, is it possible for r to be generated by a $(n-1)$-degree pseudo-Boolean function?

In order to shed some light on this issue, we first prove the result for $n = 3$. This is done by exhaustively verifying that any ranking of solutions r whose word is not a Dyck Word can be generated by a 2-degree pseudo-Boolean function f. Then, for $n > 3$, we conjecture that the result is true (notice that for $n = 4$, the study of $2^4! \approx 2 \cdot 10^{13}$ rankings is not computationally tractable) and, assuming that the conjecture is true, we give the exact number of rankings that cannot be generated by $(n-1)$-degree pseudo-Boolean functions.

Let us present the sufficient result of Proposition 1 when $n = 3$.

Proposition 2. *Let $n = 3$ and $\Omega = \{0,1\}^n$. Let r be a ranking of solutions from Ω and W the word generated by r. If W is not a Dyck Word, then there exists a 2-degree pseudo-Boolean function f whose generated ranking is r ($r_f = r$).*

Let us present a conjecture about the generalization of Proposition 2. For now on, we assume that the following conjecture is true.

Conjecture 1. Let $n \geq 2$ and $\Omega = \{0,1\}^n$. Let r be a ranking of solutions from Ω and W the word generated by r. If W is not a Dyck Word, then there exists a $(n-1)$-degree pseudo-Boolean function f whose generated ranking is r ($r_f = r$).

In Appendix A, we present two observations which could be helpful to prove Conjecture 1, which is an extension of Proposition 2. The presented observations can be extended for any $n \geq 4$ value and $(n-1)$-degree pseudo-Boolean functions. The first observation analyzes the coefficients of the 2-degree pseudo-Boolean functions and their impact on the generated ranking of solutions. The second observation studies the fitness function value of the solution 111 and specifies in which positions the solution "can be inserted" to generate a feasible ranking.

Assuming that Conjecture 1 is true, Proposition 1 and Conjecture 1 allow us to count the number of rankings of solutions that cannot be generated by $(n-1)$-degree pseudo-Boolean functions.

Corollary 6. *Let $n \geq 2$ and $\Omega = \{0,1\}^n$. Then, there are $\frac{2}{2^{n-1}+1} \cdot 2^n!$ rankings that cannot be generated by $(n-1)$-degree pseudo-Boolean functions.*

Proof. There are $Cat_{2^{n-1}}$ Dyck Words with E as dominant letter and the same number of Dyck Words with O as dominant letter. In addition, each E (O) letter of the Dyck Word corresponds to any even (odd) solution, which implies that there are $\left(2^{n-1}!\right)^2$ rankings that generate that particular Dyck Word. Consequently, the number of rankings that cannot be generated by $(n-1)$-degree pseudo-Boolean functions is

$$2 \cdot Cat_{2^{n-1}} \cdot \left(2^{n-1}!\right)^2 = 2 \cdot \frac{2^n!}{2^{n-1}!(2^{n-1}+1)!} \cdot \left(2^{n-1}!\right)^2 = \frac{2}{2^{n-1}+1} \cdot 2^n!. \tag{3.29}$$

□

Furthermore, because of Note 1, when $\Omega = \{0,1\}^n$, a ranking that cannot be generated by $(n-1)$-degree pseudo-Boolean functions is impossible to be generated by m-degree pseudo-Boolean functions, where $m < n-1$. Consequently, the previous number is also an upper bound of the number of rankings that cannot be generated by pseudo-Boolean functions of degree $m < n-1$. Note that, the proportion of rankings that can only be generated by n-degree pseudo-Boolean functions (functions which fulfill $a_{1\ldots n} \neq 0$) tends to 0 when n tends to infinity.

Example 9. For $n = 3$, the number of rankings that cannot be generated by 2-degree pseudo-Boolean functions is

$$\frac{2}{2^{n-1}+1} \cdot 2^n! = \frac{2}{5} \cdot 8! = 16128. \tag{3.30}$$

Consequently, for $n = 3$, there are exactly 24192 possible rankings that can be generated by 2-degree pseudo-Boolean functions out of 40320; that is, 60% of all the possible rankings.

3.3.3 Study of pseudo-Boolean functions of degree $m < n-1$

The presented results up to this point are based on Theorem 1 when $m = n-1$. Our next step is to generalize and study the case of Dyck Words for m-degree pseudo-Boolean functions, where $m < n-1$. This section extends Proposition 1 for any $n \geq 3$ and $m < n-1$. However, this extension shows the necessary condition for a ranking to be generated by a pseudo-Boolean function of degree $m < n-1$, not the sufficient condition.

First, a variation of Definition 14 is presented.

Definition 15. ***Dyck Word defined by*** s. *Let us consider* Δ_i^s *the difference between the number of* E *and* O *letters for the first* i *letters in a word* W^s, $1 \leq i \leq 2^n$. *Then, the word* W^s *is a Dyck Word, with* E *(*O*) as dominant letter, if for any* i, $\Delta_i^s \geq 0$ *(*$\Delta_i^s \leq 0$*).*

With Definition 15, we present an extension of Proposition 1.

Lemma 8. *Let* $n \geq 3$ *and* $\Omega = \{0,1\}^n$. *Let* r *be a ranking of solutions from* Ω, s *a set of binary variables such that* $n \geq |s| > m \geq 1$ *and* W^s *the word of the ranking* r *defined by* s. *If* W^s *is a Dyck Word, then* r *cannot be generated by an* m*-degree pseudo-Boolean function.*

Proof. The proof of the lemma is analogous to the proof of Proposition 1. □

Bear in mind that Lemma 8 does not focus on a specific set s. Therefore, for a ranking r_f, there might be more than one possible way to apply Lemma 8 and to prove that r_f cannot be generated by a pseudo-Boolean function of degree $m \leq n-1$.

In addition, using a similar argument of the proof of Proposition 1, Equations (3.11) and (3.12) from Corollary 5 can be also used to show new rankings that cannot be generated by m-degree pseudo-Boolean functions. For example, it is possible to define new groups of solutions (such as $G_1 = \mathcal{E}_s \cap \mathcal{O}_{s'}$ and $G_2 = \mathcal{O}_s \cap \mathcal{E}_{s'}$, or $G_1 = \mathcal{E}_s \cap \mathcal{E}_{s'}$ and $G_2 = \mathcal{O}_s \cap \mathcal{O}_{s'}$) and show several new rankings that cannot be generated by a pseudo-Boolean function of degree m.

Example 10. For $n = 4$, the ranking

$$r = [0111\ \ 0001\ \ 0010\ \ 1001\ \ 1010\ \ 1111\ \ 1100\ \ 0100\ \ 0000\ \ 1011\ \ 1000\ \ 0011\ \ 1101\ \ 0101\ \ 0110\ \ 1110]^T \quad (3.31)$$

cannot be generated by a 2-degree pseudo-Boolean function because for the set $s = \{1,2,3\}$

$$W^s = [E\ \ E\ \ E\ \ E\ \ E\ \ E\ \ E\ \ E\ \ O\ \ O\ \ O\ \ O\ \ O\ \ O\ \ O\ \ O]^T \quad (3.32)$$

is a Dyck Word.

Once Lemma 8 has been presented, the opposite research question is studied: for $n \geq 3$ and $m < n-1$, can we prove that any ranking which has no Dyck Words defined by all subset of variables s such that $|s| \geq m+1$ can be generated by an m-degree pseudo-Boolean function? In Example 11, a counterexample is presented.

Example 11. Let $n = 4$ and r be the following ranking:

$$r = [0100\ \ 0101\ \ 1000\ \ 1001\ \ 0000\ \ 1011\ \ 0011\ \ 1100\ \ 1010\ \ 1110\ \ 0111\ \ 0110\ \ 1101\ \ 0001\ \ 0010\ \ 1111]^T. \quad (3.33)$$

We will observe that: (a) for any set of variables s such that $|s| \geq 3$, W^s is not a Dyck Word; and (b) the ranking cannot be generated by a 2-degree pseudo-Boolean function.

(a) Let us calculate the words W^s defined by the sets s such that $|s| \geq 3$.

s	W^s
$\{1,2,3,4\}$	$[O\ E\ O\ E\ E\ O\ E\ E\ E\ O\ O\ E\ O\ O\ O\ E]^T$
$\{1,2,3\}$	$[E\ O\ O\ E\ O\ O\ O\ E\ E\ O\ E\ O\ O\ E\ E\ E]^T$
$\{1,2,4\}$	$[O\ E\ E\ O\ O\ E\ O\ E\ O\ O\ O\ E\ O\ E\ E\ E]^T$
$\{1,3,4\}$	$[E\ O\ E\ O\ O\ O\ E\ O\ E\ O\ O\ E\ E\ E\ O\ E]^T$
$\{2,3,4\}$	$[E\ E\ E\ E\ O\ O\ E\ O\ O\ E\ O\ O\ O\ O\ E\ E]^T$

(3.34)

Therefore, for any set s such that $|s| \geq 3$, the presented ranking has no Dyck Words defined by s.

(b) Let $s = \{1,2,3\}$ and $s' = \{1,2,4\}$. By Corollary 5, if a 2-degree pseudo-Boolean function can generate r, then

$$\sum_{x \in \mathcal{E}_s \cap \mathcal{E}_{s'}} f(x) = \sum_{x \in \mathcal{O}_s \cap \mathcal{O}_{s'}} f(x) \quad (3.35)$$

must be fulfilled. However,

$$\mathcal{E}_s \cap \mathcal{E}_{s'} = \{1100, 0001, 0010, 1111\} \text{ and } \mathcal{O}_s \cap \mathcal{O}_{s'} = \{0000, 0011, 1110, 1101\} \quad (3.36)$$

and, by definition of the ranking r,

$$f(0000) > f(1100),\ f(0011) > f(0001),\ f(1110) > f(0010) \text{ and } f(1101) > f(1111). \quad (3.37)$$

Consequently,

$$\sum_{x \in \mathcal{E}_s \cap \mathcal{E}_{s'}} f(x) < \sum_{x \in \mathcal{O}_s \cap \mathcal{O}_{s'}} f(x), \quad (3.38)$$

which implies that r cannot be generated by a 2-degree pseudo-Boolean function.

In addition, based on Corollary 5, the following result is obtained.

Corollary 7. *Let $n \geq 3$ and $j \in \{1,\ldots,n\}$. Let r be a ranking and $s = \{1,\ldots,n\}\backslash\{j\}$ a set of variables. Let $C : x_j = c_j$ be a constraint and $W_{f|_C}$ the word of the ranking generated by $f|_C$. If $W_{f|_C}$ is a Dyck Word, then r cannot be generated by a $(n-2)$-degree pseudo-Boolean function.*

Proof. Let $s' = \{1,\ldots,n\}$. Then, by Corollary 5,

$$\sum_{x\in\mathcal{E}_s\cap\mathcal{O}_{s'\setminus s}} f(x) = \sum_{x\in\mathcal{O}_s\cap\mathcal{O}_{s'\setminus s}} f(x) \quad \text{and} \quad \sum_{x\in\mathcal{E}_s\cap\mathcal{E}_{s'\setminus s}} f(x) = \sum_{x\in\mathcal{O}_s\cap\mathcal{E}_{s'\setminus s}} f(x)$$
$$\iff \sum_{x\in\mathcal{E}_s|x_j=0} f(x) = \sum_{x\in\mathcal{O}_s|x_j=0} f(x) \quad \text{and} \quad \sum_{x\in\mathcal{E}_s|x_j=1} f(x) = \sum_{x\in\mathcal{O}_s|x_j=1} f(x). \tag{3.39}$$

Consequently, if $W_{f|_C}$ is a Dyck Word (no matter if $x_j = 0$ or $x_j = 1$), one of the previous equalities is not fulfilled, which implies that r cannot be generated by a $(n-2)$-degree pseudo-Boolean function. □

Therefore, the study of Dyck Words allows us to recognize rankings that cannot be generated by any pseudo-Boolean function of degree $m < n-1$. The reason for not obtaining the opposite result of Lemma 8 (similar to the case of $n = 3$ with Proposition 2) is that, for all $|s| > m$, Lemma 8 studies each equality of Equation (3.9) of Theorem 1 independently, whereas Theorem 1 ensures that all the equalities of Equation (3.9) are fulfilled at the same time.

To remark the dissimilarities between Lemma 8 and Corollary 5, we present Example 12 for 3-dimensional linear pseudo-Boolean functions.

Example 12. Let $n = 3$ and $m = 1$. In this example, we present: (a) the number of rankings of solutions that cannot be discarded by Lemma 8 (the upper bound of the number of possible rankings that can be generated by linear pseudo-Boolean functions); and (b) the number of rankings of solutions that can be generated by linear pseudo-Boolean functions (counted by Corollary 5).

In the following table, we show if a solution is even or odd according to a set s.

s	111	110	101	100	011	010	001	000
$\{1,2,3\}$	E	O	O	E	O	E	E	O
$\{1,2\}$	E	O	O	E	E	O	O	E
$\{1,3\}$	E	O	E	O	O	E	O	E
$\{2,3\}$	E	E	O	O	O	O	E	E

(3.40)

(a) Lemma 8 (or, equivalently, the equalities of Equation (3.9) of Theorem 1) implies that any linear pseudo-Boolean function cannot have any Dyck Word over the sets s such that $|s| \geq 2$ (deduced from the following 4 equalities):

$$\begin{array}{l}(a.1)\ \sum_{x\in\mathcal{E}_{\{1,2,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2,3\}}} f(x);\\ (a.2)\ \sum_{x\in\mathcal{E}_{\{1,2\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2\}}} f(x);\\ (a.3)\ \sum_{x\in\mathcal{E}_{\{1,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,3\}}} f(x);\\ (a.4)\ \sum_{x\in\mathcal{E}_{\{2,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{2,3\}}} f(x).\end{array} \tag{3.41}$$

To bound the number of rankings, we have counted the rankings that generate a Dyck Word over one, two, three and four sets of variables ($\{1,2,3\}$, $\{1,2\}$,$\{1,3\}$ and $\{2,3\}$), and then we apply the inclusion-exclusion principle.

$$|\{\text{Rankings generated by linear pseudo-Boolean functions}\}| \leq \sum_{I\subseteq\{1,\ldots,4\}} (-1)^{|I|}\,|\cap_{i\in I} R'_i| =$$
$$= 40320 - 64512 + 30720 - 4032 = 2496, \tag{3.42}$$

where R'_i is the set of rankings that fulfills the equality $(a.i)$.

(b) Corollary 5 implies that any linear pseudo-Boolean function must fulfill all the following 12 equalities:

$$\begin{array}{ll}
(b.1) & \sum_{x\in\mathcal{E}_{\{1,2,3\}}\cap\mathcal{O}_{\{1,2\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2,3\}}\cap\mathcal{E}_{\{1,2\}}} f(x);\\
(b.2) & \sum_{x\in\mathcal{E}_{\{1,2,3\}}\cap\mathcal{E}_{\{1,2\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2,3\}}\cap\mathcal{O}_{\{1,2\}}} f(x);\\
(b.3) & \sum_{x\in\mathcal{E}_{\{1,2,3\}}\cap\mathcal{O}_{\{1,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2,3\}}\cap\mathcal{E}_{\{1,3\}}} f(x);\\
(b.4) & \sum_{x\in\mathcal{E}_{\{1,2,3\}}\cap\mathcal{E}_{\{1,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2,3\}}\cap\mathcal{O}_{\{1,3\}}} f(x);\\
(b.5) & \sum_{x\in\mathcal{E}_{\{1,2,3\}}\cap\mathcal{O}_{\{2,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2,3\}}\cap\mathcal{E}_{\{2,3\}}} f(x);\\
(b.6) & \sum_{x\in\mathcal{E}_{\{1,2,3\}}\cap\mathcal{E}_{\{2,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2,3\}}\cap\mathcal{O}_{\{2,3\}}} f(x);\\
(b.7) & \sum_{x\in\mathcal{E}_{\{1,2\}}\cap\mathcal{O}_{\{1,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2\}}\cap\mathcal{E}_{\{1,3\}}} f(x);\\
(b.8) & \sum_{x\in\mathcal{E}_{\{1,2\}}\cap\mathcal{E}_{\{1,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2\}}\cap\mathcal{O}_{\{1,3\}}} f(x);\\
(b.9) & \sum_{x\in\mathcal{E}_{\{1,2\}}\cap\mathcal{O}_{\{2,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2\}}\cap\mathcal{E}_{\{2,3\}}} f(x);\\
(b.10) & \sum_{x\in\mathcal{E}_{\{1,2\}}\cap\mathcal{E}_{\{2,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,2\}}\cap\mathcal{O}_{\{2,3\}}} f(x);\\
(b.11) & \sum_{x\in\mathcal{E}_{\{1,3\}}\cap\mathcal{O}_{\{2,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,3\}}\cap\mathcal{E}_{\{2,3\}}} f(x);\\
(b.12) & \sum_{x\in\mathcal{E}_{\{1,3\}}\cap\mathcal{E}_{\{2,3\}}} f(x) = \sum_{x\in\mathcal{O}_{\{1,3\}}\cap\mathcal{O}_{\{2,3\}}} f(x).
\end{array} \tag{3.43}$$

Similar to (a), we have counted the number of rankings that generate a Dyck Word over i sets of variables, $i = 1, \ldots, 12$, and then we apply the inclusion-exclusion principle to count the exact number of rankings that can be generated by linear pseudo-Boolean functions.

$$|\{\text{Rankings generated by linear pseudo-Boolean functions}\}| = \sum_{I\subseteq\{1,\ldots,12\}} (-1)^{|I|}\,|\cap_{i\in I} R_i| =$$

$$\begin{aligned}
=&40320 - 322560 + 1196160 - 2693664 + 4082640\\
&- 4368624 + 3368400 - 1875312 + 743376 - 203280 + 36288 - 3840 + 192\\
=&96,
\end{aligned} \tag{3.44}$$

where R_i is the set of rankings that fulfills the equality $(b.i)$.

Example 12 shows why Corollary 5 obtains the exact number rankings that can be generated by pseudo-Boolean functions of degree $m < n$ and shows why Lemma 8 does not. Therefore, Lemma 8 is not a sufficient condition to prove which rankings can be generated. A future work is to take advantage of the analysis of the words of the ranking and study the features that remain to achieve the sufficient condition.

3.4 Conclusions

In this chapter, we have presented a characterization of pseudo-Boolean functions of degree $m < n$, where n is the size of the search space, and the necessary conditions of a ranking of solutions to be generated by an m-degree pseudo-Boolean function. For the characterization, according to the parity of zeros of each solution (with respect to a set of variables), we have shown the equalities that any m-degree pseudo-Boolean function must fulfill. Based on those equalities, we present the word of a ranking (defined by a set of variables), we introduce Dyck Words and we present a necessary condition: if the word of a ranking defined by a set s

such that $|s| > m$ is a Dyck Word, then the ranking is impossible to be generated by an m-degree pseudo-Boolean function. On the other hand, we present a conjecture about the sufficient condition of a ranking to be generated by a $(n-1)$-degree pseudo-Boolean function. In Appendix A, we show two observations about Conjecture 1.

4

Generation and study of artificial instances

4.1 Introduction

In Chapter 3, it is shown and proved that there exist rankings of solutions that cannot be generated by a pseudo-Boolean function of degree $m < n$. Consequently, regarding rankings of solutions, the number of possible scenarios is fewer than $2^n!$. Because of that, our next step is to study those possible rankings and their main features.

As mentioned in Chapter 1, researchers choose real-world instances or they generate artificial instances in order to evaluate the performance of the proposed algorithm. In this chapter, we will focus on the latter group. To generate artificial instances, when there is no premeditated selection of the parameters of the problem to generate instances, uniform distributions are considered to fix the parameter values. The idea behind this procedure is that if we sample coefficients uniformly at random to generate instances, then all the feasible scenarios (and, equivalently, rankings of solutions) are generated uniformly as well. Unfortunately, this is not always true.

The main objective of the experiments presented in this chapter is not only to show that sampling coefficients uniformly at random generates "biased fitness functions" (in terms of frequency), but also to extract features and characteristics of the rankings of solutions generated by this process. The study of the features of the generated rankings will allow us to understand why some algorithms perform better in most of the instances of the studied problem.

To the best of our knowledge, there is one "initial" reference which is closely related to our analysis: [19]. In the mentioned article, the authors prove that, when the algorithm only considers the ranking of solutions to compare solutions, sampling in the space of coefficients uniformly at random is not equivalent to sampling instances in the space of functions uniformly at random. To do so, the authors consider the Linear Ordering Problem (and, briefly, the Quadratic Assignment Problem and the Permutation Flowshop Scheduling Problem) and analyze the instances generated by sampling coefficients uniformly at random. They observe that, from all the possible rankings that can be generated by the definition of the problem, there are some rankings which are sampled more frequently. Furthermore, the authors define a grouping of the rankings according to their frequency in the sample and they analyze the inequalities that each group of rankings induces. Based on that work, in [54] the authors count exactly how many rankings of solutions the Linear Ordering Problem and the Traveling Salesman Problem can exactly generate and which rankings of solutions can be obtained

by both problems. The authors of [19] and [54] illustrate their conclusions considering permutation-based COPs. In this chapter, the considered COPs to carry out a similar analysis are the UBQP and the NPP.

This chapter is organized as follows. In Section 4.2 the experimental results over the UBQP are shown and an analysis of the obtained results is discussed. In Section 4.3 the experiments for the NPP are detailed and analyzed. Finally, Section 4.4 concludes the chapter.

4.2 Experimental analysis of the rankings of the UBQP

In this section, we have conducted some experiments on the rankings of solutions generated by the UBQP (similar to the experiments carried out in [19]). The objectives of our experiments are to study those rankings in terms of their frequency when the coefficients of the UBQP matrix are sampled uniformly at random and to extract characteristics of them.

The experiments conducted in this section are done for the case $n = 3$. The main drawback of our experiments is that for $n \geq 4$ it is not computationally tractable. As mentioned in Chapter 3, note that when $n = 4$, the cardinality of the space of possible rankings of solutions is $2^4!$.

As observed in Example 9, the number of possible rankings of solutions generated by the UBQP is 24192. Considering that, we have generated a representative sample of instances of the UBQP by sampling the coefficients of the UBQP matrix uniformly at random. For $n = 3$, the UBQP requires 6 coefficients to describe an instance: $a_1, a_2, a_3, a_{12}, a_{13}, a_{23}$. The considered space to generate the coefficients of the UBQP matrix to generate the representative sample is the hypercube of dimension 6 centered at the origin: $[-0.5, 0.5]^6$ (to avoid the unbounded space $\mathbb{R}^6$). In order to have a representative sample of the rankings generated by the UBQP, initially we have generated 5 million rankings and then the sample size has been increasing by 1 million until all the possible 24192 rankings have been generated at least once. After generating a sample of 27 million rankings, all the rankings have been generated at least once. In Figure 4.1, we have ordered the 24192 rankings according to the number of times that each ranking has been generated.

In Table 4.1, a summary of the number of different rankings of solutions that have been sampled according to their frequencies in the sample are shown. In this table, we take into account the sample, order the rankings by their frequency, and observe the deciles: that is to say, how many of them represent $(10d)\%$ of the sample, for $d = 1, \ldots, 10$. For example, the 89 most frequent rankings of the generated sample represent approximately 2.7 million of the generated rankings (10%); the 220 most frequent rankings of the generated sample represent approximately 5.4 million of the generated rankings (20%); and so on. It is clear that a few rankings represent most of the sample, which clearly means that there are rankings which are more intriguing to analyze.

To see the main features of the most frequent rankings, we have focused on three characteristics: number of local optimal solutions with respect to the Hamming distance, the similarities of the rankings and the probability of occurrence.

Table 4.1: Number of different rankings of the UBQP ($n = 3$) ordered by their frequencies in the sample and the percentage they represent.

Size of the sample	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Number of different rankings	89	220	418	672	1006	1469	2221	3609	6621	24192

First, we have calculated the number of local optimal solutions of each ranking according to the Hamming distance. For the Hamming distance, two solutions are neighbors if the distance between them is 1. In Figure 4.1, the plot is divided in three figures, with different colors, according to the number of local optimal solutions.

We observe that the most frequent rankings have only one optimal solution, with significant difference with the rest of rankings. In the ordered list of rankings with respect to their occurrence in the sample, the most frequent ranking with two local optimal solutions is placed in the 607th position (its frequency in the sample is 9839). Furthermore, the most frequent ranking with three local optimal solutions is placed in the 6617th position (its frequency in the sample is 540). This is very intriguing knowing that, from all the possible rankings generated by the UBQP, most of the rankings have two optimal solutions. In Table 4.2, the number of possible rankings and sampled rankings are shown. It is clear that the distribution of the number of rankings with one, two or three local optimal solutions and the distribution of the sampled rankings with one, two or three local optimal solutions are completely different. For the UBQP, 6912 rankings have one local optimal solution (28.6% of the rankings), 15840 rankings have two optimal solutions (65.5% of the rankings) and the rest of rankings have three optimal solutions (5.9% of the rankings). But in the generated sample for the UBQP, the majority of the rankings generated by sampling coefficients uniformly at random have one local optimal solution (79.76%), 19.69% of the rankings have two local optimal solutions and very few have three local optimal solutions (0.542%).

Next, based on the generated sample of rankings, we have calculated the exact frequency/probability of generating a specific ranking of solutions by sampling the coefficients of the UBQP matrix uniformly at random. This measures exactly the "regions" in which each ranking is generated by the UBQP. To do so, we have calculated the hypervolume of each ranking in the defined hypercube $[-0.5, 0.5]^6$; that is to say, we calculate the regions of the hypercube in which all the points of a specific region generate a ranking r_f. So, we divide the hypercube in 24192 regions. Based on the previous results (Table 4.2), we expected in advance that the measures of each region of the hypercube would not be the same.

To calculate the hypervolume of each region, we have considered the system of inequalities that each ranking defines and calculate the implicit region defined by the system of inequalities and the hypercube $[-0.5, 0.5]^6$. Consequently, the hypervolume is obtained integrating 1 in the calculated region, and because the hypervolume of the hypercube is 1, the obtained result is also the probability of generating a specific ranking of solutions by sampling the coefficients of the UBQP matrix uniformly at random. However, even if this process is exact, it is worth mentioning that there have been some computational issues in the calculation of

Table 4.2: Number of the rankings generated by the UBQP for $n = 3$. Each row groups the rankings according to the number of local optimal solutions. In the second and third columns, the number and percentage of different rankings generated by the UBQP are shown. In the fourth and fifth columns, the number of sampled rankings are shown (from the 27M size sample). In the last column, the 95% confidence interval (CI) of the number of sampled rankings is shown.

	Rankings (24192)		Sampled rankings (27M)		
L. Opt. Sol.	#	%	#	%	95% CI
1	6912	28.57	21535236	79.760	(79.745, 79.775)
2	15840	65.48	5318316	19.697	(19.682, 19.712)
3	1440	5.95	146448	0.542	(0.540, 0.545)

Table 4.3: First 4 non-symmetric rankings with the largest hypervolume values generated by the UBQP for $n = 3$.

Ranking	Hypervolume
$f(111) > f(101) > f(011) > f(001) > f(100) > f(110) > f(000) > f(010)$	$0.0013237847 \sim 32a$
$f(111) > f(110) > f(011) > f(101) > f(010) > f(100) > f(001) > f(000)$	$0.0012966579 \sim 31a$
$f(110) > f(010) > f(000) > f(100) > f(001) > f(011) > f(111) > f(101)$	$0.0012152778 \sim 29a$
$f(111) > f(110) > f(101) > f(011) > f(010) > f(100) > f(001) > f(000)$	$0.0011013455 \sim 27a$

the exact hypervolumes of some rankings. We believe that the issues are due to the dimension of the space (6) and some particularly small regions. Therefore, sampling would provide an estimation of the hypervolume of each region generated by the UBQP.

A main observation of these hypervolumes is that there exists a symmetry of the rankings. Two rankings of solutions are symmetric and have the same hypervolume value if the difference between both rankings is a permutation of the bits (in other words, for all the solutions, permute the bits according to a rule) and/or a reversion of the ranking of solutions (in other words, the optimal solution in the first ranking is the worst solution in the second ranking, the second best solution in the first ranking is the second worst solution in the second ranking, and so on).

Example 13. The rankings

$$r_f = [111\ 101\ 011\ 001\ 100\ 110\ 000\ 010]^T \tag{4.1}$$

and

$$r_f = [001\ 000\ 011\ 010\ 100\ 101\ 110\ 111]^T \tag{4.2}$$

are symmetric rankings because r'_f is r_f after a reversion and the permutation of the bits (1 3 2) (explicitly, $x_3x_2x_1 \to x_1x_3x_2$):

$$\begin{bmatrix}111\\101\\011\\001\\100\\110\\000\\010\end{bmatrix} \overset{\text{Reversion}}{\Longrightarrow} \begin{bmatrix}010\\000\\110\\100\\001\\011\\101\\111\end{bmatrix} \overset{\text{Permutation}}{\underset{(1\ 3\ 2)}{\Longrightarrow}} \begin{bmatrix}001\\000\\011\\010\\100\\101\\110\\111\end{bmatrix}. \tag{4.3}$$

So, when $n = 3$, each ranking has $2 \times n! = 12$ rankings (including itself) which are symmetric and have the same hypervolume value. Therefore, the regions of the hypercube can be grouped in sets of 12 regions and there might be $24192/12 = 2016$ different hypervolume values. In Table 4.3, the first 4 non-symmetric rankings with the largest and different hypervolume values are shown.

Even if the obtained largest hypervolume values are very small, let us take into account that if all the rankings had the same probability to be sampled (or, equivalently, the assumption in question in [19] was true), the values would be $a = 1/24192 = 0.0000413770$. Consequently, there is a significant difference of the values. The 4 groups of rankings of Table 4.3 (the 48 symmetric rankings) represent almost 6% of the hypercube (in the case that the regions were uniform, the 4 groups of rankings would represent almost 0.2% of the hypercube).

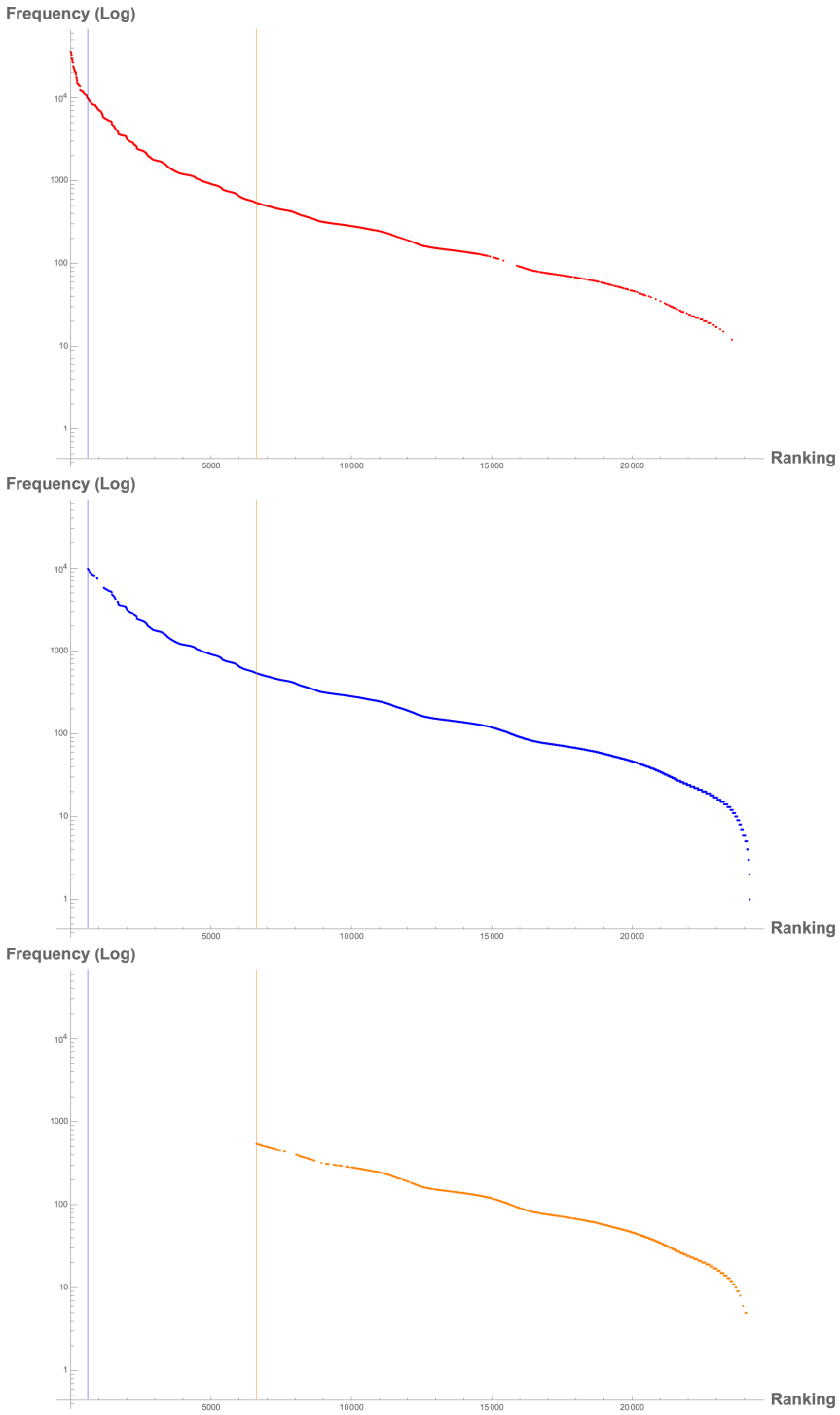


Fig. 4.1: Frequency of the 24192 rankings of the UBQP ($n = 3$) generated in a 27M size sample, in descending order of frequency (Y axis in logarithmic scale) and divided in three figures according to the number of local optimal solutions. Considering the Hamming distance 1, the top figure shows the rankings with one local optimal solution (the global optimum), colored in red. The center figure shows the rankings with two local optimal solutions, colored in blue. Finally, the bottom figure shows the rankings with three local optimal solutions, colored in orange. The colored vertical lines indicate the most frequent ranking with two and three optimal solutions.

4.3 Experimental analysis of the rankings of the NPP

In this section, we have conducted several experiments on the rankings generated by the NPP. First, we observe how many rankings are generated by the problem sampling integer numbers uniformly at random. Then, we study those rankings and analyze their features.

Because the NPP can be reformulated as a particular case of the UBQP, it is already known that the problem cannot generate all the rankings of solutions by sampling coefficients of the NPP uniformly at random (for any integer value n). Additionally, before starting with the experiments, it is necessary to elaborate "injective NPP instances" and local optimal solutions. By definition of the NPP, two opposite solutions have the same fitness function value: $f(x_n \dots x_1) = f((1-x_n) \dots (1-x_1))$, for any solution $x_n \dots x_1$. Hence, two assumptions have been considered: (i) we only consider the solutions such that $x_1 = 1$ for the rankings generated by the NPP, and (ii) we study instances of the NPP which are injective (that is to say, for any $Z' \subset Z$, Z' does not have a perfect partition). On the other hand, when we consider local optimal solutions (regarding the Hamming distance), we must consider the opposite solutions (the ones such that $x_1 = 0$) to make realistic conclusions. For example, the solutions 0011 and 1101 are neighbors because the Hamming distance between 0011 and 0010 (which is the opposite solution of 1101) is 1.

The experiments conducted to study the NPP are done for the cases $n \in \{3, 4, 5\}$. For the cases $n \in \{3, 4\}$, similar to the initial step of Section 4.2, first we have exhaustively counted how many rankings of solutions can be generated by instances of the NPP. To do so, we have analyzed whether or not a ranking of solutions generates a consistent system of inequalities (regarding the definition of the NPP, in which the number of coefficients is n). When $n = 3$, there are 4 solutions such that $x_1 = 1$ (which implies that the number of possible rankings is $4! = 24$) and the number of rankings generated by the NPP is 6. When $n = 4$, the number of rankings generated by the NPP is 168 (out of the $8! = 40320$ possible rankings). For the case $n = 5$, a different avenue has been followed because we have not been able to calculate in advance the exact number of different rankings that can be generated by the NPP.

4.3.1 Cases $n \in \{3, 4\}$

For $n \in \{3, 4\}$, to generate an instance, we sample n integer values from the set $\{1, \dots, 2^k\}$ uniformly at random, where $k \geq n$. We have tested the results for several k values and sample sizes, obtaining similar results. Because of that, we will only show the results for a sample whose size is 1M and $k = n + 2$. From all the sampled instances, we have only considered the injective rankings.

When $n = 3$, we have obtained the 6 rankings of solutions, and they follow a symmetry: from one ranking, the rest of rankings are obtained by permuting the bits. The main difference among the different samples generated for $n = 3$ is that when the value of k increases, the number of non-injective instances is reduced. Notwithstanding the value of k, the 6 rankings are generated uniformly. Consequently, in this particular case, sampling integers uniformly at random is equivalent to sampling NPP instances uniformly at random.

Nevertheless, when $n = 4$, we have obtained the 168 rankings of solutions. If we consider the symmetry of the rankings obtained by the permutation of the bits (from one ranking, we obtain $4! = 24$ symmetric rankings), there are 7 non-symmetric rankings. In Figure 4.2, we have ordered the 168 rankings according to the number of times that each ranking has been generated in the sample, and the plot is divided in three figures, with different colors, according to the number of local optimal solutions. In Table 4.4, the number of possible rankings and sampled rankings are shown.

It is obvious that, even if the NPP can be reformulated as a particular case of the UBQP, the obtained results for the NPP (Table 4.4) differs significantly from the results of the UBQP (Table 4.2). For the NPP, 96 (out of 168) rankings have one local optimal solution, 48 have two local optimal solutions and 24 have three local optimal solutions. Nevertheless, in the generated sample, nearly half of the sample is composed of rankings with one local optimal solution (49.54%), a quarter of rankings with two local optimal solutions (24.67&) and the remaining quarter of rankings with three local optimal solutions (25.8%).

To conclude the experiments of the case $n = 4$, we study the rankings generated by the NPP. A meaningful characteristic of our sample is that we identify 3 group of rankings according to the number of times that each ranking has been sampled. There are 24 rankings that each ranking has been sampled more than 8600 times; there are 72 rankings that each ranking has been sampled between 5500 and 6000 times; and the last 72 rankings have been sampled less than 2900 each. This is similar to the grouping that appears in [19], where the authors group the instances generated by the LOP in four classes labeled as "S rankings", "M rankings", "L rankings" and "XL rankings".

4.3.2 Case $n = 5$

First, to generate each ranking, we sample 5 integer values from the set $\{1, \ldots, 20000\}$ uniformly at random. Due to the symmetry of the rankings obtained by the permutation of bits, the total number of rankings generated by the NPP and the number of rankings generated by the NPP with l local optimal solutions must be divisible by $5! = 120$. Therefore, we have stopped increasing the sample when the total number of different rankings and the number of different rankings with l local optimal solutions (generated by the NPP) were divisible by 120. This scenario has been obtained with a sample of 5 million rankings, whose number of different rankings is 32760 (273 non-symmetric rankings). From all the sampled instances, we have only considered the injective rankings.

In Figure 4.3, we have ordered the 32760 rankings according to the number of times that each ranking has been generated, and the plot is divided in four figures, with different colors, according to the number of local optimal solutions. We can observe a similarity between the shapes of Figure 4.1 and Figure 4.3, but the local optimal distribution is completely different. The most frequent rankings have three local optimal solutions. Moreover, in the ordered list of rankings with respect to their occurrence in the sample, the most frequent ranking with one local optimal solution is placed in the 590th position (its frequency in the sample is 824);

Table 4.4: Number of the rankings generated by the NPP for $n = 4$. Each row groups the rankings according to the number of local optimal solutions. In the second and third columns, the number and percentage of different rankings generated by the NPP are shown. In the fourth and fifth columns, the number of sampled injective rankings are shown (from the 1M size sample). In the last column, the 95% confidence interval (CI) of the number of sampled rankings is shown.

	Rankings (168)		Sampled rankings (1000000)		
L. Opt. Sol.	#	%	#	%	95% CI
1	96	57.14	408296	49.54	(49.43,49.65)
2	48	28.57	203309	24.67	(24.58,24.76)
3	24	14.29	212613	25.8	(25.90,25.70)

the most frequent ranking with two local optimal solutions is placed in the 587th position (its frequency in the sample is 827); and, lastly, the most frequent ranking with four local optimal solutions is placed in the 1207th position (its frequency in the sample is 577). In addition, in Table 4.5, a summary of the number of rankings generated in the sample are shown. From all the different rankings generated by the NPP, 4800, 8160, 16800 and 3000 rankings have one, two, three and four local optimal solutions, respectively. On the other hand, the number of rankings generated in the sample with one, two, three and four local optimal solutions are 13.34%, 18.32%, 59.43% and 8.91%, respectively. If we compare the obtained results with the case $n = 4$ (Table 4.4), the most intriguing result is that the proportion of the number of different rankings and the rankings generated in the 5M sample (columns 3 and 5 of Table 4.5) seem more similar, although the differences are statistically significative (with respect to Pearson's chi-squared test).

Table 4.5: Number of the rankings generated by the NPP for $n = 5$. Each row groups the rankings according to the number of local optimal solutions. In the second and third columns, the number and percentage of different rankings generated by the NPP are shown. In the fourth and fifth columns, the number of sampled injective rankings are shown (from the 5M size sample). In the last column, the 95% confidence interval (CI) of the number of sampled rankings is shown.

	Rankings (32760)		Sampled rankings (5000000)		
L. Opt. Sol.	#	%	#	%	95% CI
1	4800	14.65	665803	13.34	(13.31,13.37)
2	8160	24.91	914194	18.32	(18.29,18.35)
3	16800	51.28	2965922	59.43	(59.39,59.48)
4	3000	9.16	444431	8.91	(8.88,8.93)

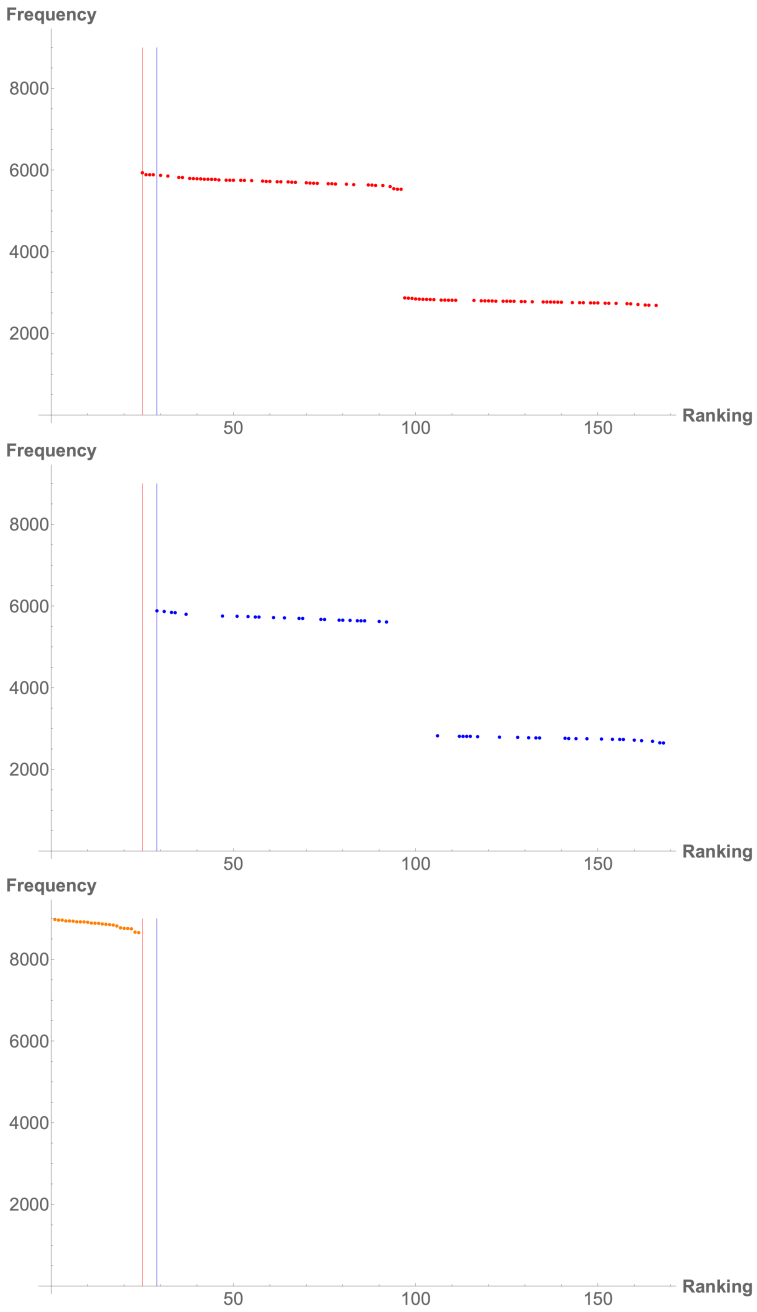


Fig. 4.2: Frequency of the 168 rankings of the NPP ($n = 4$) generated in a 1M size sample, in descending order of frequency and divided in three figures according to the number of local optimal solutions. Considering the Hamming distance 1, the top figure shows the rankings with one local optimal solution (the global optimum), colored in red. The center figure shows the rankings with two local optimal solutions, colored in blue. Finally, the bottom figure shows the rankings with three local optimal solutions, colored in orange. The colored vertical lines indicate the most frequent ranking with one and two optimal solutions.

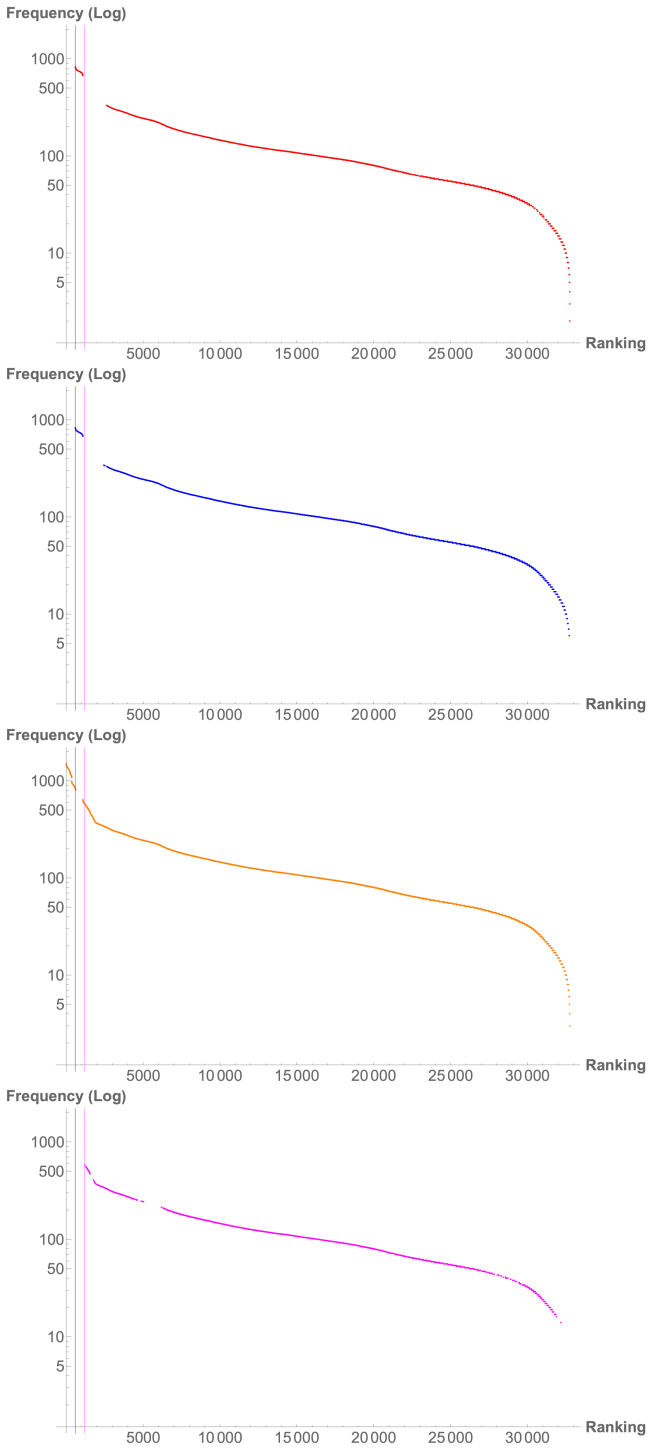


Fig. 4.3: Frequency of the 32760 rankings of the NPP ($n = 5$) generated in a 5M size sample, in descending order of frequency (Y axis in logarithmic scale) and divided in four figures according to the number of local optimal solutions. Considering the Hamming distance 1, the first figure shows the rankings with one local optimal solution (the global optimum), colored in red. The second figure shows the rankings with two local optimal solutions, colored in blue. The third figure shows the rankings with three local optimal solutions, colored in orange. Finally, the last figure shows the rankings with four local optimal solutions, colored in light magenta. The colored vertical lines indicate the most frequent ranking with one, two and four optimal solutions.

4.4 Conclusions

In this chapter, we have presented experimental analyses of the rankings generated by the UBQP for $n = 3$ and the NPP for $n \in \{3, 4, 5\}$. It is confirmed that sampling coefficients uniformly at random to generate instances of the problem does not always generate instances of the problem uniformly at random, at least in terms of the rankings generated. For example, whereas in the case of the NPP for $n = 3$ we have generated instances of the problem uniformly, in the cases of the UBQP for $n = 3$ and the NPP for $n \in \{4, 5\}$, the generated samples are biased. Furthermore, we have presented an analysis of the generated samples of rankings of solutions, studying several properties of them, such as the number of local optimal solutions and the probability of the occurrence.

Part II

Study of EDAs

5

A mathematical analysis of EDAs with distance-based exponential models

5.1 Introduction

Our second main motivation of this thesis is to explore permutation-based EDAs, similar to the works presented in the literature in the field of binary EDAs, to improve their understanding and performance and to solve COPs efficiently.

As mentioned in Chapter 1, theoretical studies can focus on many different objectives. For EDAs designed for binary-based COPs, the first theoretical studies focus on convergence analysis, whereas current research is often based on runtime analysis. Many mathematical frameworks have been presented in the literature to gain knowledge of binary EDAs. Nevertheless, permutation-based EDAs have not gained the same attention of researchers and there are no mathematical frameworks in the literature to study this kind of algorithms. Inspired by the path followed for the theoretical studies of binary EDAs, our first motivation is to generate a mathematical model that can be used to analyze permutation-based EDAs theoretically. Our proposed analysis over a permutation-based EDA focuses on convergence analysis. Convergence analysis is a very gripping starting point for original analyses to gain insights into the studied algorithms and to know in which scenarios the algorithm is guaranteed to converge to the optimal model by its design.

On the other hand, some permutation-based EDAs and applications that have been considered throughout this thesis are the following ones. In [103], the authors compare two permutation-based EDAs (Edge Histogram Based Sampling Algorithm and Node Histogram Based Sampling Algorithm), both designed by the authors, and their performances are compared over the Quadratic Assignment Problem and the Flowshop Scheduling Problem. In [18], the authors present an EDA in which, at each iteration, the algorithm learns a Mallows model. In addition, they present some experiments to solve the Permutation FlowShop Scheduling Problem to compare its performance with other permutation-based EDAs and they obtained better results in several cases. In [17], the authors expand the EDA used in [18] and they present the Generalized Mallows-EDA (GM-EDA). They also experiment with hybrid versions of GM-EDA and they present competitive results in comparison to other state-of-the-art algorithms. In [92], the authors consider the Generalized Mallows Distribution to design an EDA which is used to solve the Vehicle Routing Problem with Time Windows.

As we can observe, permutation-based EDAs have presented strong competitive results in the solution of some practical problems. However, in spite of the excellent results obtained by these algorithms, it is still not clear which mechanisms allow these algorithms to obtain these results. Furthermore, the advances in the design

of new EDAs for permutation-based COPs have not been complemented with their mathematical modeling due to the wide range of possible situations that the algorithm can involve. Because of that, our second motivation is to study the reasons and the characteristics of the used algorithms to achieve the presented results.

Considering both motivations, in this chapter, the convergence behavior of the Mallows-EDA has been studied. To do so, a mathematical modeling based on dynamical systems is presented to achieve our objective. Our first goal is to present a mathematical framework which allows the reproducibility of this study to different distance-based exponential models and new fitness functions. Then, we consider the presented framework to calculate the convergence behavior of the algorithm for several fitness functions. The studied functions are the constant function, the *needle in a haystack* (analogous to the definition presented in [95]) and the Mallows model. Our second goal is to carry out an analysis so as to provide new knowledge on the convergence behavior of permutation-based algorithms. Moreover, for the analyzed objective functions in this thesis, the obtained results are unexpected. We have observed that, for the scenarios in which the initial probability distribution is the uniform distribution or the fitness function is constant, the model converges to the optimal solution. However, in the rest of studied simple scenarios, the algorithm can converge to a degenerate distribution not necessarily centered at the optimal solution, or to a non-degenerate probability distribution. To determine the limit behavior of the algorithm, the equations to recognize the fixed points of the dynamical system are shown. These obtained results are dissimilar to the existing results in the literature for binary EDAs (for example, in [48, 49, 119], the studied algorithms converge to degenerate distributions centered at local optima or global optimum of the studied fitness function). Finally, our final goal is to present the obtained knowledge in this study to lay the basis for upcoming research in this area. As far as we know, our results are the first theoretical analysis given in the literature for permutation-based EDAs, and show the obstacles in achieving high quality theoretical results in this unexplored area. The presented analysis shows that, given an objective function, the initial probability distribution determines the limit behavior of the algorithm. Therefore, our first proposed algorithmic adaptation is to apply alternative initializations for obtaining high quality solutions. On the other hand, another proposed work is to analyze the expected number of iterations to achieve a high quality or optimal solution for the first time and connect it with the current tendency of the theoretical studies of EDAs.

This chapter is organized as follows. In Section 5.2, the basic concepts related with the Mallows model and our mathematical framework are introduced. In Section 5.3, the convergence behavior of the framework is studied for a constant objective function f. In Section 5.4, the function f analyzed is a *needle in a haystack* function. In Section 5.5, the function f analyzed is a Mallows model. In Sections 5.3, 5.4 and 5.5, two initial distributions are considered for the analysis: the uniform distribution and a Mallows probability distribution. Finally, in Section 5.6, conclusions are presented.

5.2 EDA based on Mallows models

5.2.1 Notation

The solutions of the studied optimization problems are permutations of length n. Let us denote by Σ_n the n-permutation space ($|\Sigma_n| = n! = N$) and $f : \Sigma_n \longrightarrow \mathbb{R}$ the function to maximize. Let us denote by σ a permutation from Σ_n or a solution of the function f. Throughout this chapter, $\sigma(i)$ represents the position

of the element i in the solution σ. Moreover, let us define an adjacent transposition of a permutation σ as a swap of two consecutive elements. Additionally, σ^{-1} is the inverse permutation of σ.

In Algorithm 2 (in Chapter 1), the general pseudocode of an EDA has been introduced. Still, there exists another possible interpretation of a step of an EDA in which probability distributions are considered as the main mathematical tool to study the algorithm. In this second description, the algorithm starts the iteration i from a probability distribution P_i and a population D_i is sampled. Then, D_i^S is selected and finally a new probability distribution is learnt for the next iteration, $P_i^L = P_{i+1}$. Throughout this study, the last description has been considered the main interpretation of EDAs for a better comprehension of Sections 5.2.2 and 5.2.4.

The probability distributions can be represented using probability vectors. Let us denote by $p_i(\sigma)$ the probability of σ under P_i. Therefore, we can denote by $P_i = (p_i(\sigma_1), \ldots, p_i(\sigma_N))$ the probability distribution of the population at iteration i. If we are studying EDAs with finite populations, the vector P_i can be considered as the "empirical probability mass function" of D_i (and analogous with P_i^S from the population D_i^S). We must emphasize that this representation of the populations by probability vectors is conceptual and it is really helpful for our proposed theoretical study, but it cannot be applied in practical EDAs due to the required memory. Moreover, the subscripts used for the permutations of the probability vectors distinguish the N permutations of Σ_n where an order has been set up. The space of possible probability vectors Ω_n is defined in the following way:

$$\Omega_n = \left\{ (p(\sigma_1), p(\sigma_2), \ldots, p(\sigma_N)) : \sum_{j=1}^{N} p(\sigma_j) = 1,\ 0 \leq p(\sigma_j) \leq 1,\ j = 1, \ldots, N \right\}. \tag{5.1}$$

To avoid the trivial case, it is assumed that any initial probability vector P_0 satisfies that $p_0(\sigma_j) < 1$, for $j = 1, \ldots, N$ (D_0 is not formed only by one specific solution). Note that Ω_n contains degenerate distributions. Let us denote by $1_{\sigma_k} = \{(p(\sigma_1), \ldots, p(\sigma_N)) \in \Omega_n \mid p(\sigma_k) = 1\}$ the degenerate probability distribution centered at σ_k.

Hence, if P_i are considered the references of each step of an EDA, then the EDA can be considered a sequence of probability distributions, each one given by a stochastic transition rule $\mathcal{G}$:

$$P_0 \underset{\mathcal{G}}{\longrightarrow} P_1 \underset{\mathcal{G}}{\longrightarrow} P_2 \underset{\mathcal{G}}{\longrightarrow} \cdots, \tag{5.2}$$

that is, $P_i = \mathcal{G}(P_{i-1}) = \mathcal{G}^i(P_0),\ \forall i \in \mathbb{N}$. Given a probability distribution P_i, the operator $\mathcal{G}$ outputs the probability distribution obtained after sequentially applying the sampling, the selection operator and the learning step. In this chapter, the considered algorithm to analyze is the Mallows-EDA [18] and the selection operator used throughout this analysis is a 2-tournament selection. The details are explained in Sections 5.2.3 and 5.2.4.

Hence, our objective is to study the convergence behavior described as follows:

$$\lim_{i \longrightarrow \infty} \mathcal{G}^i(P_0). \tag{5.3}$$

5.2.2 EDAs based on expectations

The application of the EDA schema to deal with optimization problems can involve an unapproachable variety of situations and behaviors. Due to this difficulty and following the ideas presented in the literature, our proposed mathematical modeling studies the expected probability distribution generated after one iteration of the algorithm. So, our proposed framework studies the deterministic function $G : \Omega_n \longrightarrow \Omega_n$ which assigns the expected probability distribution of the operator $\mathcal{G} : \Omega_n \longrightarrow \Omega_n$, similar to the idea followed in [48]:

$$P_{i+1} = G(P_i) = E[\mathcal{G}(P_i)] = E[(a \circ \phi)(P_i)] = \sum_{P \in \Omega_n} a(P) \cdot p(\phi(P_i) = P), \tag{5.4}$$

where $a(P)$ is the probability distribution obtained after applying the approximation step, ϕ is the selection operator and $p(\phi(P_i) = P)$ is the probability to obtain P from P_i. The details of our proposed selection operator and approximation step are explained in Section 5.2.4

Moreover, $P_i = G^i(P_0)$. Studying the expected probability distribution, each time the algorithm is applied, the deterministic operator G removes the random drift and avoids ending in a different probability distribution. Another equivalent interpretation of the deterministic operator G is the study of EDAs when the population size of D_i and D_i^S tends to infinity [31, 32, 107, 119]. By the Glivenko-Canteli theorem [25], when the population size tends to infinity, the empirical probability distribution of D_i and D_i^S converge to the underlying probability distribution of D_i and D_i^S, respectively. Under this assumption, P_i and P_i^S can be thought of as the population and the selected population at iteration i: in other words, P_i and P_i^S replace the populations D_i and D_i^S of the finite model, respectively. Therefore, our study can be thought of as the analysis of an EDA that works with the limit distributions of large populations. In Algorithm 3 the general pseudocode of an EDA based on expectations is shown.

Algorithm 3 General pseudocode of an EDA based on expectations

```
Obtain an initial probability distribution P_0
while No convergence do
  Compute the probability of selection from P_i by means of φ (selection operator): P_i^S
  Compute P_i^L to approximate P_i^S
  P_{i+1} = P_i^L
  i = i + 1
end while
Return Final probability distribution
```

Typical selection operators ϕ are n-tournament selection, proportional selection and truncation selection [11, 119].

Therefore, the operator G induces a deterministic sequence:

$$P_0 \underset{G}{\longrightarrow} P_1 \underset{G}{\longrightarrow} P_2 \longrightarrow \cdots \tag{5.5}$$

and the new objective is to study

$$\lim_{i \longrightarrow \infty} G^i(P_0). \tag{5.6}$$

In Section 5.2.4, the function G used to study the convergence behavior of the algorithm is defined.

5.2.3 Mallows model

The Mallows model [78] is a distanced-based exponential probability model over permutations. Under this model, the probability value of every permutation $\sigma \in \Sigma_n$ depends on two parameters: a central permutation σ_0 and a spread parameter θ. The Mallows model is defined as follows:

$$P(\sigma) = \frac{1}{\varphi(\theta, \sigma_0)} e^{-\theta d(\sigma, \sigma_0)}, \tag{5.7}$$

where d is an arbitrary distance function defined over the permutation space, $d(\sigma, \sigma_0)$ is the distance from σ to the central permutation σ_0, and $\varphi(\theta, \sigma_0) = \sum_{\sigma \in \Sigma_n} e^{-\theta d(\sigma, \sigma_0)}$ is the normalization constant. Due to the definition of the Mallows model, it is considered the analogous distribution of the Gaussian distribution over permutations. To simplify notation, let us denote by $\mathrm{MM}(\sigma_0, \theta)$ a Mallows probability distribution centered at σ_0 and with spread parameter θ. Bear in mind that when $\theta = 0$, $\mathrm{MM}(\sigma_0, 0)$ is a uniform probability distribution for any $\sigma_0 \in \Sigma_n$.

An important property of a Mallows model is that any two permutations at the same distance from the central permutation have the same probability value. Hence, we can group the permutations according to their distance to the central permutation.

Different distances can be used with the Mallows model, such as Cayley distance, Hamming distance or, the most used distance in the literature for the Mallows model, Kendall tau distance [61], which is the one we use in our EDA analysis.

Definition 16. *Kendall tau distance $d_\tau(\sigma, \pi)$ counts the number of pairwise disagreements between σ and π. It can be mathematically defined as follows:*

$$d_\tau(\sigma, \pi) = |\{(i, j) : i < j, (\sigma(i) < \sigma(j) \wedge \pi(i) > \pi(j)) \vee (\sigma(i) > \sigma(j) \wedge \pi(i) < \pi(j))\}| \tag{5.8}$$

where $\sigma(i)$ is the position of the element i in the permutation σ (and similarly with $\sigma(j)$, $\pi(i)$ and $\pi(j)$).

By definition, Σ_n with d_τ is a metric space. For simplification purposes, let us denote by $\sigma\pi$ the composition of σ and π (i.e., $\sigma\pi = \sigma \circ \pi$) and $d(\sigma, \pi)$ the Kendall tau distance between σ and π. According to the definition, the distance between two permutations is a non-negative integer between 0 and $D = n(n-1)/2 = \binom{n}{2}$. A property of Kendall tau distance is that, for any $\sigma, \pi \in \Sigma_n$, $d(\sigma, \pi) + d(\pi, I'\sigma) = d(\sigma, I'\sigma) = D$, where $I' = (n\ n-1\ \cdots 1)$. Consequently,

$$2 \sum_{\pi \in \Sigma_n} d(\sigma, \pi) = \sum_{\pi \in \Sigma_n} (d(\sigma, \pi) + d(\pi, I'\sigma)) = \sum_{\pi \in \Sigma_n} D = N \cdot D. \tag{5.9}$$

Another property is that Kendall tau distance has the right invariance property; that is, $d(\sigma, \pi) = d(\sigma\rho, \pi\rho)$ for every permutation $\sigma, \pi, \rho \in \Sigma_n$ [61]. Consequently, the normalization constant of the Mallows model can without loss of generality be written as $\varphi(\theta)$.

Kendall tau distance can be equivalently written as

$$d(\sigma, \pi) = \sum_{i=1}^{n-1} V_i(\sigma, \pi), \tag{5.10}$$

where $V_i(\sigma, \pi)$ is the minimum number of adjacent swaps to set the value $\pi(i)$ in the i-th position of σ [80]. It is worth noting that there exists a bijection between any permutation σ of Σ_n and the vector $(V_1(\sigma, I), \ldots, V_{n-1}(\sigma, I))$, where I represents the identity permutation and $V_i(\sigma, I) \in \{0, \ldots, n-i\}$, $\forall i = 1, \ldots, n-1$. Furthermore, the components $V_i(\sigma, I)$ are independent when σ is uniform on Σ_n.

Finally, with Kendall tau distance, the Mallows model with central permutation σ_0 and spread parameter θ and the Mallows model with central permutation $I'\sigma_0$ and spread parameter $-\theta$ are equivalent [39]. Therefore, without loss of generality, we assume that $\theta > 0$.

5.2.4 Mathematical modeling

As mentioned previously, in this section we present a mathematical framework to study the convergence behavior of a Mallows-EDA by a deterministic operator based on expectations. Before presenting our proposed mathematical modeling, we want to present how the Mallows-EDA is defined in [18].

The main characteristic of the Mallows-EDA is that the learned probability distribution is a Mallows probability distribution. To learn a Mallows model, σ_0 and θ parameters must be estimated. By the maximum likelihood estimation method, the exact parameters are calculated. The log-likelihood function for a finite population $\{\sigma_1, \ldots, \sigma_M\}$ is as follows [39]:

$$-M\theta \sum_{i=1}^{n-1} \bar{V}_i - M \log \varphi(\theta), \tag{5.11}$$

where $\bar{V}_i$ denotes the observed mean for V_i: $\bar{V}_i = \sum_{j=1}^{M} V_i(\sigma_j, \sigma_0)/M$. As we can observe in Equation (5.11), the value $-M\theta \sum_{i=1}^{n-1} \bar{V}_i$ depends on σ_0 and θ, whereas the value $-M \log \varphi(\theta)$ only depends on θ. Therefore, for a fixed non-negative value θ, maximizing the log-likelihood function is equivalent to minimizing $\sum_{i=1}^{n-1} \bar{V}_i$. This problem is also known as the rank aggregation problem and the Kemeny ranking problem and it is an NP-hard problem [4, 9]. This makes the theoretical analysis very complex.

Therefore, given a sample of M permutations $\{\sigma_1, \ldots, \sigma_M\}$, the first step to obtain the maximum likelihood estimators of the Mallows model is to obtain a permutation σ_0 which minimizes $\sum_{i=1}^{n-1} \bar{V}_i$. Let us denote by $\hat{\sigma}_0$ the estimated central permutation for the previous minimization problem. Once we obtain $\hat{\sigma}_0$, the maximum likelihood estimator of θ, denoted by $\hat{\theta}$, is obtained by solving the following equation [39]:

$$\sum_{i=1}^{n-1} \bar{V}_i = \frac{n-1}{e^{\theta} - 1} - \sum_{i=1}^{n-1} \frac{n-i+1}{e^{(n-i+1)\theta} - 1}. \tag{5.12}$$

Despite the fact that previous theoretical studies that use dynamical systems ([48, 118], for example) have closed formulae, the solution of this equation has not. For that reason, a numerical method such as, e.g. Newton-Raphson, has to be used to solve the equation. This is another reason that shows the complexity of the theoretical analysis. Once $\hat{\sigma}_0$ and $\hat{\theta}$ are estimated, the Mallows model is completely defined and it is used to sample new solutions for the next iteration of the algorithm. In Algorithm 4 the general pseudocode of Mallows-EDA defined in [18] is shown.

Algorithm 4 General pseudocode of Mallows-EDA

Obtain an initial population D_0
while Stop criteria = FALSE **do**
  Select a subset of individuals from the population D_i: D_i^S
  Estimate σ_0: $\hat{\sigma}_0$
  Estimate θ based on $\hat{\sigma}_0$: $\hat{\theta}$
  Sample a new set of individuals using MM($\hat{\sigma_0}, \hat{\theta}$): $D_{i+\frac{1}{2}}$
  Generate a new population D_{i+1} with D_i and $D_{i+\frac{1}{2}}$
  $i = i + 1$
end while
Return Best individual of the final population

Throughout this analysis, in order to study the convergence behavior of the Mallows-EDA based on expectations, the deterministic operator $G = a \circ \phi$ is used. This operator is a composition of the selection operator ϕ and the approximation step a used to learn the Mallows model. Hence, the operator ϕ returns the expected selection probability of the solutions from P_i and the function a uses the maximum likelihood estimation method to learn a Mallows model from P_i^S.

The selection operator studied in this work has been the widely used 2-tournament selection, but it is worth mentioning that the use of any selection operator based on rankings of solutions which satisfy impartiality and no degeneration properties defined in [32] will produce the same results. This selection operator is based on the ranking of solutions according to the objective function f and cannot assign extreme probabilities. Given the probability distribution P_i at iteration i and assuming a maximization problem, the expected probability of selecting a solution σ is the sum of all the binary selections in which σ and a solution π with a lower or equal fitness function value has been chosen, that is:

$$p_i^S(\sigma) = 2 \sum_{\pi | f(\sigma) > f(\pi)} p_i(\sigma) p_i(\pi) + \sum_{\pi | f(\sigma) = f(\pi)} p_i(\sigma) p_i(\pi). \tag{5.13}$$

Once we have P_i^S calculated, the function a deals with the probabilities $p_i^S(\sigma)$ to learn a new Mallows model which is the probability distribution of the next generation. In order to work with the probability vectors and the expected probability distributions and to estimate σ_0 and θ, Equations (5.11) and (5.12) must be reformulated. To do so, the value $\bar{V}_i$ is calculated using $p^S(\sigma)$ as the proportion of the solution σ in the selected population by the weighted average value of $V_i(\sigma, \sigma_0)$. So, we have

$$\bar{V}_i = \sum_{\sigma \in \Sigma_n} V_i(\sigma, \sigma_0) \cdot p^S(\sigma). \tag{5.14}$$

Therefore,

$$\sum_{i=1}^{n-1} \bar{V}_i = \sum_{i=1}^{n-1} \sum_{\sigma \in \Sigma_n} V_i(\sigma, \sigma_0) \cdot p^S(\sigma) = \sum_{\sigma \in \Sigma_n} d(\sigma, \sigma_0) \cdot p^S(\sigma). \tag{5.15}$$

So the maximum likelihood estimator of σ_0 from the expected selected population is the following:

$$\hat{\sigma}_0 = \arg\min_{\sigma\in\Sigma_n} \sum_{\pi\in\Sigma_n} d(\pi,\sigma)\cdot p^S(\pi). \tag{5.16}$$

The maximum likelihood estimator of σ_0 might not be unique. In Sections 5.4 and 5.5, we will observe some P^S probability distributions in which the estimated central permutation is not unique.

To estimate θ, we can use Equation (5.12) in the same way as with finite populations and solve the following equation:

$$\sum_{\sigma\in\Sigma_n} d(\sigma,\hat{\sigma}_0)\cdot p^S(\sigma) = \frac{n-1}{e^\theta-1} - \sum_{i=1}^{n-1}\frac{n-i+1}{e^{(n-i+1)\theta}-1}. \tag{5.17}$$

Throughout this study, two observations related to the estimation of the spread parameter are considered. Firstly, the right-hand side of Equation (5.17) is not defined when $\theta = 0$. Still, the right-hand side of Equation (5.17) tends to $\binom{n}{2}/2$ when θ tends to 0 and $\theta = 0$ is a removable singularity (see proof in Proposition 3 of Appendix B).

Considering this observation, the following lemma proves that when the estimated central permutation is unique, then the estimated spread parameter has a positive value. It is worth mentioning that Lemma 9 is independent of the objective function f and the iteration i of the algorithm.

Lemma 9. *Let P_i be a Mallows probability distribution with central permutation σ_0 and spread parameter $\theta \geq 0$, and P_i^S the probability distribution after a 2-tournament selection over P_i. Let $\hat{\sigma}_0$ be the unique estimator of the central permutation of P_{i+1}. Then, the value $\hat{\theta}$ which solves the following equation*

$$\sum_{\sigma\in\Sigma_n} d(\sigma,\hat{\sigma}_0)\cdot p^S(\sigma) = \frac{n-1}{e^{\hat{\theta}}-1} - \sum_{i=1}^{n-1}\frac{n-i+1}{e^{(n-i+1)\hat{\theta}}-1} \tag{5.18}$$

is a positive value. Equivalently, $\sum_{\sigma\in\Sigma_n} d(\sigma,\hat{\sigma}_0)\cdot p^S(\sigma)$ is a value lower than $\binom{n}{2}/2$.

Proof. First, let us consider the function g:

$$g(\theta) = \begin{cases} \frac{n-1}{e^\theta-1} - \sum_{i=1}^{n-1}\frac{n-i+1}{e^{(n-i+1)\theta}-1}, & \text{if } \theta \neq 0 \\ \frac{1}{2}\binom{n}{2}, & \text{if } \theta = 0. \end{cases} \tag{5.19}$$

The function g is a continuous decreasing function such that $g(\theta) + g(-\theta) = \binom{n}{2}$, $\lim_{\theta\longrightarrow-\infty} g(\theta) = \binom{n}{2}$ and $\lim_{\theta\longrightarrow\infty} g(\theta) = 0$ (see proof in Proposition 4 of Appendix B).

Secondly, for any $\hat{\sigma}_0$ and $\hat{\theta}$ parameters, $\sum_{\sigma\in\Sigma_n} d(\sigma,\hat{\sigma}_0)\cdot p^S(\sigma)$ is a value from the interval $(0, \binom{n}{2})$. In particular,

$$\sum_{\sigma\in\Sigma_n} d(\sigma,\hat{\sigma}_0)\cdot p^S(\sigma) + \sum_{\sigma\in\Sigma_n} d(\sigma,I'\hat{\sigma}_0)\cdot p^S(\sigma) = \binom{n}{2}\sum_{\sigma\in\Sigma_n} p^S(\sigma) = \binom{n}{2}. \tag{5.20}$$

Considering that, by hypothesis, $\hat{\sigma}_0$ is the unique estimator of the central permutation of P_{i+1},

$$\sum_{\sigma\in\Sigma_n} d(\sigma,\hat{\sigma}_0)\cdot p^S(\sigma) < \sum_{\sigma\in\Sigma_n} d(\sigma,I'\hat{\sigma}_0)\cdot p^S(\sigma) \tag{5.21}$$

is obtained and therefore

$$\sum_{\sigma \in \Sigma_n} d(\sigma, \hat{\sigma}_0) \cdot p^S(\sigma) < \frac{1}{2}\binom{n}{2}. \tag{5.22}$$

□

The second observation is that in the approximation step of our algorithm, at any iteration, if P^S is a Mallows model with central permutation σ_0 and spread parameter θ, then the learned Mallows model is the same one: $\hat{\sigma}_0 = \sigma_0$ and $\hat{\theta} = \theta$. The argument to prove this observation is that the probabilities of the solutions are ordered inversely according to their distance to σ_0. Hence, Equation (5.16) obtains the minimum value at σ_0 and it is unique. Furthermore, when $\hat{\theta} = \theta$, Equation (5.17) is fulfilled because P^S is a Mallows model. Another way to understand this observation is that when we work with infinite population and the sampling step is not needed, the probability distribution is kept constant. To simplify notation, let us consider the uniform distribution as a Mallows model with central permutation $\sigma_0 \in \Sigma_n$ and spread parameter 0.

In addition, it is assumed that the algorithm learns 1_{σ_k} probability distribution if $P^S = 1_{\sigma_k}$. Note that 1_{σ_k} is obtained as the limit distribution of $\mathrm{MM}(\sigma_k, \theta)$ when θ tends to infinity.

Once we have defined the selection operator and how we learn a new probability distribution, our operator G is defined. The schema of one iteration of the algorithm is the following:

$$\cdots \longrightarrow P_i \underbrace{\overset{\phi}{\longrightarrow} P_i^S \overset{a}{\longrightarrow} P_{i+1}}_{G = a \circ \phi} \longrightarrow \cdots, \tag{5.23}$$

where ϕ is 2-tournament selection and a is the approximation step that learns a Mallows probability distribution by the maximum likelihood estimation method.

The aim of the following sections is to apply our proposed mathematical modeling in some scenarios. Each scenario considers an objective function f and an initial probability distribution P_0. Our objective is to calculate $G^i(P_0)$ when i tends to infinity. To do so, $G^i(P_0)$ are calculated, for $i = 1, 2, 3, \ldots$, and the results are analyzed. In some particular cases, it is enough to calculate $G(P_0)$ to induce the limit behavior of the algorithm. For the most difficult cases, we study the fixed points of the algorithm and their attraction behavior, following the same idea used in the literature as in [48], among others.

In order to simplify the analysis and to present the tools and methods used to achieve our objectives, in this study we have considered three specific cases for the objective function. In Section 5.3, f is a constant function; in Section 5.4, f is a *needle in a haystack* function; and in Section 5.5, f is defined by a Mallows model. Objective functions such as the constant function and the *needle in a haystack* function have been used in many studies of different algorithms in the literature, whereas the Mallows model has been studied as an example of a unimodal objective function and to analyze the relation among the learned Mallows probability distributions by our dynamical system and the objective function. For these cases, we have considered P_0 as a uniform distribution or a Mallows model.

5.3 Limiting behavior for a constant function

In this first scenario, the function f to optimize is constant: $f(\sigma) = c,\ \forall \sigma \in \Sigma_n$. Hence, any solution can be considered as a global optimum. In this situation, it is proved that the algorithm keeps the initial probability distribution forever. We can summarize all the results from this section in Theorem 2.

Theorem 2. *If f is a constant function and P a Mallows probability distribution, then $G(P) = P$.*

Proof. Starting from any Mallows model $\mathrm{MM}(\sigma_0, \theta)$, let us observe the first iteration of the algorithm and calculate $G(P)$. It is proved that the selection method keeps the same distribution, and then the learned parameters are σ_0 and θ.

When f is a constant function, all the solutions are global optima. So, the selection probability of each solution is the same as the initial probability:

$$p^S(\sigma) = p(\sigma), \forall \sigma \in \Sigma_n \Longrightarrow P^S = P. \tag{5.24}$$

Given that $P^S = P$, the next step of the algorithm is to estimate the parameters to learn a Mallows model from P. By the observation from Section 5.2.4 about the estimation of the parameters from a Mallows model, it is deduced that $\hat{\sigma}_0 = \sigma_0$ and $\hat{\theta} = \theta$. Consequently, it is proved that when f is a constant function, $G(P) = P$ for any Mallows distribution P. □

5.4 Limiting behavior for a *needle in a haystack* function

In the next case, f is a *needle in the haystack* function centered at σ^*; the function is constant except for one solution σ^*, which is the optimal solution. Let us define

$$f(\sigma) = \begin{cases} c', \ \sigma = \sigma^* \\ c, \ \ \sigma \neq \sigma^* \end{cases} \tag{5.25}$$

such that $c' > c$.

In this section, the analysis focuses on the evolution and the convergence behavior of the algorithm when the fitness function can only take two possible values, one value for the optimal solution and the second value for any other solution. The analysis has been separated into three sections. In Section 5.4.1, the case when P_0 is a uniform distribution is considered. In this particular case, the main procedure of the algorithm is shown and some general results are explained. As a result of this analysis, the case when P_0 is a Mallows model centered at σ^* is analyzed, which is mentioned in Section 5.4.2. Finally, in Section 5.4.3, P_0 is a Mallows model centered at $\sigma_0 \neq \sigma^*$. In this case, a general observation among the rest of Mallows models is explained. To do so, the fixed points of the algorithm are calculated.

5.4.1 P_0 a uniform initial probability distribution

In this section, it is proved that when the initial probability distribution is a Mallows distribution centered at the optimal solution of the *needle in the haystack* function, the algorithm converges to the degenerate distribution centered at the optimum. The obtained result in this section can be summarized in the following lemma.

Lemma 10. *Let f be a* needle in a haystack *function centered at σ^* and P_0 a Mallows model with central permutation σ^* and spread parameter $\theta_0 \geq 0$. Then, the proposed EDA always converges to the degenerate distribution centered at σ^*.*

Proof. Let us start the demonstration from the case that P_0 is a uniform distribution. In order to calculate the limit behavior of the algorithm, let us start by calculating $G(P_0)$, starting from the computation of P_0^S. In this case, there are two different cases to analyze in the selection step. If σ^* is chosen to take part in the tournament, then it has an equal or higher function value than any other permutation, so σ^* is always selected. For the permutations $\sigma \neq \sigma^*$, they behave in the same way as when f is a constant function. So the probability after selection is as follows:

$$p_0^S(\sigma) = \begin{cases} p_0(\sigma)(2 - p_0(\sigma)), & \text{if } \sigma = \sigma^* \\ p_0(\sigma)(1 - p_0(\sigma^*)), & \text{if } \sigma \neq \sigma^*. \end{cases} \tag{5.26}$$

This same argument can be used for any iteration of the algorithm for the selection operator.

After the selection probability has been computed, let us study the estimation of the parameters for the Mallows models. Let us start with the estimation of the central permutation in different iterations of the algorithm, and after that, the estimated spread parameters.

At the first iteration of Algorithm 3, in order to calculate $\hat{\sigma}_0$ for P_1, it is necessary to calculate the solution of Equation (5.16) using P_0^S. Bear in mind that for any $\sigma \neq \sigma^*$,

$$\sum_{\pi \in \Sigma_n \setminus \{\sigma, \sigma^*\}} d(\pi, \sigma) \cdot p_0^S(\pi) = \sum_{\pi \in \Sigma_n \setminus \{\sigma, \sigma^*\}} d(\pi, \sigma^*) \cdot p_0^S(\pi) \tag{5.27}$$

because the selection probabilities for all the permutations except σ^* are the same, and the right invariance property over the Kendall tau distance ensures that the number of solutions at each distance is the same: that is, for a fixed $d \in \{0, \ldots, D\}$, $|\{\pi \in \Sigma_n : d(\pi, \sigma) = d\}|$ is constant for any $\sigma \in \Sigma_n$ (see Definition 17).

Let $\sigma \neq \sigma^*$. Thus, $d(\sigma, \sigma^*) = d > 0$. Therefore, considering Equation (5.27) and $p_0^S(\sigma^*) > p_0^S(\sigma)$,

$$\begin{aligned} \sum_{\pi \in \Sigma_n} d(\pi, \sigma) \cdot p_0^S(\pi) &= \sum_{\pi \in \Sigma_n \setminus \{\sigma, \sigma^*\}} d(\pi, \sigma) \cdot p_0^S(\pi) + d \cdot p_0^S(\sigma^*) + 0 \cdot p_0^S(\sigma) \\ &> \sum_{\pi \in \Sigma_n \setminus \{\sigma, \sigma^*\}} d(\pi, \sigma^*) \cdot p_0^S(\pi) + d \cdot p_0^S(\sigma) + 0 \cdot p_0^S(\sigma^*) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p_0^S(\pi), \end{aligned} \tag{5.28}$$

and it proves that the maximum likelihood estimator of the central permutation is σ^*.

So P_1 is a Mallows model with central permutation σ^*. Because of the uniqueness of the estimated central permutation and by Lemma 9, the estimated spread parameter of P_1 is a positive value. In order to generalize the obtained results to any iteration of the algorithm, let us calculate the central permutation of P_2. To determine P_1^S, we consider Equation (5.26) from P_1. Accordingly, for each solution, the lower the distance to σ^*, the higher the probability of selecting the solution is. Therefore, to calculate P_2, we can repeat the same argument of Section 5.3 to prove that $\hat{\sigma}_0 = \sigma^*$. This same argument can be repeated for any iteration $i > 2$.

Once it has been proved that σ^* is the estimated central permutation for the learned Mallows model at any iteration of the algorithm, let us study the estimation of θ. As mentioned previously, there is no closed formula for the solution of Equation (5.17). Hence, instead of calculating the value of θ, we follow a different avenue to prove the limiting behavior of the algorithm. Knowing by Lemma 9 that the estimated spread parameter $\hat{\theta}$ at any iteration of the algorithm is positive, we prove that the estimated spread parameter increases in two consecutive iterations.

Particularly, Equation (5.17) is analyzed to see if the spread parameter at iteration $i+1$ is a higher or lower value than the spread parameter at iteration i. To this end, two consecutive iterations are considered and the difference between $\sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_i^S(\sigma)$ and $\sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_{i+1}^S(\sigma)$ is analyzed. Without loss of generality, let us analyze the relation when $i = 0$.

The difference between the values of the left-hand side of (5.17) depends on the values $p_0^S(\sigma)$ and $p_1^S(\sigma)$, $\forall\sigma \in \Sigma_n$. Firstly, remember that $\sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_0^S(\sigma)$ was used to calculate the spread parameter of the Mallows probability distribution P_1. Hence, by the definition of the operator a it holds that $\sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_0^S(\sigma)$ and $\sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_1(\sigma)$ are the same value (this argument has been used to specify that the estimated parameters of a Mallows model to learn a new Mallows model are the same). Let us denote $C = \sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_1(\sigma)$ and compare it with $\sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_1^S(\sigma)$. Using Equation (5.26) for P_1,

$$\begin{aligned}\sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_1^S(\sigma) &= \sum_{\sigma\in\Sigma_n\setminus\{\sigma^*\}} d(\sigma,\sigma^*)\cdot p_1^S(\sigma)\\ &\overset{(5.26)}{=} \sum_{\sigma\in\Sigma_n\setminus\{\sigma^*\}} d(\sigma,\sigma^*)\cdot p_1(\sigma)\left(1-p_1(\sigma^*)\right) = C\left(1-p_1(\sigma^*)\right) < C. \end{aligned} \tag{5.29}$$

This implies that the left-hand side of Equation (5.17) decreases in two consecutive iterations. Hence, as the function g defined in Equation (5.19) is a strictly decreasing function over θ, the spread parameter increases after one iteration of the algorithm. So, θ_2 is a higher value than θ_1.

Using the same reasoning for any iteration, we can observe that at each iteration $p(\sigma^*)$ increases, whereas for all $\sigma \neq \sigma^*$ $p(\sigma)$ decreases. Moreover,

$$\sum_{\sigma\in\Sigma_n} d(\sigma,\sigma^*)\cdot p_j^S(\sigma) = C\left(1-p_1(\sigma^*)\right)\left(1-p_2(\sigma^*)\right)\cdots\left(1-p_j(\sigma^*)\right) < C\left(1-p_1(\sigma^*)\right)^j \underset{j\to\infty}{\longrightarrow} 0. \tag{5.30}$$

Consequently, θ tends to infinity when the number of iterations increases.

Therefore, after applying our modeling departing from a uniform distribution to a *needle in a haystack* function, the algorithm converges to a Mallows model with central permutation σ^* and spread parameter θ which tends to infinity. Hence, the distribution in the limit is the degenerate distribution centered at σ^*. □

5.4.2 P_0 a Mallows probability distribution with central permutation σ^* and spread parameter θ_0

This case is the same as the one in Section 5.4.1 after the first iteration. Hence, the algorithm converges to a degenerate distribution centered at σ^*.

5.4.3 P_0 a Mallows probability distribution with central permutation σ_0, where $d(\sigma^*, \sigma_0) = d^* \geq 1$, and spread parameter θ_0

Due to the difficulty of this case in comparison with the previous ones, the analysis of the convergence behavior of the algorithm is made from a new point of view. In this section, our objectives are to study the possible fixed points of the algorithm and to analyze the behavior of our dynamical system. A probability distribution is a fixed point of the algorithm if, after one iteration, the algorithm does not estimate a different probability distribution: that is to say, $G(P) = P$. Consequently, the algorithm will always estimate the same probability distribution.

In Section 5.4.3, the following proof idea is used:

i) In Section 5.4.3.1, the fixed points of the algorithm are calculated.
   - First, it is proved that any degenerate distribution is a fixed point.
   - Then, non-degenerate fixed points are calculated.

ii) In Section 5.4.3.2, the attraction of the fixed points is studied.

iii) Finally, in Section 5.4.3.3, the performance of the algorithm is analyzed for different initial probability distributions P_0.

5.4.3.1 Computation of the fixed points

For our first aim of Section 5.4.3, let us calculate the fixed points of our dynamical system G. First, let us realize that any degenerate distribution is a fixed point of the discrete dynamical system G. The selection probability departing from 1_{σ_k} is:

$$p^S(\sigma) = \begin{cases} 1, & \text{if } \sigma = \sigma_k \\ 0, & \text{otherwise.} \end{cases} \tag{5.31}$$

Therefore, the probabilities of the solutions after the selection operator keep the same values of 1_{σ_k}, that is, $P^S = 1_{\sigma_k} = P$. Hence, bearing in mind that in Section 5.2.4 it has been assumed that the estimated model from a degenerate distribution is the same degenerate distribution, $G(1_{\sigma_k}) = 1_{\sigma_k}$ is obtained.

However, the degenerate distributions are not the only fixed points of the discrete dynamical system G. By definition of G, any Mallows probability distribution for which the algorithm learns the same distribution is a fixed point; in other words, after the selection operator, if the algorithm estimates the same central permutation and spread parameter as in the previous distribution, then the Mallows probability distribution is a fixed point. In Lemma 11, a formal result of this idea is presented, showing which two equations are sufficient to achieve a fixed Mallows probability distribution.

Lemma 11. *Let P be a Mallows probability distribution with central permutation σ_0 and spread parameter $\theta_0 < \infty$. If for all $\sigma \neq \sigma_0$,*

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p^S(\pi) < \sum_{\pi \in \Sigma_n} d(\pi, \sigma) p^S(\pi) \tag{5.32}$$

and

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p^S(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi). \tag{5.33}$$

are fulfilled, then $G(P) = P$.

Proof. By the maximum likelihood estimator of the parameters of the Mallows model, Inequality (5.32) ensures $\hat{\sigma}_0 = \sigma_0$. In order to prove that $\hat{\theta} = \theta_0$, considering by hypothesis that P is a Mallows model and by Equations (5.17) and (5.33),

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p^S(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) = \frac{n-1}{e^{\theta_0} - 1} - \sum_{i=1}^{n-1} \frac{n-i+1}{e^{(n-i+1)\theta_0} - 1}. \tag{5.34}$$

□

Inequality (5.32) ensures $\hat{\sigma}_0 = \sigma_0$ and Equation (5.33) obtains $\hat{\theta} = \theta_0$. Inequality (5.32) and Equation (5.33) can be written consecutively: for all $\sigma \neq \sigma_0$,

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma) p^S(\pi) \overset{\hat{\sigma}_0 = \sigma_0}{>} \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p^S(\pi) \overset{\hat{\theta} = \theta_0}{=} \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi). \tag{5.35}$$

Lemma 11 presents a sufficient situation to achieve fixed points of the algorithm. Unfortunately, Lemma 11 does not present "the necessary condition" because of one very particular case: when $G(P) = P$, it cannot be ensured that σ_0 obtains the minimum value at Inequality (5.32) (perhaps there are more permutations which obtain the minimum value), even if $\hat{\theta} = \theta_0$. In the case that σ_0 is the unique solution of Inequality (5.32), then Lemma 11 would present the necessary condition to be a fixed point. To avoid these specific scenarios and the equality case in Inequality (5.32), which represent zero Lebesgue measure sets, from now on we will consider that σ_0 is the estimated central permutation. In practice, the EDA can be designed to have a preference criteria for ties.

Based on Lemma 11, our next objective is to observe the sufficient conditions to achieve fixed points of the algorithm when f is a *needle in a haystack* function. First, it is studied when $\hat{\theta} = \theta_0$, and then whether or not $\hat{\sigma}_0 = \sigma_0$ is satisfied. Let us study Equation (5.33).

$$\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p^S(\pi) = \sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$$

$$\overset{(5.26)}{\iff} p(\sigma^*)\cdot d(\sigma^*,\sigma_0) + (1-p(\sigma^*))\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi) = \sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$$

$$\iff p(\sigma^*)\cdot d(\sigma^*,\sigma_0) = p(\sigma^*)\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$$

$$\iff d(\sigma^*,\sigma_0) = \sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi). \tag{5.36}$$

From Equation (5.36) we can deduce that $\mathrm{MM}(\sigma_0,\theta_0)$ is not a fixed point if $d(\sigma^*,\sigma_0) \geq D/2$. This is due to the fact that the right-hand side of Equation (5.17) tends to 0 when θ tends to infinity and the supreme possible value of $\sum_{\pi\in\Sigma_n} d(\pi,\sigma)p(\pi)$ is $D/2$. Consequently, $\mathrm{MM}(\sigma_0,\theta_0)$ is not a fixed point if $d(\sigma^*,\sigma_0) \geq D/2$. Note that this also means that if we start with $P_0 \sim \mathrm{MM}(\sigma_0,\theta_0)$ such that $d(\sigma^*,\sigma_0) \geq D/2$, then the algorithm can only converge to a solution σ unequal to σ_0 such that $d(\sigma^*,\sigma) < D/2$.

Let us observe whether $\hat{\sigma}_0 = \sigma_0$ is fulfilled when $\hat{\theta} = \theta_0$ and $d(\sigma^*,\sigma_0) < D/2$ (considering the case that the estimated central permutation is unique):

$$\hat{\sigma}_0 = \sigma_0 \iff \sum_{\pi\in\Sigma_n} d(\pi,\sigma)p^S(\pi) > \sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p^S(\pi), \forall\sigma \neq \sigma_0. \tag{5.37}$$

The right-hand side of the equation is simplified by Equation (5.36):

$$\sum_{\pi\in\Sigma_n} d(\pi,\sigma)p^S(\pi) > d(\sigma^*,\sigma_0), \forall\sigma \neq \sigma_0. \tag{5.38}$$

By the definition of the selection probability (Equation (5.26)),

$$p(\sigma^*)d(\sigma^*,\sigma) + (1-p(\sigma^*))\sum_{\pi\in\Sigma_n} d(\pi,\sigma)p(\pi) > d(\sigma^*,\sigma_0), \forall\sigma \neq \sigma_0. \tag{5.39}$$

Solving for the summation in the left-hand side of the inequality,

$$\sum_{\pi\in\Sigma_n} d(\pi,\sigma)p(\pi) > \frac{d(\sigma^*,\sigma_0) - p(\sigma^*)d(\sigma^*,\sigma)}{1-p(\sigma^*)}, \forall\sigma \neq \sigma_0. \tag{5.40}$$

The value of the right-hand side of Inequality (5.40) can vary according to $d(\sigma^*,\sigma)$. In order to avoid repeating the same proof for different values of $d(\sigma^*,\sigma)$, let us consider the maximum possible value of the right-hand side of Inequality (5.40), which is the worst possible case, and prove it. Substituting the expression $d(\sigma^*,\sigma_0) - p(\sigma^*)d(\sigma^*,\sigma)$ by $d(\sigma^*,\sigma_0)$, we obtain the following inequality:

$$\sum_{\pi\in\Sigma_n} d(\pi,\sigma)p(\pi) > \frac{d(\sigma^*,\sigma_0)}{1-p(\sigma^*)}. \tag{5.41}$$

On the left-hand side of Inequality (5.41), the sum depends on σ. In order to prove for all $\sigma \neq \sigma_0$, let us take the smallest possible value. Considering that P is a Mallows model centered at σ_0, the probabilities are

ordered according to their distance to σ_0. So, from the set $\Sigma_n \backslash \{\sigma_0\}$, any solution σ at distance 1 from σ_0 has the lowest value $\sum_{\pi \in \Sigma_n} d(\pi, \sigma) p(\pi)$, because $d(\pi, \sigma) = d(\pi, \sigma_0) \pm 1$. Rewriting the previous equation for a solution σ at distance 1 from σ_0,

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma) p(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) + \sum_{\substack{\pi \in \Sigma_n \\ d(\pi, \sigma_0) < d(\pi, \sigma)}} p(\pi) - \sum_{\substack{\pi \in \Sigma_n \\ d(\pi, \sigma_0) > d(\pi, \sigma)}} p(\pi) > \frac{d(\sigma^*, \sigma_0)}{1 - p(\sigma^*)}$$

$$\overset{(5.36)}{\iff} d(\sigma^*, \sigma_0) + \sum_{\substack{\pi \in \Sigma_n \\ d(\pi, \sigma_0) < d(\pi, \sigma)}} p(\pi) - \sum_{\substack{\pi \in \Sigma_n \\ d(\pi, \sigma_0) > d(\pi, \sigma)}} p(\pi) > \frac{d(\sigma^*, \sigma_0)}{1 - p(\sigma^*)}$$

$$\iff \sum_{\substack{\pi \in \Sigma_n \\ d(\pi, \sigma_0) < d(\pi, \sigma)}} p(\pi) - \sum_{\substack{\pi \in \Sigma_n \\ d(\pi, \sigma_0) > d(\pi, \sigma)}} p(\pi) > \frac{p(\sigma^*) d(\sigma^*, \sigma_0)}{1 - p(\sigma^*)}. \tag{5.42}$$

In order to simplify the previous equation, let us introduce some new notation and definitions.

Definition 17. *For any σ in Σ_n and $d = 0, \ldots, D$, let us denote*

$$m_n(d) = |\{\pi \in \Sigma_n : d(\pi, \sigma) = d\}|. \tag{5.43}$$

The sequence A008302 in The On-Line Encyclopedia of Integer Sequences (OEIS) [87] shows the first values and some properties of $m_n(d)$ numbers.

Definition 18. *For any σ and τ in Σ_n such that $d(\sigma, \tau) = 1$, and $d = 0, \ldots, D$, let us denote*

$$\mathcal{D}_d = \{\pi \in \Sigma_n : d(\pi, \sigma) = d \text{ and } d(\pi, \tau) = d + 1\} \tag{5.44}$$

and $m_n^1(d) = |\mathcal{D}_d|$.

The sequence of non-negative numbers $m_n^1(d)$ has been added in OEIS [87] (sequence A307429) and several properties have been explained in Appendix C. To rewrite Inequality (5.42), Properties *(ii)*, *(iii)* and *(iv)* from Appendix C have been used. These enunciate that $m_n(d) = m_n^1(d) + m_n^1(d-1)$, $m_n^1(d) = m_n^1(D - d - 1)$ and that $m_n^1(d) > m_n^1(d-1)$ when $d \in \{0, \ldots, d_{max}\}$, where $d_{max} = (D/2) - 1$ when D is even and $d_{max} = \lfloor D/2 \rfloor$ when D is odd. Remembering that $\varphi(\theta) = \sum_{\sigma \in \Sigma_n} e^{-\theta d(\sigma, \sigma_0)}$ is the normalization constant for the Mallows probability distribution, Inequality (5.42) can be rewritten in the following way (let us denote $d(\sigma^*, \sigma_0) = d^*$):

$$
\begin{aligned}
&\sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\sigma_0)<d(\pi,\sigma)}} p(\pi) - \sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\sigma_0)>d(\pi,\sigma)}} p(\pi) > \frac{p(\sigma^*)d(\sigma^*,\sigma_0)}{1-p(\sigma^*)}\\
\iff &\sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\sigma_0)<d(\pi,\sigma)}} \frac{e^{-\hat{\theta}d(\pi,\sigma_0)}}{\varphi(\hat{\theta})} - \sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\sigma_0)>d(\pi,\sigma)}} \frac{e^{-\hat{\theta}d(\pi,\sigma_0)}}{\varphi(\hat{\theta})} > \frac{e^{-d^*\hat{\theta}}}{\varphi(\hat{\theta})}\cdot\varphi(\hat{\theta})\cdot\frac{d^*}{\varphi(\hat{\theta})-e^{-d^*\hat{\theta}}}\\
\iff &(\varphi(\hat{\theta})-e^{-d^*\hat{\theta}})\left(\sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\sigma_0)<d(\pi,\sigma)}} e^{-\hat{\theta}d(\pi,\sigma_0)} - \sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\sigma_0)>d(\pi,\sigma)}} e^{-\hat{\theta}d(\pi,\sigma_0)}\right) > d^*\cdot\varphi(\hat{\theta})\cdot e^{-d^*\hat{\theta}}\\
\iff &(\varphi(\hat{\theta})-e^{-d^*\hat{\theta}})\sum_{i=0}^{D}\left(m_n^1(i)-m_n^1(i-1)\right)e^{-i\hat{\theta}} > d^*\cdot\varphi(\hat{\theta})\cdot e^{-d^*\hat{\theta}} = \sum_{i=0}^{D} d^*\cdot m_n(i)\cdot e^{-(d^*+i)\hat{\theta}}\\
\iff &\varphi(\hat{\theta})\sum_{i=0}^{D}\left(m_n^1(i)-m_n^1(i-1)\right)e^{-i\hat{\theta}} > \sum_{i=0}^{D}\left(m_n^1(i)-m_n^1(i-1)+d^*\cdot m_n(i)\right)e^{-(d^*+i)\hat{\theta}}\\
\iff &\sum_{i=0}^{D}\sum_{j=0}^{D} m_n(i)\cdot\left(m_n^1(j)-m_n^1(j-1)\right)e^{-(i+j)\hat{\theta}} > \sum_{i=0}^{D}\left((d^*+1)m_n^1(i)+(d^*-1)m_n^1(i-1)\right)e^{-(d^*+i)\hat{\theta}}.
\end{aligned}
\tag{5.45}
$$

The proof of Inequality (5.45) is shown in Appendix D. Therefore, the learned central permutation from $P\sim \mathrm{MM}(\sigma_0,\hat{\theta})$ is σ_0. To sum up, $P\sim\mathrm{MM}(\sigma_0,\theta_0)$ is a fixed point if $d(\sigma^*,\sigma_0)<D/2$ and θ_0 fulfills Equation (5.36).

5.4.3.2 Attraction of the fixed points

In Section 5.4.3.1, all the fixed points of the algorithm, degenerates and non-degenerates, have been studied. Let us define a fixed point of the dynamical system attractive (attractor in [95]) if any Mallows model P near the fixed point will converge to it: that is to say, any P that has the same central estimator as the fixed point and a spread parameter value θ "close" to $\hat{\theta}$ (in the limit sense) will converge to the fixed point. In addition, from the study of the fixed points, several observations have been derived.

For example, from Equation (5.36), the attraction of the non-degenerate fixed points is totally deduced. Let us denote by $\hat{\theta}_{d^*}$ the minimum spread parameter values which fulfill Equation (5.36) according to $d(\sigma^*,\sigma_0)$. In Equation (5.33), $\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p^S(\pi)$ and $\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$ are compared to observe when the estimated spread parameter value remains the same value. Let us denote by $\hat{\theta}$ the spread parameter value which fulfills Equation (5.33). However, $\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p^S(\pi)$ and $\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$ can be compared for any other spread parameter value θ_0. For example, when $\theta_0<\hat{\theta}_{d^*}$, $d(\sigma^*,\sigma_0)<\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$ and $\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p^S(\pi)<\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$, and consequently the learned spread parameter is greater than θ_0; and when $\theta_0>\hat{\theta}_{d^*}$, then the learned spread parameter decreases. This observation shows us that the non-degenerate fixed points are attractive.

Another observation is that for sufficiently large θ_0 we obtain $d(\sigma^*,\sigma_0)>\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$ and, consequently, $\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p^S(\pi)>\sum_{\pi\in\Sigma_n} d(\pi,\sigma_0)p(\pi)$, which implies that $\hat{\theta}_0<\theta_0$. Hence, all the degenerate

fixed points centered at $\sigma \neq \sigma^*$ are not attractive. Consequently, the algorithm ends in a non-degenerate fixed point centered at $\sigma \neq \sigma^*$ or in the degenerate distribution centered at σ^*.

Moreover, Equation (5.37) shows us the condition to estimate σ_0 as the central permutation. Hence, there exists a spread parameter value $\tilde{\theta}_{d^*}$ (dependent on $d(\sigma^*, \sigma_0) < D/2$) such that if $\theta_0 < \tilde{\theta}_{d^*}$, then the estimated central permutation is not σ_0. If $\theta_0 = \tilde{\theta}_{d^*}$, then the algorithm can estimate more than one central permutation and its behavior will depend on the estimated central permutation. However, we will not focus on those exact Mallows models because they represent a zero Lebesgue measure set. In Figure 5.1, the first values of $\hat{\theta}$ which fulfill Equation (5.36) and $\tilde{\theta}_{d^*}$ are displayed for $n = 4, 5, 6$ and 7 and their respective d^* values, showing the proved result. The Y axis is plotted in logaritmic scale to recognize all the lines. In addition, for a fixed value n, it can be verified that when d increases, due to the fact the right-hand side of Equation (5.36) is a decreasing function, Equation (5.36) is fulfilled for a lower $\hat{\theta}_d$ value.

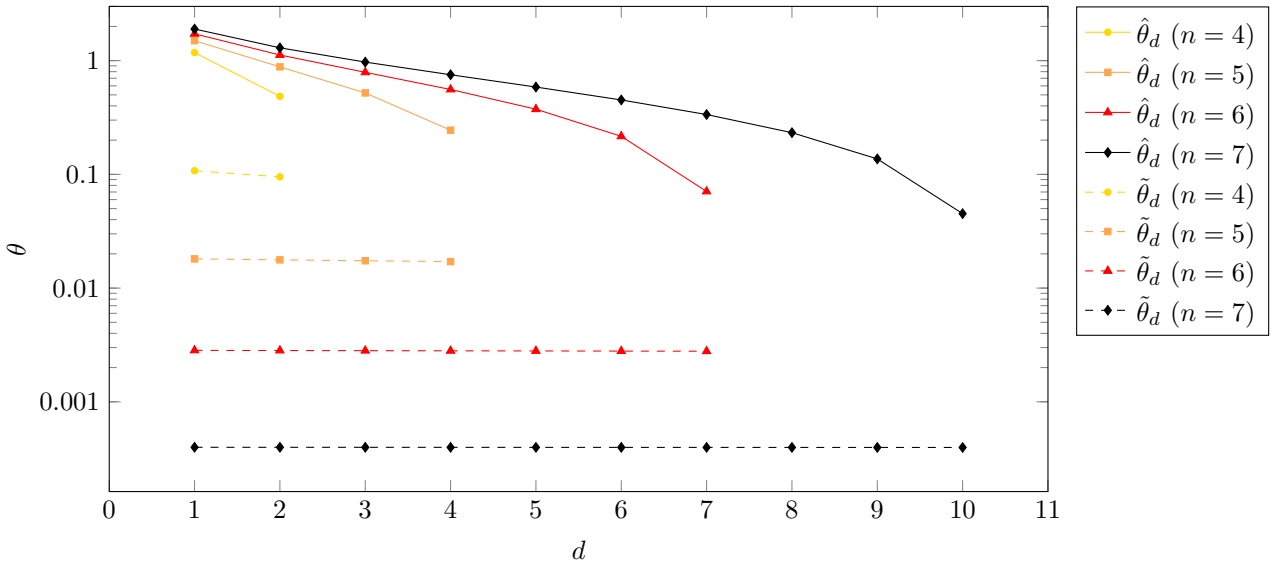


Fig. 5.1: Spread parameter values in which Equation (5.36) (continuous lines) and Equation (5.37) (dashed lines) are fulfilled. Each line represents the value n ($n = 4, 5, 6, 7$) and each point depends on d ($d = 1, \ldots, \lceil D/2 \rceil - 1$).

5.4.3.3 Convergence behavior of the algorithm

After analyzing the attraction of the fixed points, the next step is to study the evolution of the estimated Mallows models; that is, when the algorithm estimates a new central permutation which is different from σ_0, is it possible to limit the number of scenarios of the algorithm in advance? Can we know which fixed point is the convergence point of the algorithm in any situation?

In many cases it is shown to which fixed point the algorithm converges. The main result that is given about the convergence point of the algorithm is Lemma 12. Lemma 12 demonstrates that the algorithm estimates

a central permutation which must be in a set of solutions dependent on σ^* and σ_0. In addition, for any σ_0, there exists a spread parameter value $\tilde{\theta}(\sigma_0)$ such that if $\theta_0 < \tilde{\theta}(\sigma_0)$, then the algorithm estimates a new central permutation different from σ_0.

In order to prove Lemma 12, let us consider Definition 19.

Definition 19. *Let Σ_n be the search space with metric $d(\cdot,\cdot)$. Let σ and π be two solutions of Σ_n. Then, the segment from σ to π, $C(\sigma,\pi)$, is the set with the permutations $\tau \in \Sigma_n$ such that σ, π and τ fulfill the equality in the triangle inequality.*

$$C(\sigma,\pi) = \{\tau \in \Sigma_n : d(\sigma,\tau) + d(\tau,\pi) = d(\sigma,\pi)\}. \tag{5.46}$$

Let us call $\tau \in C(\sigma,\pi)$ a solution between σ and π. Hence, $C(\sigma,\pi)$ is the set that includes all the permutations between σ and π. Let us call the segment from σ to π unique when $|C(\sigma,\pi)| = d(\sigma,\pi) + 1$.

Two swaps are disjoint if the intersection of the sets of elements exchanged by each swap is empty.

Lemma 12. *Let $d(\cdot,\cdot)$ be the Kendall tau distance and f an objective function such that its maximal solution is σ^* and for any $\sigma,\pi \in \Sigma_n$, $d(\sigma,\sigma^*) > d(\pi,\sigma^*)$ if and only if $f(\sigma) \leq f(\pi)$. Let P_0 be a Mallows model with central permutation σ_0, where $d(\sigma^*,\sigma_0) \geq 1$, and spread parameter θ_0. Then, the operator G always estimates a solution $\tau \in C(\sigma^*,\sigma_0)$ as the central permutation of the learned Mallows model.*

Before presenting the proof of Lemma 12, let us consider some preliminary ideas about our permutation space Σ_n and how the solutions can be organized and classified according to their description and the Kendall tau distance d. To do so, let us study the Cayley graph described by (Σ_n, d) metric space.

Let us denote by $CG(V,E)$ the Cayley graph in which $V = \Sigma_n$ and

$$E = \{(\sigma,\pi) \in \Sigma_n \times \Sigma_n \mid d(\sigma,\pi) = 1\}. \tag{5.47}$$

This graph has been studied in [37, 104]. Lemma 2.4 of [37] shows that there are two kinds of cycles formed in $CG(\Sigma_n, E)$. Because d distance has the right invariance property, without loss of generality, let us simplify the notation and explain the two possible cycles formed by the adjacent swaps using the identity permutation I as the reference solution. Let us denote by $[i]$ the adjacent transposition that exchanges the elements of the positions i and $i+1$ ($i = 1,\ldots,n-1$). For example, $[i] \circ I$ represents the solution such that elements of the positions i and $i+1$ from I are swapped ($[i] \circ I = (1 \cdots i+1\ i\ \cdots n)$). Analogously, let us consider a second adjacent transposition $[j]$.

- If $[j] \circ [i] \circ I = [i] \circ [j] \circ I$, then there is a unique 4-cycle in $CG(\Sigma_n, E)$ passing through I, $[i] \circ I$ and $[j] \circ I$. Moreover, the 4-cycle is formed by the following solutions:

$$\{I,\ [i] \circ I,\ [j] \circ [i] \circ I,\ [j] \circ I\}. \tag{5.48}$$

- If $[j] \circ [i] \circ I \neq [i] \circ [j] \circ I$, then $[i] \circ [j] \circ [i] \circ I = [j] \circ [i] \circ [j] \circ I$ and there is a unique 6-cycle in $CG(\Sigma_n, E)$ passing through I, $[i] \circ I$ and $[j] \circ I$. Moreover, the 6-cycle is formed by the following solutions:

$$\{I,\ [i] \circ I,\ [j] \circ [i] \circ I,\ [i] \circ [j] \circ [i] \circ I,\ [i] \circ [j] \circ I,\ [j] \circ I\}. \tag{5.49}$$

By the definition of the generation of the cycles, the distances among the solutions of the same cycle are minimal. That is to say, the distance between two solutions of the same cycle is the number of edges between both solutions in the cycle.

The next observation is that considering any 4-cycle, a partition of Σ_n in 4 sets can be defined.

$$\{\pi_1,\ \pi_2 = [i] \circ \pi_1,\ \pi_3 = [j] \circ \pi_1,\ \pi_4 = [j] \circ [i] \circ \pi_1\}. \tag{5.50}$$

Without loss of generality, let us comment the particular case $\pi_1 = I$, and the same arguments can be applied for any other cycle. If $\pi_1 = I$, then $\pi_2 = (\cdots i+1\ i\ \cdots j\ j+1\ \cdots)$; $\pi_3 = (\cdots i\ i+1\ \cdots j+1\ j\ \cdots)$; and $\pi_4 = (\cdots i+1\ i\ \cdots j+1\ j\ \cdots)$. In order to simplify the notation, the solutions of the 4-cycle can be classified according to the relative positions of the couple i and $i+1$ and the couple j and $j+1$. So, a partition $\{S_1, S_2, S_3, S_4\}$ of Σ_n is defined as follows:

$$\begin{array}{l} S_1 = \{\sigma \in \Sigma_n \mid \sigma(i) < \sigma(i+1) \wedge \sigma(j) < \sigma(j+1)\}; \\ S_2 = \{\sigma \in \Sigma_n \mid \sigma(i) > \sigma(i+1) \wedge \sigma(j) < \sigma(j+1)\}; \\ S_3 = \{\sigma \in \Sigma_n \mid \sigma(i) < \sigma(i+1) \wedge \sigma(j) > \sigma(j+1)\}; \\ S_4 = \{\sigma \in \Sigma_n \mid \sigma(i) > \sigma(i+1) \wedge \sigma(j) > \sigma(j+1)\}. \end{array} \tag{5.51}$$

It is evident that the partition is well-defined. Moreover, among these 4 sets, for each pair of sets a bijection can be described:

$$\begin{array}{ccccccc} S_1 & \longrightarrow & S_2 & \longrightarrow & S_3 & \longrightarrow & S_4 \\ \pi_{S_1} & \longmapsto & \pi_{S_2} & \longmapsto & \pi_{S_3} & \longmapsto & \pi_{S_4} \end{array} \tag{5.52}$$

such that

$$\begin{cases} \pi_{S_1}(i) = \pi_{S_2}(i+1) = \pi_{S_3}(i) = \pi_{S_4}(i+1) \\ \pi_{S_1}(i+1) = \pi_{S_2}(i) = \pi_{S_3}(i+1) = \pi_{S_4}(i) \\ \pi_{S_1}(j) = \pi_{S_2}(j) = \pi_{S_3}(j+1) = \pi_{S_4}(j+1) \\ \pi_{S_1}(j+1) = \pi_{S_2}(j+1) = \pi_{S_3}(j) = \pi_{S_4}(j) \\ \pi_{S_1}(k) = \pi_{S_2}(k) = \pi_{S_3}(k) = \pi_{S_4}(k), \qquad \text{for any } k \neq i, i+1, j, j+1. \end{cases} \tag{5.53}$$

An important property of this defined partition is that if $\sigma \in S_1$, then $d(\pi_1, \sigma) < d(\pi_2, \sigma) = d(\pi_3, \sigma) < d(\pi_4, \sigma)$ is fulfilled and analogously with the solutions of the sets S_2, S_3 and S_4.

The previous idea can be repeated with two non-disjoint adjacent swaps, forming a 6-cycle and defining a partition of Σ_n in 6 sets, and for any cycle. In addition, we can extend the idea by using just one adjacent swap. In this last case, we can define a partition of Σ_n in two sets and a bijection between the sets, according to the relative position of the two elements permuted by the swap. This property is the main argument of the proof of Lemma 12.

Once we know how the solutions are organized in the metric space (Σ_n, d), Lemma 12 is proved by induction as follows. For any solution $\tau \notin C(\sigma^*, \sigma_0)$, there exists another solution $\rho_1 \in \Sigma_n$ such that ρ_1 is "closer" to σ^* and σ_0 than τ and fulfills the following inequality:

$$\sum_{\pi \in \Sigma_n} d(\pi, \tau) p^S(\pi) > \sum_{\pi \in \Sigma_n} d(\pi, \rho_1) p^S(\pi). \tag{5.54}$$

In this way, the argument can be applied for all the solutions not included in $C(\sigma^*, \sigma_0)$ and, therefore, for any solution $\tau \notin C(\sigma^*, \sigma_0)$, there is a solution $\rho \in C(\sigma^*, \sigma_0)$ such that ρ fulfills Inequality (5.54) with regard to τ.

Proof. For any $\tau \notin C(\sigma^*, \sigma_0)$, there are two possible cases: (1) there is a solution ρ_1 such that $d(\tau, \rho_1) = 1$, $d(\tau, \sigma^*) = d(\rho_1, \sigma^*) + 1$ and $d(\tau, \sigma_0) = d(\rho_1, \sigma_0) + 1$ and (2) there is no such solution ρ_1.

In the first case, if i and j are the elements swapped in the adjacent transposition between τ and ρ_1, it means that any solution of $C(\sigma^*, \sigma_0)$ keeps the same relative order between the elements i and j as ρ_1 does. So,

$$\begin{aligned}
&\sum_{\pi \in \Sigma_n} d(\pi, \tau) p^S(\pi) > \sum_{\pi \in \Sigma_n} d(\pi, \rho_1) p^S(\pi) \\
\iff &\sum_{\pi \in \Sigma_n} d(\pi, \rho_1) p^S(\pi) + \sum_{\substack{\pi \in \Sigma_n \\ d(\pi,\tau) > d(\pi,\rho_1)}} p^S(\pi) - \sum_{\substack{\pi \in \Sigma_n \\ d(\pi,\tau) < d(\pi,\rho_1)}} p^S(\pi) > \sum_{\pi \in \Sigma_n} d(\pi, \rho_1) p^S(\pi) \\
\iff &\sum_{\substack{\pi \in \Sigma_n \\ d(\pi,\tau) > d(\pi,\rho_1)}} p^S(\pi) - \sum_{\substack{\pi \in \Sigma_n \\ d(\pi,\tau) < d(\pi,\rho_1)}} p^S(\pi) > 0. \qquad (5.55)
\end{aligned}$$

Let us consider the following bijection:

$$\begin{aligned}
S_\tau = \{\sigma \in \Sigma_n \mid d(\sigma, \tau) < d(\sigma, \rho_1)\} &\longrightarrow S_\rho = \{\sigma \in \Sigma_n \mid d(\sigma, \tau) > d(\sigma, \rho_1)\} \\
\sigma_\tau &\longmapsto \sigma_\rho
\end{aligned} \qquad (5.56)$$

such that $\sigma_\tau(i) = \sigma_\rho(j)$, $\sigma_\tau(j) = \sigma_\rho(i)$ and $\sigma_\tau(k) = \sigma_\rho(k)$, for any $k \neq i, j$. According to the relative position of i and j, σ_ρ is closer to σ^* and σ_0 than σ_τ and therefore, $p^S(\sigma_\rho) > p^S(\sigma_\tau)$ is achieved. Consequently, Inequality (5.55) is obtained.

In the second case, let us suppose that there are no swaps from τ that decrease the distance to σ^* and σ_0 at the same time. First, let us consider an adjacent swap $[i]$ from τ that reduces the distance to σ^*. Let us denote $\rho' = [i] \circ \tau$. Therefore, similar to the first case, a bijection can be defined according to the relative position of the elements in the positions i and $i+1$ in τ. Analogously, let us consider a second swap $[j]$ from τ that reduces the distance to σ_0, denote $\rho'' = [j] \circ \tau$ and define a bijection for the elements positioned at j and $j+1$ in τ. The transpositions $[i]$ and $[j]$ define a unique cycle passing through τ. Moreover, by definition of the swaps and the segment $C(\sigma^*, \sigma_0)$ and the bijections defined in (5.52), this situation can only happen when the swaps $(i\ i+1)$ and $(j\ j+1)$ are not disjoint, which implies that the formed cycle has length 6. Besides, this cycle also implies that if we denote by ρ_τ the furthest solution of the cycle from τ, then ρ_τ is closer to σ^* and σ_0 at the same time than τ. Figure 5.2 presents the unique possible scenario. Hence, $d(\sigma^*, \rho_\tau) + d(\rho_\tau, \sigma_0) < d(\sigma^*, \tau) + d(\tau, \sigma_0)$.

Let us rewrite the sum $\sum_{\pi \in \Sigma_n} d(\pi, \tau) p^S(\pi)$:

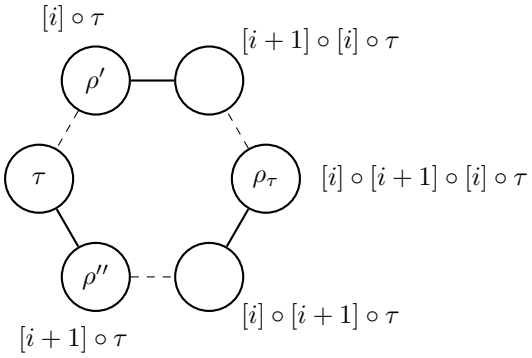


Fig. 5.2: Example of the generated 6-cycle over τ with two non-disjoint adjacent swaps.

$$\begin{aligned}\sum_{\pi\in\Sigma_n} d(\pi,\tau)p^S(\pi) = &\sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\rho')<d(\pi,\tau)\\ d(\pi,\rho')<d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi) + \sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\rho')<d(\pi,\tau)\\ d(\pi,\rho')=d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi)\\ &+ \sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\rho')>d(\pi,\tau)\\ d(\pi,\rho')>d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi) + \sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\rho')>d(\pi,\tau)\\ d(\pi,\rho')=d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi).\end{aligned} \tag{5.57}$$

We distribute the sums in two groups, depending on whether or not $d(\pi,\rho') = d(\pi,\rho'')$.

$$\begin{aligned}&\left[\sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\rho')<d(\pi,\tau)\\ d(\pi,\rho')=d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi) + \sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\rho')>d(\pi,\tau)\\ d(\pi,\rho')=d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi)\right] +\\ &+\left[\sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\rho')<d(\pi,\tau)\\ d(\pi,\tau)<d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi) + \sum_{\substack{\pi\in\Sigma_n\\ d(\pi,\rho')>d(\pi,\tau)\\ d(\pi,\tau)>d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi)\right].\end{aligned} \tag{5.58}$$

To prove that the first square brackets sum is a positive value, for a solution $\pi \in \Sigma_n$, if $d(\pi,\rho') = d(\pi,\rho'') < d(\pi,\tau)$, then $d(\pi,\rho_\tau) < d(\pi,\tau)$. So, if we denote by $(i\ i+1\ i+2)$ the set of elements which are permuted in the 6-cycle, we define the following bijection:

$$\begin{aligned}S_\tau = \{\sigma \in \Sigma_n \mid d(\sigma,\rho_\tau) - d(\sigma,\tau) = 3\} &\longrightarrow S_\rho = \{\sigma \in \Sigma_n \mid d(\sigma,\tau) - d(\sigma,\rho_\tau) = 3\}\\ \sigma_\tau &\longmapsto \sigma_\rho\end{aligned} \tag{5.59}$$

such that $\sigma_\tau(i) = \sigma_\rho(i+2)$, $\sigma_\tau(i+2) = \sigma_\rho(i)$ and $\sigma_\tau(k) = \sigma_\rho(k)$, for any $k \neq i, i+2$. Therefore, a correspondence between both sets is shown, and by the definition of the sets, $p^S(\sigma_\rho) > p^S(\sigma_\tau)$ is obtained for all $\sigma_\tau \in S_\tau$.

For the second square bracket, if $d(\pi,\rho') < d(\pi,\tau) < d(\pi,\rho'')$, then $d(\pi,\tau) = d(\pi,\rho') + 1 = d(\pi,\rho'') - 1$, and if $d(\pi,\rho') > d(\pi,\tau) > d(\pi,\rho'')$, then $d(\pi,\tau) = d(\pi,\rho') - 1 = d(\pi,\rho'') + 1$. So, the second square bracket of (5.58) can be rewritten in the following way:

$$\begin{aligned}
&\sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')<d(\pi,\tau) \\ d(\pi,\tau)<d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi) + \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')>d(\pi,\tau) \\ d(\pi,\tau)>d(\pi,\rho'')}} d(\pi,\tau)p^S(\pi) \\
&= \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')\neq d(\pi,\rho'')}} d(\pi,\rho')p^S(\pi) + \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')<d(\pi,\tau) \\ d(\pi,\tau)<d(\pi,\rho'')}} p^S(\pi) - \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')>d(\pi,\tau) \\ d(\pi,\tau)>d(\pi,\rho'')}} p^S(\pi) \\
&= \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')\neq d(\pi,\rho'')}} d(\pi,\rho'')p^S(\pi) - \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')<d(\pi,\tau) \\ d(\pi,\tau)<d(\pi,\rho'')}} p^S(\pi) + \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')>d(\pi,\tau) \\ d(\pi,\tau)>d(\pi,\rho'')}} p^S(\pi). \qquad (5.60)
\end{aligned}$$

Therefore, depending on θ_0, it can be ensured that

$$\sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')<d(\pi,\tau) \\ d(\pi,\tau)<d(\pi,\rho'')}} p^S(\pi) - \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')>d(\pi,\tau) \\ d(\pi,\tau)>d(\pi,\rho'')}} p^S(\pi) > 0 \text{ or } - \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')<d(\pi,\tau) \\ d(\pi,\tau)<d(\pi,\rho'')}} p^S(\pi) + \sum_{\substack{\pi\in\Sigma_n \\ d(\pi,\rho')>d(\pi,\tau) \\ d(\pi,\tau)>d(\pi,\rho'')}} p^S(\pi) > 0. \qquad (5.61)$$

Consequently, there is a solution $\rho_1 \in \{\rho', \rho''\}$ such that

$$\sum_{\pi\in\Sigma_n} d(\pi,\tau)p^S(\pi) > \sum_{\pi\in\Sigma_n} d(\pi,\rho_1)p^S(\pi). \qquad (5.62)$$

So, for $\tau \notin C(\sigma^*,\sigma_0)$, there exists a solution $\rho_1 \in \Sigma_n$ such that $d(\rho_1,\tau) = 1$ and ρ_1 fulfills Inequality (5.54). If $\rho_1 \notin C(\sigma^*,\sigma_0)$, then by the same arguments, there exists another solution $\rho_2 \in \Sigma_n$ such that $d(\rho_1,\rho_2) = 1$ and ρ_2 fulfills Inequality (5.54) with regard to ρ_1, and so on. Because $\tau \notin C(\sigma^*,\sigma_0)$, at least one induction step must fulfill the first situation explained in this proof (fulfilling Inequality (5.55)). Consequently, ρ_i is a solution from $C(\sigma^*,\sigma_0)$ such that it is a better estimator than $\rho_1,\ldots,\rho_{i-1}$ and τ. □

Lemma 12 shows us that the algorithm estimates central permutations from the set $C(\sigma^*,\sigma_0)$. Bear in mind that during the proof of Lemma 12, the particular expression of f has not been used. Therefore, for our particular case, we can deduce Corollary 8.

Corollary 8. *Let f be a* needle in a haystack *function centered at σ^* and P_0 a Mallows model with central permutation σ_0, where $d(\sigma^*,\sigma_0) = d^* \geq 1$, and spread parameter θ_0. Then, the operator G always estimates a solution $\tau \in C(\sigma^*,\sigma_0)$ as the central permutation of the learned Mallows model.*

Proof. When f is a *needle in a haystack* function, then $f(\sigma) < f(\sigma^*)$ for any $\sigma \neq \sigma^*$. Hence, the conditions of Lemma 12 are fulfilled. □

To summarize, the operator G ends in a non-degenerate fixed point or in the degenerate distribution centered at σ^*. The non-degenerate fixed points are centered at solutions σ such that $d(\sigma^*, \sigma) < D/2$. In addition, when the algorithm estimates a different solution of σ_0, the learned central estimator is a solution from $C(\sigma^*, \sigma_0)\backslash\{\sigma_0\}$.

All the results of Sections 5.4.1,5.4.2 and 5.4.3 are briefly shown in Table 5.1. In the first column, the section is shown. In the second and third columns, the initial parameters of P_0 (σ_0 and θ_0) are described. Finally, in the last column, the explanations of the performance of the algorithm for each situation can be found.

Table 5.1: Classification of the behaviors of the EDA. f: Needle in a haystack(σ^*) and $P_0 \sim \text{MM}(\sigma_0, \theta_0)$, where $D = n(n-1)/2$.

Section	Initial σ_0	Initial θ_0	Performance
5.4.1	$\sigma \in \Sigma_n$	$\theta_0 = 0$ (P_0 uniform)	The algorithm converges to the degenerate distribution centered at σ^*.
5.4.2	σ^*	$\theta_0 > 0$	The algorithm converges to the degenerate distribution centered at σ^*.
5.4.3	$\sigma \in \Sigma_n$ s.t. $0 < d(\sigma^*, \sigma) < D/2$	$\theta_0 > 0$	There exists a spread parameter value $\tilde{\theta}_d$ in which $\min_{\tau \in \Sigma_n} \sum_{\pi \in \Sigma_n} d(\pi, \tau) p^S(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma) p^S(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p^S(\pi)$ for a particular $\sigma \in C(\sigma^*, \sigma_0)\backslash\{\sigma_0\}$, by Corollary 8. • If $\theta_0 < \tilde{\theta}_d$, then $\hat{\sigma}_0 \neq \sigma_0$. So, the algorithm estimates a new central permutation $\sigma' \in C(\sigma^*, \sigma_0)\backslash\{\sigma_0\}$ and the convergence behavior of the operator G is the same as the case when $P_0 \sim \text{MM}(\sigma', \hat{\theta})$. • If $\theta_0 = \tilde{\theta}_d$, then $\hat{\sigma}_0 \in C(\sigma^*, \sigma_0)$. According to the estimated central permutation, if $\hat{\sigma}_0 \neq \sigma_0$, it behaves as the case $\theta_0 < \tilde{\theta}_d$; otherwise, it behaves as the case $\theta_0 > \tilde{\theta}_d$. • If $\theta_0 > \tilde{\theta}_d$, then the algorithm converges to the fixed point $\text{MM}(\sigma_0, \hat{\theta}_d)$ such that $\hat{\theta}_d$ is the spread parameter value where $\sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p^S(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) = d(\sigma^*, \sigma_0)$.
5.4.3	$\sigma \in \Sigma_n$ s.t. $d(\sigma^*, \sigma) \geq D/2$	$\theta_0 > 0$	If $\hat{\sigma}_0 = \sigma_0$, then $\hat{\theta} < \theta_0$, and the algorithm will be at the same situation as the beginning with a lower spread parameter. Otherwise, the algorithm estimates a new central estimator $\sigma' \in C(\sigma^*, \sigma_0)\backslash\{\sigma_0\}$. Consequently, the operator G cannot converge to any solution $\sigma \in \Sigma_n$ such that $d(\sigma^*, \sigma) \geq D/2$.

5.5 Limiting behavior for a Mallows model function

In this section, the function f to optimize is a Mallows probability distribution with central permutation σ^* and spread parameter $\theta^* > 0$, without loss of generality. The Mallows model has been studied as an example of a unimodal objective function with different quality of solutions according to their distance to the central permutation. The objective of this section is to analyze the relation among the learned Mallows probability distributions by our dynamical system and the objective function. For that reason, we believe that it is a motivating starting point to study unimodal functions. In Section 5.5.1, the initial probability distribution P_0 is a uniform distribution and the procedure of the algorithm at each iteration is analyzed. In Section 5.5.2, P_0 is a Mallows probability distribution centered at $\sigma \neq \sigma^*$. In this scenario, the fixed points of the algorithm and the convergence behavior of the algorithm are studied, in a similar way as in Section 5.4.3.

5.5.1 P_0 a uniform initial probability distribution

In this section, it is proved that when the initial probability distribution and the fitness functions are Mallows models centered at the same solution, the algorithm converges to the degenerate distribution centered at the optimum. The obtained result is summarized in the following lemma.

Lemma 13. *Let f be a Mallows model centered at σ^* and spread parameter θ^* and P_0 a Mallows model with central permutation σ^* and spread parameter $\theta_0 \geq 0$. Then, the proposed EDA always converges to the degenerate distribution centered at σ^*.*

Proof. For this particular scenario, we have studied how the algorithm performs at each iteration, analogous to Section 5.4.1. Let us start the demonstration from the case that P_0 is a uniform distribution. First, in order to calculate $P_1 = G(P_0)$, let us calculate P_0^S.

Bear in mind that the 2-tournament does not consider the exact function values of the solutions. In other words, by the definition of the Mallows probability distribution, a solution is selected more often if it is closer to σ^*, and to study the selection between two solutions, their distances to σ^* are compared. With this property in mind, we can rewrite Equation (5.13) in the following way: for any iteration of the algorithm i,

$$p_i^S(\sigma) = 2 \sum_{\substack{\pi \in \Sigma_n \\ d(\sigma,\sigma^*) < d(\pi,\sigma^*)}} p_i(\sigma) p_i(\pi) + \sum_{\substack{\pi \in \Sigma_n \\ d(\sigma,\sigma^*) = d(\pi,\sigma^*)}} p_i(\sigma) p_i(\pi). \tag{5.63}$$

The next step is to estimate the central permutation and spread parameter from P_0^S to learn P_1. First, to estimate σ_0, let us order the solutions increasingly according to their distance from σ^*. Remember that two solutions have the same probability to be selected if they are at the same distance from σ^*. For any $\sigma \in \Sigma_n$,

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma) \cdot p_0^S(\pi) = \sum_{d=0}^{D} \left(p_0^S(\tilde{\sigma}_d) \sum_{\substack{\pi \in \Sigma_n \\ d(\pi,\sigma^*) = d}} d(\pi, \sigma) \right), \tag{5.64}$$

where $\tilde{\sigma}_d$ denotes a solution at distance d from σ^*: $d(\tilde{\sigma}_d, \sigma^*) = d$.

By Equation (5.63), $p_0^S(\tilde{\sigma}_0) > p_0^S(\tilde{\sigma}_1) > \cdots > p_i^S(\tilde{\sigma}_D)$. So, by Equation (5.16), the maximum likelihood estimator of σ_0 must minimize $\sum_{\pi \in \Sigma_n} d(\pi, \hat{\sigma}_0) \cdot p_0^S(\pi)$, knowing that the selection probabilities are ordered according to their distance to σ^* (the lower the distance from σ^* to π, the higher the value $p^S(\pi)$ is). For that reason, the maximum likelihood estimator of σ_0 is σ^*, and consequently, P_1 follows a Mallows model with central permutation σ^* and a positive spread parameter θ_1, as a consequence of Lemma 9.

The previous arguments can be used for any iteration. Hence, P_i is a Mallows model with central permutation σ^* and spread parameter $\theta_i > 0$, for any $i \in \mathbb{N}$. In order to see the evolution of the algorithm and the convergence behavior, let us prove that θ_i increases at each iteration. To this end, the difference between the values of the left-hand side of Equation (5.17) in two consecutive iterations are analyzed: $\sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p_i^S(\pi)$ and $\sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p_{i+1}^S(\pi)$. By the same arguments used in Section 5.4.1, the equality $\sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p_i^S(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p_{i+1}(\pi)$ is obtained. Let us use the sequence $m_n(0), m_n(1), \ldots, m_n(D)$ given in Definition 17 and simplify the notation of the probabilities. By definition of the selection operator, for any $\sigma \in \Sigma_n$ such that $d(\sigma, \sigma^*) = d$, $p^S(\sigma)$ can be rewritten in the following way:

$$p^S(\sigma) = p(\sigma) \left(2 \left(1 - \sum_{i=0}^{d-1} m_n(i) p(\tilde{\sigma}_i) \right) - m_n(d) p(\tilde{\sigma}_d) \right). \tag{5.65}$$

Hence,

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p^S(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p(\pi) + \sum_{d=1}^{D} m_n(d) \cdot d \cdot p(\tilde{\sigma}_d) \left(1 - 2 \sum_{i=0}^{d-1} m_n(i) p(\tilde{\sigma}_i) - m_n(d) p(\tilde{\sigma}_d) \right), \tag{5.66}$$

Let us define the function h:

$$h(\theta) = \sum_{d=1}^{D} m_n(d) \cdot d \cdot p(\tilde{\sigma}_d) \left(1 - 2 \sum_{i=0}^{d-1} m_n(i) p(\tilde{\sigma}_i) - m_n(d) p(\tilde{\sigma}_d) \right). \tag{5.67}$$

For any $\theta \geq 0$, $h(\theta)$ is a negative value (see proof in Proposition 7 of Appendix D). Consequently,

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p^S(\pi) < \sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) \cdot p(\pi), \tag{5.68}$$

and due to the fact that the function g defined in Equation (5.19) is a strictly decreasing function over θ, we obtain $\theta_{i+1} > \theta_i$.

Therefore, after applying our modeling, departing from a uniform distribution, to a function defined as a Mallows model, the algorithm converges to the degenerate distribution centered at σ^*. □

5.5.2 P_0 a Mallows probability distribution with central permutation σ_0, where $d(\sigma^*, \sigma_0) = d^* \geq 1$, and spread parameter θ_0

The algorithm can experience many different behaviors depending on σ^* and σ_0. However, there are groups of different central permutations σ_0 such that the algorithm behaves analogously. The analogy of the analysis

with different central permutations can be obtained by means of symmetry among the solutions of Σ_n. Due to the difficulty of studying all of them, we have worked in a similar way as in Section 5.4.3. In Section 5.5.2, the following proof idea is used:

i) In Section 5.5.2.1, the fixed points and their attraction are calculated.
   - First, it is observed that any degenerate distribution is a fixed point.
   - Then, the equations such that any non-degenerate fixed point must fulfill are calculated.

ii) In Section 5.5.2.2, the convergence behavior of the algorithm is explained and an example is shown.

A summary of all the results obtained in Section 5.5 is shown in Table 5.2 at the end of this section.

5.5.2.1 Fixed points of the algorithm and their attraction

The case $n = 2$ will not be explained because of its simplicity. From now on, let us suppose that $n \geq 3$ and study the fixed points of our discrete dynamical system G. As in Section 5.4.3.1, knowing that any degenerate distribution is a fixed point of the discrete dynamical system G, let us focus on the non-degenerate fixed points.

For any Mallows probability distribution P, $G(P) = P$ if and only if the estimated central permutation and spread parameter are the same as those of P. So, if Equation (5.35) is fulfilled, then P is a non-degenerate fixed point. Let us study the equality of Equation (5.35). We say that P is a candidate fixed point if it satisfies Equation (5.33). Note that if P is a candidate fixed point, then $\hat{\theta} = \theta$.

$$
\begin{aligned}
& \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p^S(\pi) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) \\
\overset{(5.63)}{\iff} & \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) \left(\sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) > d(\pi, \sigma^*)}} 2p(\tau) + \sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) = d(\pi, \sigma^*)}} p(\tau) \right) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) \\
\iff & \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) \left(\sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) > d(\pi, \sigma^*)}} p(\tau) \right) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) \left(\sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) < d(\pi, \sigma^*)}} p(\tau) \right) \\
\iff & \sum_{\pi \in \Sigma_n} \sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) > d(\pi, \sigma^*)}} p(\pi) p(\tau) \left[d(\pi, \sigma_0) - d(\tau, \sigma_0) \right] = 0 \\
\iff & \sum_{\pi \in \Sigma_n} \sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) > d(\pi, \sigma^*)}} e^{-\theta(d(\pi, \sigma_0) + d(\tau, \sigma_0))} \left[d(\pi, \sigma_0) - d(\tau, \sigma_0) \right] = 0. \qquad (5.69)
\end{aligned}
$$

As can be observed, Equation (5.69) shows the first condition for a Mallows probability distribution P centered at σ_0 to be a fixed point. Equation (5.69) has at least one solution θ (depending on n, σ^* and σ_0, it may have more than one). One way to calculate the number of candidate fixed points centered at σ_0

is to count the number of roots in Equation (5.69) by Sturm's theorem [99]. The exponential polynomial in $\theta \in [0, +\infty)$ can be transformed into a polynomial defined in $(0, 1]$ (transforming $e^{-\theta} = x$) in order to apply Sturm's theorem. Moreover, the roots can be numerically solved to find the values of θ in which $P \sim \mathrm{MM}(\sigma_0, \theta)$ are candidate fixed points.

Moreover, for any pair of permutations π, τ (w.l.o.g., $d(\tau, \sigma^*) > d(\pi, \sigma^*)$), if we choose the pair of permutations $I'\pi, I'\tau$ where $I' = (n\ n-1\ \cdots 1)$, the following similarities can be observed:

$$\begin{cases} d(\tau, \sigma^*) > d(\pi, \sigma^*) \iff D - d(I'\tau, \sigma^*) > D - d(I'\pi, \sigma^*) \iff d(I'\tau, \sigma^*) < d(I'\pi, \sigma^*) \\ d(\pi, \sigma_0) - d(\tau, \sigma_0) = D - d(I'\pi, \sigma_0) - D + d(I'\tau, \sigma_0) = d(I'\tau, \sigma_0) - d(I'\pi, \sigma_0). \end{cases} \tag{5.70}$$

Hence,

$$e^{-\theta(d(\pi,\sigma_0)+d(\tau,\sigma_0))}\left[d(\pi,\sigma_0) - d(\tau,\sigma_0)\right] = e^{-2D\theta}e^{\theta\left(d(I'\tau,\sigma_0)+d(I'\pi,\sigma_0)\right)}\left[d(I'\tau,\sigma_0) - d(I'\pi,\sigma_0)\right]. \tag{5.71}$$

Therefore, for any $\sigma_0 \in \Sigma_n$, let us define the function H as follows:

$$H(\sigma_0, \theta) = \sum_{i=1}^{2D-1} H_i e^{-i\theta} = \sum_{\pi \in \Sigma_n} \sum_{\substack{\tau \in \Sigma_n \\ d(\tau,\sigma^*) > d(\pi,\sigma^*)}} e^{-\theta(d(\pi,\sigma_0)+d(\tau,\sigma_0))}\left[d(\pi,\sigma_0) - d(\tau,\sigma_0)\right]. \tag{5.72}$$

By Equation (5.71), $H_i = H_{2D-i}$. In addition, $H(\sigma_0, \theta) = -H(I'\sigma_0, \theta)$ for any $\sigma_0 \in \Sigma_n$ and θ. Consequently, $H(\sigma_0, \hat{\theta}) = 0$ if and only if $H(I'\sigma_0, \hat{\theta}) = 0$. So, if P is a candidate fixed point with central permutation σ_0 and spread parameter $\hat{\theta}$, then a Mallows probability distribution with central permutation $I'\sigma_0$ and spread parameter $\hat{\theta}$ is a candidate fixed point as well.

In addition, from the previous observation, it has been equivalently shown that

$$\sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p^S(\pi) < \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) \iff \sum_{\pi \in \Sigma_n} \sum_{\substack{\tau \in \Sigma_n \\ d(\tau,\sigma^*) > d(\pi,\sigma^*)}} e^{-\theta(d(\pi,\sigma_0)+d(\tau,\sigma_0))}\left[d(\pi,\sigma_0) - d(\tau,\sigma_0)\right] < 0 \tag{5.73}$$

and analogous for the opposite inequality. So, when θ tends to infinity, the highest exponential coefficient of $H(\sigma_0, \theta)$ determines if the value is positive or not.

Considering all the observations of Equation (5.69) and Inequality (5.73), in comparison with the results from Sections 5.4.3.1 and 5.4.3.2, some new scenarios have been observed. The first one is that for a fixed permutation σ_0, there can be more than one candidate fixed point. Hence, the algorithm can converge to more than one probability distribution centered at σ_0. Moreover, from Equation (5.72), similarities between σ_0 and $I'\sigma_0$ have been observed. Secondly, information about the attraction of the fixed points has been analyzed, even if the candidate fixed points are fixed points or not. From Inequality (5.73) whether or not if the degenerate distribution centered at σ_0 is an attractive fixed point can be studied. Furthermore, knowing the attraction of the degenerate distribution, the attraction of all the candidate fixed points is completely defined. Reordering all the candidate fixed points centered at σ_0 according to their spread parameters, they alternate their attraction in order not to obtain two consecutive candidate fixed points with the same attraction. Consequently, the last objective is to observe when a candidate fixed point is a fixed point.

To study if a candidate fixed point is a fixed point, it is necessary to observe if the estimated central permutation $\hat{\sigma}_0$ from a candidate fixed point P centered at σ_0 is exactly σ_0. So as to obtain the same central permutation, the inequality of Equation (5.35) must be fulfilled at the solution $\hat{\theta}$ of Equation (5.69) (assuming the uniqueness of the central permutation). Hence, for all $\sigma \neq \sigma_0$,

$$
\begin{aligned}
& \sum_{\pi \in \Sigma_n} d(\pi, \sigma) p^S(\pi) > \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) \\
\iff & \sum_{\pi \in \Sigma_n} d(\pi, \sigma) p(\pi) \left(1 + \sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) > d(\pi, \sigma^*)}} p(\tau) - \sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) < d(\pi, \sigma^*)}} p(\tau) \right) > \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) \\
\iff & \sum_{\pi \in \Sigma_n} \sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) > d(\pi, \sigma^*)}} p(\pi) p(\tau) \left[d(\pi, \sigma) - d(\tau, \sigma) \right] > \sum_{\pi \in \Sigma_n} p(\pi) \left[d(\pi, \sigma_0) - d(\pi, \sigma) \right] \\
\iff & \sum_{\pi \in \Sigma_n} \sum_{\substack{\tau \in \Sigma_n \\ d(\tau, \sigma^*) > d(\pi, \sigma^*)}} p(\pi) p(\tau) \left[d(\tau, \sigma) - d(\pi, \sigma) \right] < \sum_{\pi \in \Sigma_n} p(\pi) \left[d(\pi, \sigma) - d(\pi, \sigma_0) \right]. \qquad (5.74)
\end{aligned}
$$

Inequality (5.74) shows us the condition to estimate σ_0 as the learned central permutation. Even though it can be completely separated according to their dependence to the distance from σ^*, a general solution cannot be observed (without knowing the particular values of the probabilities and distances) which tells us in advance if Inequality (5.74) is fulfilled or not. Actually, some experimental results show that there are candidate fixed points which do not fulfill Inequality (5.74).

In Figure 5.3, an example of the attraction of the fixed points is shown for $n = 5$. The X axis shows σ_0, numerically indexed according to their distance to σ^*, and the Y axis represents the values of θ_0 which fulfill Equation (5.69). Therefore, each dot represents a candidate fixed point. The yellow or gray color of the dot represents the attraction of the fixed point if it is a fixed point, whereas the point is orange if it does not fulfill Inequality (5.74). For any central permutation σ_0, the degenerate fixed points have been illustrated.

To summarize, Inequality (5.74) ensures exactly which candidates are the fixed points of our dynamical system.

5.5.2.2 Convergence behavior of the algorithm

Before introducing the convergence behavior of the algorithm, let us state Corollary 9, deduced from Lemma 12.

Corollary 9. *Let f be a Mallows model centered at σ^* and spread parameter θ^* and P_0 a Mallows model with central permutation σ_0, where $d(\sigma^*, \sigma_0) = d^* \geq 1$, and spread parameter θ_0. Then, the operator G always estimates a solution $\tau \in C(\sigma^*, \sigma_0)$ as the central permutation of the learned Mallows model.*

Proof. When f is a Mallows model centered at σ^* and spread parameter $\theta^* > 0$, for any $\sigma, \pi \in \Sigma_n$, $f(\sigma) > f(\pi)$ if and only if $d(\sigma, \sigma^*) < d(\pi, \sigma^*)$. Hence, the conditions of Lemma 12 are fulfilled. □

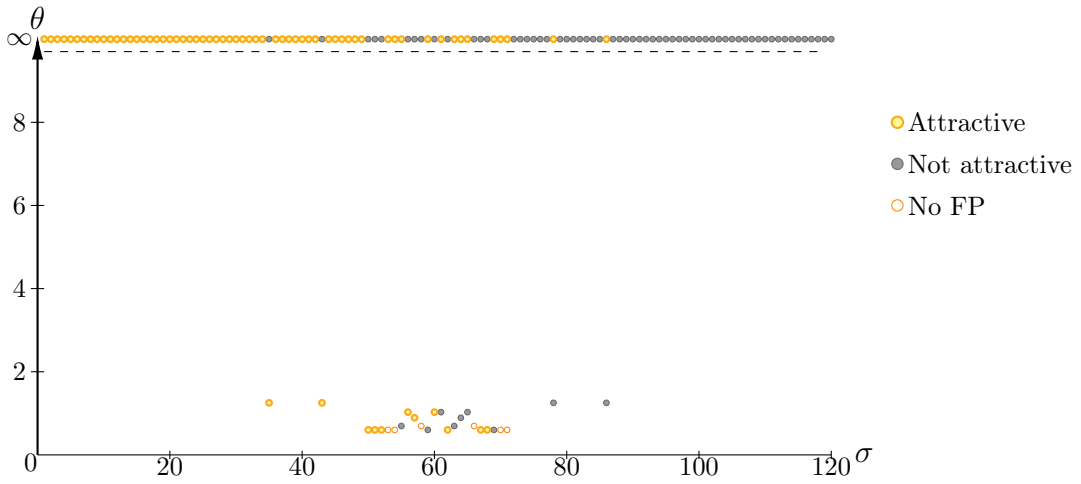


Fig. 5.3: Candidate fixed points of our algorithm (σ and θ values such that $\text{MM}(\sigma, \theta)$ fulfills Equation (5.69)) and their attraction ($n = 5$). The X axis differentiate all the permutations of Σ_5. The Y axis shows the spread parameter values.

Once we have Corollary 9 and we know the fixed points and their attraction, the behavior of the algorithm is totally defined and it can be summarized in the following way:

- For any $P_0 \sim \text{MM}(\sigma_0, \theta_0)$, there exists a spread parameter value $\theta'(\sigma_0)$ dependent on σ_0 such that if $\theta_0 < \theta'(\sigma_0)$, then Inequality (5.74) is not fulfilled for all σ. In that case, by Corollary 9, the estimated central permutation after one iteration of the algorithm is a solution from $C(\sigma^*, \sigma_0) \backslash \{\sigma_0\}$.
- If $\theta_0 > \theta'(\sigma_0)$, then the algorithm estimates σ_0 as the central permutation of the learned Mallows model. Let us classify the different possible behaviors according to the number of fixed points centered at σ_0:
  - If there are no non-degenerate solutions centered at σ_0 (there are no solutions for Equation (5.69)), then the only fixed point centered at σ_0 is the degenerate distribution 1_{σ_0}. In this case, if 1_{σ_0} is attractive, the algorithm converges to it; otherwise, the estimated spread parameter decreases until an iteration when $\hat{\theta} < \theta'(\sigma_0)$ and, therefore, the estimated central permutation is not σ_0 anymore, returning back to the previous situation.
  - If there are $i \geq 1$ non-degenerate fixed points centered at σ_0, then there exist i spread parameter values $\tilde{\theta}_i$ which solve Equation (5.69) and fulfill Inequality (5.74). Hence, $\theta'(\sigma_0)$ and $\tilde{\theta}_j$ for $j = 1, \ldots, i$ divide the interval $(\theta'(\sigma_0), +\infty)$ in $i + 1$ intervals.

    Let us denote by $(\theta'(\sigma_0), \tilde{\theta}_1)$, $(\tilde{\theta}_1, \tilde{\theta}_2)$, $\ldots$, $(\tilde{\theta}_{i-1}, \tilde{\theta}_i)$ and $(\tilde{\theta}_i, +\infty)$ the $i + 1$ formed intervals; P_k the non-degenerate fixed point centered at σ_0 and spread parameter $\tilde{\theta}_k$, for $k = 1, \ldots, i$; and 1_{σ_0} the degenerate fixed point centered at σ_0. There are two possible situations, depending on whether 1_{σ_0} is attractive or not.

    If 1_{σ_0} is attractive, then P_i is not attractive and when $\theta_0 \in (\tilde{\theta}_i, +\infty)$, the algorithm converges to 1_{σ_0}. Moreover, because of the non-attraction of P_i and by the same argument, P_{i-1} is attractive and P_{i-2}

is not attractive, P_{i-3} is attractive and P_{i-4} is not attractive, and so on. Hence, if $\theta_0 \in (\tilde{\theta}_{i-2}, \tilde{\theta}_i)$, the algorithm converges to P_{i-1}; if $\theta_0 \in (\tilde{\theta}_{i-4}, \tilde{\theta}_{i-2})$, the algorithm converges to P_{i-3}; and so on.

Additionally, if 1_{σ_0} is not attractive, then P_i is attractive and P_{i-1} is not attractive, and when $\theta_0 \in (\tilde{\theta}_{i-1}, +\infty)$, the algorithm converges to P_i. Moreover, P_{i-2} is attractive and P_{i-3} is not attractive, and when $\theta_0 \in (\tilde{\theta}_{i-3}, \tilde{\theta}_{i-1})$, the algorithm converges to P_{i-2}. And so on.

Observe that when P_1 is not attractive and $\theta_0 \in (\theta'(\sigma_0), \tilde{\theta}_1)$, the algorithm estimates lower spread parameters until $\hat{\theta}_0 < \theta'(\sigma_0)$. In this case, the algorithm estimates a new central permutation from $C(\sigma^*, \sigma_0) \backslash \{\sigma_0\}$.

Figure 5.4 is presented in order to show a visualization of the possible situations. The horizontal line represents the possible θ_0 value. In each interval, a blue arrow tells us if the estimated spread parameter is higher or lower, and the attraction of each fixed point can be observed. There are four possible cases, depending on the parity of i and the attraction of 1_{σ_0}. In the first two cases, i is an odd number, and in the first and fourth cases, 1_{σ_0} is an attractive fixed point.

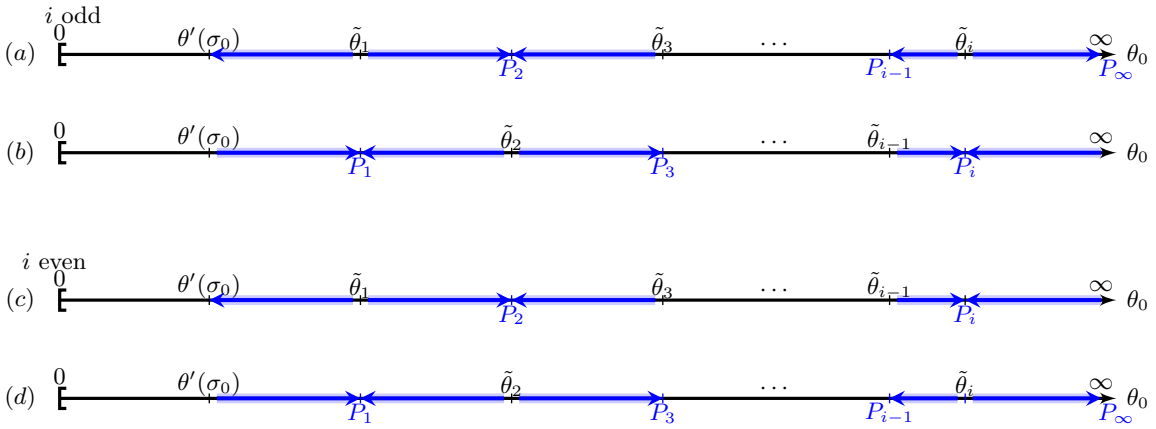


Fig. 5.4: Representation of all the possible scenarios in which the convergence behavior of the algorithm is represented. The cases are divided in 4, according to the parity of the value i and the attraction of the degenerate distribution 1_{σ_0}.

- If $\theta_0 = \theta'(\sigma_0)$, then the algorithm can randomly estimate σ_0 or another $\sigma \in C(\sigma^*, \sigma_0)$ as the new central permutation. In the former case, if the fixed point with the lowest spread parameter centered at σ_0 is attractive, the algorithm will converge to it. Otherwise, the algorithm learns a probability distribution centered at σ_0 and spread parameter $\hat{\theta} < \theta_0$, and it behaves analogous as to the case $\theta_0 < \hat{\theta}_0$. In the latter case, the algorithm estimates a new central permutation σ and spread parameter $\hat{\theta}$, and it will be analogous as $P_0 \sim \mathrm{MM}(\sigma, \hat{\theta})$.

Let us present an example in order to illustrate the behavior described above.

Example 14. Let us consider $n = 5$, f a Mallows model centered at $\sigma^* = I$ and P_0 a Mallows probability distribution with central permutation $\sigma_0 = (21543)$ and spread parameter θ_0. To observe the behavior of the

algorithm, let us calculate the candidate fixed points by Equation (5.69) and the minimum spread parameter value $\theta'(\sigma_0)$ that allows the estimation of σ_0 as the learned central permutation, by Inequality (5.74).

In this particular case, there is only one solution which fulfills Equation (5.69): $\tilde{\theta} \approx 1.2519$. Moreover, Inequality (5.74) shows that the equality is obtained when $\theta'(\sigma_0) \approx 0.2770$. Therefore, a Mallows probability distribution centered at σ_0 with spread parameter value $\tilde{\theta}$ is a fixed point of our mathematical modeling. In addition, if $\theta_0 > \tilde{\theta}$, then $\hat{\theta} < \theta_0$. This last observation implies that the degenerate distribution centered at σ_0 is not attractive, and consequently, $\mathrm{MM}(\sigma_0, \tilde{\theta})$ is an attractive fixed point. Knowing the attraction of the fixed points, the value of θ_0 determines the behavior of the algorithm.

- If $\theta_0 < \theta'(\sigma_0)$, then $\hat{\sigma}_0 \in C(\sigma^*, \sigma_0) \backslash \{\sigma_0\}$. Hence, after one iteration, the algorithm restarts the process with a new central permutation and spread parameter. For example, if $\theta_0 = 0.2760$, then the learned Mallows model after one iteration of the algorithm is $\mathrm{MM}((12453), 0.4016)$; and if $\theta_0 = 0.2700$, then the learned Mallows model is $\mathrm{MM}(\sigma^*, 0.3994)$.
- If $\theta_0 > \theta'(\sigma_0)$, then the algorithm converges to $\mathrm{MM}(\sigma_0, \tilde{\theta})$ distribution.
- If $\theta_0 = \theta'(\sigma_0)$, then the algorithm estimates either σ_0 or $\hat{\sigma}_0 \in C(\sigma^*, \sigma_0) \backslash \{\sigma_0\}$. In the first case, the algorithm converges to $\mathrm{MM}(\sigma_0, \tilde{\theta})$, whereas in the second case, the algorithm estimates $\mathrm{MM}((12453), 0.4023)$ probability distribution after one iteration.

For any σ_0, the same test would be repeated. All the results of Sections 5.5.1 and 5.5.2 are briefly shown in Table 5.2, mentioning the initial parameters of P_0 and explaining the performance of the algorithm.

Table 5.2: Classification of the behaviors of the EDA. $f \sim \text{MM}(\sigma^*, \theta^*)$ and $P_0 \sim \text{MM}(\sigma_0, \theta_0)$

Section	Initial σ_0	Initial θ_0	Performance
5.5.1	$\sigma \in \Sigma_n$	$\theta_0 = 0$ (P_0 uniform)	The algorithm converges to the degenerate distribution centered at σ^*.
5.5.1	σ^*	$\theta_0 > 0$	The algorithm converges to the degenerate distribution centered at σ^*.
5.5.2	$\sigma \in \Sigma_n$ s.t. $d(\sigma^*, \sigma) > 0$	$\theta_0 < \theta'(\sigma_0)$	Inequality (5.74) is not fulfilled. Hence, by Corollary 9, the algorithm estimates a new central permutation $\sigma' \in C(\sigma^*, \sigma_0) \backslash \{\sigma_0\}$. Hence, the convergence behavior of the algorithm is the same as the case when $P_0 \sim \text{MM}(\sigma', \hat{\theta})$.
5.5.2	$\sigma \in \Sigma_n$ s.t. $d(\sigma^*, \sigma) > 0$	$\theta_0 > \theta'(\sigma_0)$	The algorithm estimates σ_0 as the learned central permutation of the Mallows model. According to the number of solutions in Equation (5.69), there are several possible convergence behaviors of the algorithm: • If Equation (5.69) has no solution, then the algorithm converges to σ_0 if $\hat{\theta} > \theta_0$. Otherwise, after some iterations, the algorithm estimates a new central permutation from the segment $C(\sigma^*, \sigma_0)$. Therefore, the convergence behavior of the algorithm is the same as the case when $P_0 \sim \text{MM}(\sigma', \theta')$, being θ' the estimated spread parameter when $\hat{\sigma}_0 = \sigma'$ is obtained. • If Equation (5.69) has at least one solution, then θ_0 is in an interval between two fixed points or $\theta'(\sigma_0) < \theta_0 < \tilde{\theta}_1$. In the first situation, at least one of the fixed points is attractive and the algorithm converges to it. In the second situation, if the fixed point with the lowest spread parameter is attractive, the algorithm converges to it; otherwise, the algorithm estimates a new central permutation from the segment $C(\sigma^*, \sigma_0)$ after some iterations, and it behaves in the same way as in the case $P_0 \sim \text{MM}(\sigma', \theta')$.
5.5.2	$\sigma \in \Sigma_n$ s.t. $d(\sigma^*, \sigma) > 0$	$\theta_0 = \theta'(\sigma_0)$	The algorithm can estimate σ_0 or $\sigma \in C(\sigma^*, \sigma_0) \backslash \{\sigma_0\}$ which fulfills Inequality (5.74) (this election is random). In the former case, the algorithm behaves as the previous case; and in the latter case, the algorithm behaves as in case $\theta_0 < \theta'(\sigma_0)$.

5.6 Conclusions

We have presented a mathematical modeling to study an EDA based on Mallows models using discrete dynamical systems based on the expectations. Under this framework, we have studied the convergence behavior of the algorithm for several objective functions and initial probability distributions. Two different approaches have been followed to study the convergence behavior. For the simplest cases, the computation of one iteration of the algorithm has allowed to prove the limit behavior, whereas for the most complex cases, the fixed points of the algorithm and their attraction have been analyzed. Overall, for the latter, a wide range of possible ending probability distributions and trajectories for the algorithm have been observed, which, given its practical success [17], were by no means anticipated.

The main results can be summarized as follows. When the function to optimize is constant, all Mallows probability distributions are fixed points. When the function to optimize is a *needle in a haystack* function centered at σ^* and the initial probability distribution is a Mallows distribution centered at σ_0, the algorithm converges to the degenerate distribution centered at σ^* or to a non-degenerate Mallows distribution centered at a permutation σ in the segment between σ^* and σ_0 such that the distance between σ and σ^* is lower than $\binom{n}{2}/2$ and a spread parameter which fulfills the condition to be a (attractive) fixed point. Finally, when the function to optimize is a Mallows model centered at σ^* and the initial probability distribution is a Mallows distribution centered at σ_0, the algorithm converges to any Mallows distribution centered at a permutation in the segment between σ^* and σ_0, which is an attractive fixed point. The attraction of all the fixed points provides information in relation to the possible trajectories of the algorithm. In any case, the relation between the initial probability distribution and the objective function completely determines the convergence behavior of the algorithm. Because of that, a classification of the convergence behavior of the algorithm regarding the parameters of the Mallows model is shown.

Part III

General conclusions, Future Work and Publications

6

General conclusions

The idilic goal of the algorithm selection (the meta-algorithmic technique) is to find an association that, given a particular fitness function, returns the most efficient algorithm to solve it. However, it is necessary to understand in advance the main features of the fitness functions and algorithms to create associations among them, without running all the algorithms in the studied COPs (or COP instances). Partial advances in this direction have been produced in the literature such as analyzing the theoretical complexity of binary-based EDAs to know in which COP instances the algorithms obtain the optimal solution efficiently, creating surrogate models for black-box optimization and studying the number of different instances that each COP can generate.

In this thesis, we worked on the study of binary-based COPs and permutation-based EDAs so as to present new theoretical results which improve their understanding. In the study of binary-based COPs, we considered the Walsh transform to represent the problems. This transformation can be applied to any binary-based COPs. In this way, independently of the specific definition of the COP (UBQP, Max-Cut Problem, and so on), they are represented in a similar space. Moreover, the Walsh transform has been considered to present several new results about pseudo-Boolean functions.

In the first part of the thesis, we have studied pseudo-Boolean functions. We have analyzed the instances generated by several specific binary-based COPs in particular and the rankings of the solutions generated by them and by pseudo-Boolean functions of degree $m \leq n$ (where n is the size of the search space) in general using the Walsh transform and the Walsh decomposition. First, we have overviewed the main definition and properties of the Walsh coefficients. We observed that the Walsh transform shows the interaction among all the binary variables and that it identifies additive decomposable functions and additively separable functions. Moreover, because of the fact that the Walsh basis is orthogonal, the exact Walsh coefficients of an additively decomposable/separable function is the sum of the Walsh coefficients of its subfunctions. Therefore, the non-null Walsh coefficient associated to the largest set of binary variables determines the degree of the fitness function and there is no necessity to analyze any fitness function that can be described as a sum of subfuctions.

We have calculated the Walsh coefficients of the UBQP, the Max-Cut Problem and the NPP and we have shown how the common properties among the problems are represented in the Walsh coefficients. We observed that most of the Walsh coefficients of the mentioned problems are null values and the non-null Walsh coefficients follow several patterns. Specifically, when the fitness function is an UBQP instance, all the Walsh coefficients associated to more than two variables are zero, whereas the Walsh coefficient associated to the empty set is dependent on the rest of non-null Walsh coefficients; when the fitness function is a Max-Cut

instance, any Walsh coefficient associated to just one variable or more than two variables is zero, and the Walsh coefficient associated to the empty set is a sum of the coefficients associated to two variables; and when the fitness function is a NPP instance, it can be re-written as an equivalent fitness function whose set of non-null Walsh coefficients is the same as a Max-Cut instance. From these results, the similarities and differences known in the literature have been checked: the NPP can be redefined as a Max-Cut Problem, and the Max-Cut Problem is a particular case of the UBQP. Besides calculating the Walsh decomposition of the mentioned problems, we have also studied the opposite direction: given a Walsh polynomial, we present the exact conditions to determine if there exists an instance of one of the three studied problems whose Walsh decomposition is the given one.

Secondly, we have studied the meaning of the null Walsh coefficients over pseudo-Boolean functions of degree $m \leq n$ and its implications. For this study, we have analyzed pseudo-Boolean functions as ranking of solutions and as ranking generators. Bearing that in mind, we have defined partitions of the solutions (based on the definition of the Walsh functions) according to the number of null bits of each solution over a subset of binary variables s: even and odd solutions. These definitions allowed us to present the conditions that any pseudo-Boolean function of degree $m < n$ satisfies and, consequently, a characterization of m-degree pseudo-Boolean functions: a pseudo-Boolean function is of degree $m \leq n$ if and only if: 1) for any subset of binary variables s such that $|s| > m$, the sum of the fitness function values of all the even solutions and the sum of the fitness function values of all the odd solutions (defined by s) is the same; and 2) there exists a subset of m binary variables s such that the sum of the fitness function values of all the even solutions and the sum of the fitness function values of all the odd solutions (defined by s) is not the same.

In addition, the sets of even and odd solutions have been considered to introduce several new definitions: the word of a ranking defined by s and Dyck Words. From these definitions, we have provided a novel and easy-to-compute procedure to check when a ranking cannot be generated by an m-degree pseudo-Boolean function: if the word of a ranking defined by a subset s is a Dyck Word, then the ranking is impossible to be generated by a pseudo-Boolean function of degree $m < |s|$. Moreover, we have presented a conjecture about the sufficient condition of a ranking to be generated by a $(n-1)$-degree pseudo-Boolean function and two observations about the conjecture. Assuming that the conjecture is true, we have calculated the exact number of rankings generated by $(n-1)$-degree pseudo-Boolean functions. Nevertheless, when $m < n-1$, it has been proved that the presented analysis of the words of a ranking is not sufficient to check when a ranking can be generated by an m-degree pseudo-Boolean function.

To finish the first part of the thesis, we have presented several experiments about the rankings of solutions that can be generated by the UBQP and the NPP. We have verified that sampling coefficients uniformly at random generates "biased fitness functions" (in terms of the frequency of the ranking produced), and we have extracted features and characteristics of the rankings of solutions generated by this process. Particularly, for the NPP, when $n = 3$ it has been observed and demonstrated that generating instances by sampling integer values uniformly at random and sampling instances (rankings) of the problem uniformly at random are equivalent. However, in the cases of the UBQP for $n = 3$ and the NPP for $n \in \{4, 5\}$, the generated samples are biased (i.e., the previous equivalence is not true).

In the second part of the thesis, we have focused on the theoretical analysis of EDAs designed for permutation-based COPs. Specifically, we have studied the convergence behavior of the Mallows-EDA. To do so, a mathematical modeling based on dynamical systems has been presented. Even if our proposed mathematical modeling has been used to study Mallows-EDA, our framework allows the reproducibility of the presented study to different distance-based exponential models and different fitness functions (beyond the ones studied in this thesis). Our proposed deterministic dynamical system studies the expected probability distribution

generated after one iteration of the algorithm. The dynamical system is composed of the 2-tournament selector operator and an approximation step based on the maximum likelihood estimation method.

Next, we have considered the presented framework to calculate the convergence behavior of the algorithm for several fitness functions. The studied fitness functions are the constant function, the *needle in a haystack* and the Mallows model, and the initial probability distributions have been the uniform distribution and the Mallows distribution (in total, six scenarios have been analyzed). In the most simple scenarios, the computation of one iteration of the algorithm has allowed us to find the limit behavior. For the most complex scenarios, to determine the limit behavior of the algorithm, the fixed points of the dynamical system and their attraction have been analyzed. The nature of the fixed points (for instance, when they are attracting points) provides information in relation to the possible trajectories of the algorithm.

Overall, the obtained results have been unexpected. When the function to optimize is constant, all Mallows probability distributions are fixed points. When the function to optimize is a *needle in a haystack* or a Mallows model and the initial probability distribution is the uniform distribution, the algorithm converges to a degenerate distribution centered at the optimal solution. Finally, when the function to optimize is a *needle in a haystack* or a Mallows model and the initial probability distribution is a Mallows model, then the algorithm can converge to a degenerate distribution (not necessarily centered at the optimal solution) or to a non-degenerate probability distribution. In any case, the relation between the initial probability distribution and the objective function completely determines the convergence behavior of the algorithm. Because of that, a classification of the convergence behavior of the algorithm regarding the parameters of the Mallows model has been given. As far as we know, the presented analysis has been the first theoretical analysis given in the literature for permutation-based EDAs, and it has shown the obstacles in achieving high quality theoretical results and the dissimilarities in comparison to the existing results in the literature for binary EDAs. Given its practical success, the results were by no means anticipated.

7

Future Work

This thesis has presented several contributions in three subfields related to COPs: (1) to analyze binary-based COPs in a "common framework", (2) to better comprehend permutation-based algorithms and (3) to approach the idilic goal of defining an association function such that, given a COP (instance), the "oracle" returns the most efficient algorithm to solve it. For each presented contribution, several extensions and research lines on theoretical and practical questions have been raised and they could be tackled as future work.

i) **Extensions of the results provided.**

a) *Computation of the Walsh decomposition of new binary-based COPs.*

In Chapter 2, we have computed the Walsh decomposition of the UBQP, the Max-Cut Problem and the NPP. Still, there is much work to do. For example, among all the binary-based COPs, the limitations generated by constrained binary-based COPs on the Walsh coefficients remains unclear and the challenge is how to incorporate the constraints of the problems in the Walsh decomposition to analyze it efficiently. For example, if we consider the use of slack variables and/or penalty coefficients to reformulate constrained problems such as unconstrained problems (several examples can be found in [15]), then the new parameters have an influence in the Walsh coefficients and there is a possibility that these new Walsh coefficients can "eclipse" the rest of coefficients, which makes it difficult to present an accurate analysis.

b) *Study and compare the artificial instances of the UBQP and the NPP.*

In Chapter 4, we have generated artificial instances of the UBQP and NPP by sampling coefficients uniformly at random. Knowing that the NPP can be described as a particular case of the UBQP, it remains to take into account the rankings generated by the NPP and to compare the hypervolume of the sampling region of each ranking with respect to the NPP and the UBQP. In addition, the cases $n > 3$ for the UBQP and $n > 5$ for the NPP need to be explored, and the generation of instances of other binary-based COPs remain as future work.

c) *Prove Conjecture 1.*

In Chapter 3, we have presented a conjecture that remains to be proved. In Appendix A two ideas to prove Conjecture 1 are shown. However, in both ideas, the proof of the final step is missing.

d) *Analyze Dyck Words of pseudo-Boolean functions of degree $m < n - 1$.*

In Example 10 (Chapter 3), we have shown that analyzing Dyck Words is not sufficient to determine the exact degree of a pseudo-Boolean function without exhaustively analyzing the system of inequalities defined by the ranking. Moreover, in Example 12, we present an example of the differences between Lemma 8 and Corollary 5. Consequently, the sufficient conditions of a ranking to be generated by an m-degree pseudo-Boolean function ($m < n - 1$) have yet to be presented.

ii) **Further analyses based on the obtained results.**

a) *Analyze equivalent binary-based COPs and their influence in the Walsh decomposition.*

A future challenge is the analysis of equivalent COPs and their influence in the Walsh decomposition. As mentioned in Chapter 2, for the NPP, we can directly study the fitness-function f or its equivalent (in terms of the generated ranking of solutions) f^2. For the function f, all the Walsh coefficients associated to an even number of variables are non-null, whereas for the function f^2 there are $n(n-1)/2+1$ non-null Walsh coefficients at most (which are the Walsh coefficients associated to zero or two variables). Therefore, two equivalent fitness functions can differ in the set of non-null Walsh coefficients. Considering that, it is intriguing to know how equivalent functions are represented in the Walsh decomposition and which is the minimum set of non-null Walsh coefficients to generate a specific ranking of solutions.

b) *Obtain a better comprehension of the rankings of solutions.*

It would be ideal to search for new "patterns" of the rankings of solutions or to find a different grouping of the instances (not only the symmetries observed in Chapter 4) in order to solve them efficiently. For example, in [96], the authors employ a dimension reduction technique over a subset of features to display the instances in a two dimensional space and to separate easy and hard instances. A similar procedure based on the words of a ranking could be interesting.

c) *Generate a fast computation procedure of the words of a ranking.*

An algorithm which efficiently computes the words of a ranking and checks if there is a Dyck Word would be interesting for calculating the minimum degree of the pseudo-Boolean function which generates a specific ranking. The algorithm would also allow us to compare the analysis of the words with the resolution of the system of inequalities defined by the ranking to check which method is more efficient (or in which cases) and to consider it for the generation of surrogate models.

d) *Search for practical applications of the characterization of pseudo-Boolean functions.*

It would be interesting to continue the presented characterization of pseudo-Boolean functions and to relate it with the studies presented in the literature in the novel fields such as Quantum Annealing [23, 90] and linear extensions of posets [13, 16, 59].

e) *Study permutation-based EDAs with finite populations.*

The proposed mathematical framework in Chapter 5 to study permutation-based EDAs and the convergence behavior about Mallows-EDA considers infinite populations. Even though the behavior of the presented algorithm with a finite population can be different from that predicted from the expectations, the variety of resulted convergence situations observed in the presented modeling shows the complexity of predicting the limit distributions of finite-population EDAs.

For a first comparison between the algorithm with finite and infinite populations, an EDA with Mallows model using finite populations and the Borda count [36] to estimate the central permutation

σ_0 could be applied and their performance contrasted. In addition, it is really intriguing to observe if other permutation-based EDAs or distance-based models achieve better convergence results and not many non-desirable solutions.

f) *Transfer the obtained theoretical results into practice.*

In the studied mathematical modeling of the Mallows-EDA (Chapter 5), it has been observed that the central permutation of the initial probability distribution determines which probability distributions can be learnt by the algorithm at each iteration. Then, for practical purposes, we propose a careful choice of the initial population. For example, a logical proposal is to generate individuals that are as far as possible from each other, expanding the initial search of the optimal solution. This proposal can be compared with the initialization presented in [17, 109], in which the authors apply a preliminary step so as to guide the algorithm to find the optimal solution.

g) *Compute the runtime analysis of Mallows-EDA.*

In Chapter 5, we have analyzed the convergence behavior of Mallows-EDA. If we are interested in the runtime analysis of the algorithm, it is important to take into account some knowledge that emerges from our analysis.

A first proposal of a runtime analysis is the following one. We have observed that the estimated spread parameter value at each iteration of the algorithm can be very critical. When the estimated spread parameter value change is big, the algorithm presents several scenarios in which the learned probability distributions can be significantly different among them (because the estimated central permutation is different in each case, for example) and the probability of sampling the optimal solution depends on it. On the contrary, when the estimated spread parameter value change is small, if the central permutation is not the optimal solution, the probability of reaching it will exponentially decrease with the spread parameter value. This observation may allow us to estimate the number of iterations required by the algorithm to converge to a model and when the researchers should modify the algorithm to escape from the expected tendency of the algorithm.

Another analysis we propose is, starting from different initial probability distributions, to check if there exists a number of iterations that ensures the probability to sample the optimal solution is higher than a value and to track the probability at each iteration. If the mentioned analysis is realized, we could connect its results with the presented results in the literature for binary EDAs and observe the similarities and differences between them.

8

Publications

The research work carried out during this thesis has produced the following contributions:

8.1 Referred Journals

- **I. Unanue**, M. Merino, J.A. Lozano. A mathematical analysis of EDAs with distance-based exponential models. *Memetic Computing*, 2022, vol. 14, n. 3, p. 305-334. DOI: 10.1007/s12293-022-00371-y.
- **I. Unanue**, M. Merino, J.A. Lozano. Characterization of rankings generated by pseudo-boolean functions. *Submitted to Swarm and Evolutionary Computation*, 2023.

8.2 International Conference Communications

- **I. Unanue**, M. Merino, J.A. Lozano. A mathematical analysis of EDAs with distance-based exponential models. *In: Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 429–430 (2019). DOI: 10.1145/3319619.3321969. ISBN: 978-1-4503-6748-6.
- **I. Unanue**, M. Merino, J.A. Lozano. A general framework based on Walsh decomposition for combinatorial optimization problems. *Proceedings of 2021 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2021. DOI: 10.1109/CEC45853.2021.9504699. ISBN: 978-1-7281-8394-7.
- **I. Unanue**, M. Merino, J.A. Lozano. The natural bias of artificial instances. *Accepted in: 2023 IEEE Congress on Evolutionary Computation (CEC)*. [Best Student Paper Award]

8.3 National Conference Communications

- **I. Unanue**, M. Merino, J.A. Lozano. Mathematical modeling and analysis of combinatorial optimization problems. *In XIX Conference of the Spanish Association for Artificial Intelligence (CAEPIA)*, 2021. ISBN: 978-84-09-30514-8. [Finalist of the category Best Thesis Project]

8.4 International Stay

- 3 months research stay as a visitor in 2022 (May-July) in BONUS Research Group, Inria-Lille Nord Europe, supervised by the team leader Professor Bilel Derbel.
  `https://www.inria.fr/en/centre-inria-lille-nord-europe`

8.5 Contribution to OEIS

- **I. Unanue**, M. Merino, J. A. Lozano. Sequence A307429 in *The On-Line Encyclopedia of Integer Sequences (2019)*, published electronically at http://oeis.org/A307429.

8.6 Awards

- Nominated for Best Thesis Project in XIX Conference of the Spanish Association for Artificial Intelligence (CAEPIA).
- Best Student Paper Award in 2023 IEEE Congress on Evolutionary Computation (CEC)

References

[1] Achlioptas, D., Naor, A., and Peres, Y. (2005). Rigorous Location of Phase Transitions in Hard Optimization Problems. *Nature*, 435(7043):759–764.

[2] Ahuja, R. K., Mehlhorn, K., Orlin, J., and Tarjan, R. E. (1990). Faster Algorithms for the Shortest Path Problem. *Journal of the ACM*, 37(2):213–223.

[3] Aigner, M. (1997). *Combinatorial Theory*. Springer Berlin Heidelberg.

[4] Ali, A. and Meilă, M. (2012). Experiments with Kemeny Ranking: What Works When? *Mathematical Social Sciences*, 64(1):28–40.

[5] Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press.

[6] Bäck, T., Rudolph, G., and Schwefel, H.-P. (1993). Evolutionary Programming and Evolution Strategies: Similarities and Differences. In *Proceedings of the 2nd Annual Conference on Evolutionary Programming*, pages 11–22.

[7] Baluja, S. (1994). Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning. Technical report, USA.

[8] Barahona, F., Jünger, M., and Reinelt, G. (1989). Experiments in Quadratic 0–1 Programming. *Mathematical Programming*, 44(1):127–137.

[9] Bartholdi, J., Tovey, C. A., and Trick, M. A. (1989). Voting Schemes for which It Can Be Difficult to Tell Who Won the Election. *Social Choice and Welfare*, 6(2):157–165.

[10] Beyer, H.-G. and Schwefel, H.-P. (2002). Evolution Strategies – A Comprehensive Introduction. *Natural Computing*, 1(1):3–52.

[11] Blickle, T. and Thiele, L. (1996). A Comparison of Selection Schemes used in Evolutionary Algorithms. *Evolutionary Computation*, 4(4):361–394.

[12] Blum, C. and Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, 35(3):268–308.

[13] Bochkov, I. A. and Petrov, F. V. (2021). The Bounds for the Number of Linear Extensions via Chain and Antichain Coverings. *Order*, 38(2):323–328.

[14] Boros, E., Crama, Y., and Rodríguez-Heck, E. (2020). Compact Quadratizations for Pseudo-Boolean Functions. *Journal of Combinatorial Optimization*, 39(3):687–707.

[15] Boros, E. and Hammer, P. L. (2002). Pseudo-Boolean Optimization. *Discrete Applied Mathematics*, 123(1):155–225.

[16] Brightwell, G. and Winkler, P. (1991). Counting Linear Extensions. *Order*, 8(3):225–242.

[17] Ceberio, J., Irurozki, E., Mendiburu, A., and Lozano, J. A. (2014). A Distance-Based Ranking Model Estimation of Distribution Algorithm for the Flowshop Scheduling Problem. *IEEE Transactions on Evolutionary Computation*, 18(2):286–300.

[18] Ceberio, J., Mendiburu, A., and Lozano, J. A. (2011). Introducing the Mallows Model on Estimation of Distribution Algorithms. In *Proceedings of the 18th International Conference on Neural Information Processing - Volume Part II*, pages 461–470. Springer Berlin Heidelberg.

[19] Ceberio, J., Mendiburu, A., and Lozano, J. A. (2017). Are We Generating Instances Uniformly at Random? In *2017 IEEE Congress on Evolutionary Computation*, pages 1645–1651. IEEE.

[20] Chicano, F., Whitley, L. D., and Alba, E. (2011). A Methodology to Find the Elementary Landscape Decomposition of Combinatorial Optimization Problems. *Evolutionary Computation*, 19(4):597–637.

[21] Choi, S.-S., Jung, K., and Moon, B.-R. (2009). Lower and Upper Bounds for Linkage Discovery. *IEEE Transactions on Evolutionary Computation*, 13(2):201–216.

[22] Christie, L. A. (2016). *The Role of Walsh Structure and Ordinal Linkage in the Optimisation of Pseudo-Boolean Functions under Monotonicity Invariance.* PhD thesis, Robert Gordon University.

[23] Cruz-Santos, W., Venegas-Andraca, S. E., and Lanzagorta, M. (2019). A QUBO Formulation of Minimum Multicut Problem Instances in Trees for D-Wave Quantum Annealers. *Scientific Reports*, 9(1):17216.

[24] Deutsch, E. (1999). Dyck Path Enumeration. *Discrete Mathematics*, 204(1):167–202.

[25] Devroye, L., Györfi, L., and Lugosi, G. (1996). *A Probabilistic Theory of Pattern Recognition.* Springer New York.

[26] Doerr, B. (2021). The Runtime of the Compact Genetic Algorithm on Jump Functions. *Algorithmica*, 83(10):3059–3107.

[27] Droste, S. (2006). A Rigorous Analysis of the Compact Genetic Algorithm for Linear Functions. *Natural Computing*, 5(3):257–283.

[28] Du, K.-L. and Swamy, M. N. S. (2016). *Search and Optimization by Metaheuristics.* Springer.

[29] Duchon, P. (2000). On the Enumeration and Generation of Generalized Dyck Words. *Discrete Mathematics*, 225(1):121–135.

[30] Echegoyen, C., Mendiburu, A., Santana, R., and Lozano, J. A. (2012). Toward Understanding EDAs Based on Bayesian Networks Through a Quantitative Analysis. *IEEE Transactions on Evolutionary Computation*, 16(2):173–189.

[31] Echegoyen, C., Mendiburu, A., Santana, R., and Lozano, J. A. (2013). On the Taxonomy of Optimization Problems under Estimation of Distribution Algorithms. *Evolutionary Computation*, 21(3):471–495.

[32] Echegoyen, C., Santana, R., Mendiburu, A., and Lozano, J. A. (2015). Comprehensive Characterization of the Behaviors of Estimation of Distribution Algorithms. *Theoretical Computer Science*, 598:64–86.

[33] Eiben, A. and Smith, J. (2015). *Introduction to Evolutionary Computing.* Springer Berlin Heidelberg.

[34] Elorza, A., Hernando, L., and Lozano, J. A. (2022). Transitions from P to NP-Hardness: The Case of the Linear Ordering Problem. In *2022 IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE.

[35] Elorza, A., Hernando, L., and Lozano, J. A. (2023). Characterizing Permutation-Based Combinatorial Optimization Problems in Fourier Space. *Evolutionary Computation*, pages 1–37.

[36] Emerson, P. (2013). The Original Borda Count and Partial Voting. *Social Choice and Welfare*, 40(2):353–358.

[37] Feng, Y.-Q. (2006). Automorphism Groups of Cayley Graphs on Symmetric Groups with Generating Transposition Sets. *Journal of Combinatorial Theory, Series B*, 96(1):67–72.

[38] Feo, T. A. and Resende, M. G. (1989). A Probabilistic Heuristic for a Computationally Difficult Set Covering Problem. *Operations Research Letters*, 8(2):67–71.

[39] Fligner, M. A. and Verducci, J. S. (1986). Distance Based Ranking Models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(3):359–369.

[40] Fogel, D. B. (1998). Artificial Intelligence through Simulated Evolution. In *Evolutionary Computation: The Fossil Record*, pages 227–296. Wiley-IEEE Press.

[41] Fomin, F. V. and Kratsch, D. (2010). *Exact Exponential Algorithms*. Springer Berlin Heidelberg.

[42] Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., USA.

[43] Gendreau, M. and Potvin, J.-Y. (2005). Metaheuristics in Combinatorial Optimization. *Annals of Operations Research*, 140(1):189–213.

[44] Gendreau, M. and Potvin, J.-Y. (2010). *Handbook of Metaheuristics*. Springer US.

[45] Goldberg, D. E. (1989a). Genetic Algorithms and Walsh Functions: Part I, A Gentle Introduction. *Complex Systems*, 3:129–152.

[46] Goldberg, D. E. (1989b). Genetic Algorithms and Walsh Functions: Part II, Deception and Its Analysis. *Complex Systems*, 3:153–171.

[47] Goldberg, D. E. (1989c). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., USA.

[48] González, C., Lozano, J. A., and Larrañaga, P. (2000). Analyzing the Population Based Incremental Learning Algorithm by Means of Discrete Dynamical Systems. *Complex Systems*, 12(4):465–479.

[49] González, C., Lozano, J. A., and Larrañaga, P. (2002). Mathematical Modeling of Discrete Estimation of Distribution Algorithms. In *Estimation of Distribution Algorithms*, pages 147–163. Springer US.

[50] Hammer, P. L. and Rudeanu, S. (1968). *Boolean Methods in Operations Research and Related Areas*. Springer Berlin Heidelberg.

[51] Harik, G. R., Lobo, F. G., and Goldberg, D. E. (1999). The Compact Genetic Algorithm. *IEEE Transactions on Evolutionary Computation*, 3(4):287–297.

[52] Heckendorn, R. B. and Wright, A. H. (2004). Efficient Linkage Discovery by Limited Probing. *Evolutionary Computation*, 12(4):517–545.

[53] Henderson, K. W. (1964). Some Notes on the Walsh Functions. *IEEE Transactions on Electronic Computers*, EC-13(1):50–52.

[54] Hernando, L., Mendiburu, A., and Lozano, J. A. (2019). Characterising the Rankings Produced by Combinatorial Optimisation Problems and Finding Their Intersections. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 266–273. ACM.

[55] Hernando, L., Mendiburu, A., and Lozano, J. A. (2020). Journey to the Center of the Linear Ordering Problem. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 201–209. ACM.

[56] Höhfeld, M. and Rudolph, G. (1997). Towards a Theory of Population-Based Incremental Learning. In *1997 IEEE International Conference on Evolutionary Computation*, pages 1–5. IEEE.

[57] Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT press.

[58] Jünger, M., Lobe, E., Mutzel, P., Reinelt, G., Rendl, F., Rinaldi, G., and Stollenwerk, T. (2019). Performance of a Quantum Annealer for Ising Ground State Computations on Chimera Graphs. *arXiv preprint arXiv:1904.11965*.

[59] Kangas, K., Hankala, T., Niinimäki, T., and Koivisto, M. (2016). Counting Linear Extensions of Sparse Posets. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pages 603–609. AAAI Press.

[60] Kauffman, S. A. et al. (1993). *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press.

[61] Kendall, M. G. (1938). A New Measure of Rank Correlation. *Biometrika*, 30(1/2):81–93.

[62] Kochenberger, G., Hao, J.-K., Glover, F., Lewis, M., Lü, Z., Wang, H., and Wang, Y. (2014). The Unconstrained Binary Quadratic Programming Problem: A Survey. *Journal of Combinatorial Optimization*, 28(1):58–81.

[63] Kochenberger, G. A., Glover, F., Alidaee, B., and Rego, C. (2004). A Unified Modeling and Solution Framework for Combinatorial Optimization Problems. *OR Spectrum*, 26(2):237–250.

[64] Koza, J. R. (1994). Genetic Programming as a Means for Programming Computers by Natural Selection. *Statistics and Computing*, 4(2):87–112.

[65] Krejca, M. S. and Witt, C. (2020a). Lower Bounds on the Run Time of the Univariate Marginal Distribution Algorithm on OneMax. *Theoretical Computer Science*, 832:143–165.

[66] Krejca, M. S. and Witt, C. (2020b). Theory of Estimation-of-Distribution Algorithms. In *Theory of Evolutionary Computation*, pages 405–442. Springer.

[67] Kruskal, J. B. (1956). On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, 7(1):48–50.

[68] Kushilevitz, E. and Mansour, Y. (1993). Learning Decision Trees Using the Fourier Spectrum. *SIAM Journal on Computing*, 22(6):1331–1348.

[69] Laporte, G. (1992). The Traveling Salesman Problem: An Overview of Exact and Approximate Algorithms. *European Journal of Operational Research*, 59(2):231–247.

[70] Larrañaga, P. and Lozano, J. A. (2002). *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, volume 2. Springer Science & Business Media.

[71] Lehre, P. K. and Nguyen, P. T. H. (2019). Runtime Analysis of the Univariate Marginal Distribution Algorithm under Low Selective Pressure and Prior Noise. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1497–1505. ACM.

[72] Lengler, J., Sudholt, D., and Witt, C. (2018). Medium Step Sizes are Harmful for the Compact Genetic Algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1499–1506. ACM.

[73] Leprêtre, F., Verel, S., Fonlupt, C., and Marion, V. (2019). Walsh Functions as Surrogate Model for Pseudo-Boolean Optimization Problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 303–311. ACM.

[74] Liefooghe, A., Verel, S., and Hao, J.-K. (2014). A Hybrid Metaheuristic for Multiobjective Unconstrained Binary Quadratic Programming. *Applied Soft Computing*, 16:10–19.

[75] Macready, W. G. and Wolpert, D. H. (1996). What Makes an Optimization Problem Hard? *Complexity*, 1(5):40–46.

[76] Mahdavi Pajouh, F., Balasundaram, B., and Prokopyev, O. A. (2013). On Characterization of Maximal Independent Sets via Quadratic Optimization. *Journal of Heuristics*, 19(4):629–644.

[77] Mahnig, T. and Mühlenbein, H. (2000). Mathematical Analysis of Optimization Methods Using Search Distributions. In *Proceedings of the Genetic and Evolutionary Computation Conference Workshop Program*, pages 205–208. ACM.

[78] Mallows, C. L. (1957). Non-Null Ranking Models. *Biometrika*, 44(1/2):114–130.

[79] Manes, K., Tasoulas, I., Sapounakis, A., and Tsikouras, P. (2019). Counting Pairs of Noncrossing Binary Paths: A Bijective Approach. *Discrete Mathematics*, 342(2):352–359.

[80] Meilă, M., Phadnis, K., Patterson, A., and Bilmes, J. (2007). Consensus Ranking under the Exponential Model. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*, pages 285–294. AUAI Press.

[81] Mertens, S. (1998). Phase Transition in the Number Partitioning Problem. *Physical Review Letters*, 81(20):4281–4284.

[82] Mockus, J., Eddy, W., Mockus, A., Mockus, L., and Reklaitis, G. (1997). Examples of Continuous Optimization. In *Bayesian Heuristic Approach to Discrete and Global Optimization: Algorithms, Visualization, Software, and Applications*, pages 71–82. Springer US.

[83] Mühlenbein, H. (2008). Convergence of Estimation of Distribution Algorithms for Finite Samples. Technical report, Germany.

[84] Mühlenbein, H. and Mahnig, T. (1999). FDA-A Scalable Evolutionary Algorithm for the Optimization of Additively Decomposed Functions. *Evolutionary Computation*, 7(4):353–376.

[85] Mühlenbein, H. and Paaβ, G. (1996). From Recombination of Genes to the Estimation of Distributions I. Binary Parameters. In *Parallel Problem Solving from Nature – PPSN IV*, pages 178–187. Springer Berlin Heidelberg.

[86] Ocal, O., Kadhe, S., and Ramchandran, K. (2019). Low-Degree Pseudo-Boolean Function Recovery Using Codes. In *2019 IEEE International Symposium on Information Theory*, pages 1207–1211. IEEE.

[87] OEIS Foundation Inc. (1964). The On-Line Encyclopedia of Integer Sequences. *https://oeis.org*.

[88] Osman, I. H. and Laporte, G. (1996). Metaheuristics: A Bibliography. *Annals of Operations Research*, 63(5):511–623.

[89] Papadimitriou, C. H. and Steiglitz, K. (1998). *Combinatorial Optimization: Algorithms and Complexity*. Courier Corporation.

[90] Pastorello, D. and Blanzieri, E. (2019). Quantum Annealing Learning Search for Solving QUBO Problems. *Quantum Information Processing*, 18:1–17.

[91] Pelikan, M., Sastry, K., and Goldberg, D. E. (2002). Scalability of the Bayesian Optimization Algorithm. *International Journal of Approximate Reasoning*, 31(3):221–258.

[92] Pérez-Rodríguez, R. and Hernández-Aguirre, A. (2019). A Hybrid Estimation of Distribution Algorithm for the Vehicle Routing Problem with Time Windows. *Computers & Industrial Engineering*, 130:75–96.

[93] Prim, R. C. (1957). Shortest Connection Networks and Some Generalizations. *The Bell System Technical Journal*, 36(6):1389–1401.

[94] Reed, I. S. (1954). A Class of Multiple-Error-Correcting Codes and the Decoding Scheme. *IEEE Transactions on Information Theory*, 4(4):38–49.

[95] Shapiro, J. L. (2005). Drift and Scaling in Estimation of Distribution Algorithms. *Evolutionary Computation*, 13(1):99–123.

[96] Smith-Miles, K., Baatar, D., Wreford, B., and Lewis, R. (2014). Towards Objective Measures of Algorithm Performance Across Instance Space. *Computers & Operations Research*, 45:12–24.

[97] Smith-Miles, K. and Lopes, L. (2012). Measuring Instance Difficulty for Combinatorial Optimization Problems. *Computers & Operations Research*, 39(5):875–889.

[98] Stadler, P. F., Hordijk, W., and Fontanari, J. F. (2003). Phase Transition and Landscape Statistics of the Number Partitioning Problem. *Physical Review E*, 67(5):056701.

[99] Sturm, C. (2009). Mémoire sur la Résolution des Équations Numériques. In *Collected Works of Charles François Sturm*, pages 345–390. Springer.

[100] Sylvester, J. J. (1867). LX. Thoughts on Inverse Orthogonal Matrices, Simultaneous Signsuccessions, and Tessellated Pavements in Two or More Colours, with Applications to Newton's Rule, Ornamental Tile-Work, and the Theory of Numbers. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 34(232):461–475.

[101] Talbi, E.-G. (2009). *Metaheuristics: From Design to Implementation*, volume 74. John Wiley & Sons.

[102] Thompson, T. M. (1983). *From Error-Correcting Codes Through Sphere Packings to Simple Groups*. Number 21. Mathematical Association of America.

[103] Tsutsui, S. (2006). Node Histogram vs. Edge Histogram: A Comparison of Probabilistic Model-Building Genetic Algorithms in Permutation Domains. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1939–1946. IEEE.

[104] van De Vel, M. L. J. (1993). *Theory of Convex Structures*, volume 50. Elsevier.

[105] Venegas-Andraca, S. E., Cruz-Santos, W., McGeoch, C., and Lanzagorta, M. (2018). A Cross-Disciplinary Introduction to Quantum Annealing-Based Algorithms. *Contemporary Physics*, 59(2):174–197.

[106] Verel, S., Derbel, B., Liefooghe, A., Aguirre, H., and Tanaka, K. (2018). A Surrogate Model based on Walsh Decomposition for Pseudo-Boolean Functions. In *Parallel Problem Solving from Nature – PPSN XV*, pages 181–193. Springer International Publishing.

[107] Vose, M. D. (1999). *The Simple Genetic Algorithm: Foundations and Theory*, volume 12. MIT Press.

[108] Walsh, J. L. (1923). A Closed Set of Normal Orthogonal Functions. *American Journal of Mathematics*, 45(1):5–24.

[109] Wang, F., Li, Y., Zhou, A., and Tang, K. (2019). An Estimation of Distribution Algorithm for Mixed-Variable Newsvendor Problems. *IEEE Transactions on Evolutionary Computation*, 24(3):479–493.

[110] Whitley, D. (1994). A Genetic Algorithm Tutorial. *Statistics and Computing*, 4(2):65–85.

[111] Witt, C. (2017). Upper Bounds on the Runtime of the Univariate Marginal Distribution Algorithm on OneMax. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1415–1422. ACM.

[112] Witt, C. (2018). Domino Convergence: Why One Should Hill-Climb on Linear Functions. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1539–1546. ACM.

[113] Witt, C. (2021). On Crossing Fitness Valleys with Majority-Vote Crossover and Estimation-of-Distribution Algorithms. In *Proceedings of the 16th ACM/SIGEVO Conference on Foundations of Genetic Algorithms*, pages 1–15. ACM.

[114] Wolpert, D. H. and Macready, W. G. (1997). No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82.

[115] Wu, Z., Kolonko, M., and Möhring, R. H. (2017). Stochastic Runtime Analysis of the Cross-Entropy Algorithm. *IEEE Transactions on Evolutionary Computation*, 21(4):616–628.

[116] Yu, D.-P. and Kim, Y.-H. (2020). On the Effect of Walsh/Fourier Transform in Surrogate-Assisted Genetic Algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 235–236. ACM.

[117] Yu, T.-L., Sastry, K., Goldberg, D. E., and Pelikan, M. (2007). Population Sizing for Entropy-Based Model Building in Discrete Estimation of Distribution Algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 601–608. ACM.

[118] Zhang, Q. (2004). On Stability of Fixed Points of Limit Models of Univariate Marginal Distribution Algorithm and Factorized Distribution Algorithm. *IEEE Transactions on Evolutionary Computation*, 8(1):80–93.

[119] Zhang, Q. and Mühlenbein, H. (2004). On the Convergence of a Class of Estimation of Distribution Algorithms. *IEEE Transactions on Evolutionary Computation*, 8(2):127–136.

Appendices

A Observations about Conjecture 1

In this appendix, we present two observations about Conjecture 1 (presented in Chapter 3). In Section A.1, we explain the influence of the coefficients of the 2-degree pseudo-Boolean functions to generate rankings of solutions. In Section A.2, we analyze the words of the rankings to study which the feasible rankings are. We believe that these observations could be helpful to prove Conjecture 1 without an exhaustive verification.

A.1 Analysis of the coefficients of 2-degree pseudo-Boolean functions

Let us consider that the proof of Conjecture 1 could be done by induction. Let us consider the case $n = 3$ and f a pseudo-Boolean function of degree $m = 2$. Let us analyze the coefficients of f ($\{a_1, a_2, a_3, a_{12}, a_{13}, a_{23}\}$) and their influence to generate rankings of solutions. For a better comprehension of the argument below, Figure A.1 shows the geometric relations between coefficients, solutions, their parity and fitness function values. Note that the 3-dimensional representation could be extended for larger dimensions. The fitness function values of the 8 solutions are the following:

x	$f(x)$
111	$a_1 + a_2 + a_3 + a_{12} + a_{13} + a_{23}$
110	$a_2 + a_3 + a_{23}$
101	$a_1 + a_3 + a_{13}$
100	a_3
011	$a_1 + a_2 + a_{12}$
010	a_2
001	a_1
000	0

(A.1)

In Figure A.1a, we present a cube whose vertexes are all the fitness function values of the solutions and in which two vertexes are connected if the Hamming distance between the solutions is 1; that is to say, if two solutions differ in 1 bit, their fitness function values are connected. In Figure A.1b, we show the parity of each solution (considered in Figure A.1a). We observe that, according to the parity of zeros, the graph is a

bigraph (perfectly balanced according to the partition $\{\mathcal{E}, \mathcal{O}\}$). Our analysis will focus on the reorderings of the fitness function values (Figure A.1a) and their implications on the generated words (Figure A.1b).

Let us divide the set of coefficients in two groups: the set of coefficients that depend on a single bit, $\{a_1, a_2, a_3\}$; and the set of coefficients that depend on two bits, $\{a_{12}, a_{13}, a_{23}\}$. The analysis starts from the rankings that can be generated with the former group of bits, and then we add the coefficients of the latter group to generate and to study all the rankings generated by pseudo-Boolean functions of degree 2. For each added coefficient, we observe which new rankings are generated and we check if their words are not Dyck Words.

In Figure A.1c, we show the coefficients a_1, a_2, a_3 that influence in the fitness function values of Figure A.1a. The study of Figure A.1c is analogous to the study of pseudo-Boolean functions of degree 1 ($a_{12} = a_{13} = a_{23} = 0$). From this figure, several observations can be made:

a) According to each edge, if we define a relative order between two values (if two adjacent vertex values are compared and an inequality is fixed), then the same relative order must be kept for all its parallel edges. The formal definition is the following one: for any a_i and $s \subseteq \{a_j, a_k\}$, if $a_i > 0$ ($a_i < 0$), then $a_i + \sum_{a \in s} a > \sum_{a \in s} a$ ($a_i + \sum_{a \in s} a < \sum_{a \in s} a$). This implies that several relative orderings between even and odd solutions are completely connected to other even and odd solutions (the number of connected relative orderings depends on the cardinality of s). Furthermore, if we want to swap two adjacent solutions in a ranking (reverse a relative order), all the solutions that are dependent on the same relative order must be swapped in the ranking at the same time.

   In Figure A.1c, we have colored the edges in such a way that the edges of the same color are parallel and therefore they must keep the same fixed relative order.

b) For any values of the coefficients a_1, a_2, a_3, if a vertex of the cube (Figure A.1c) is the maximum value, then the opposite vertex (the vertex at Hamming distance 3) is the minimum value.

Therefore, if we consider the parity of Figure A.1b, there are only four possible words generated for 1-degree pseudo-Boolean functions: $EOOOEEEO$, $EOOEOEEO$, $OEEEOOOE$ and $OEEOEOOE$. In any case, the generated word is not a Dyck Word. In addition, this scenario proves that the number of different rankings that can be generated by 1-degree pseudo-Boolean functions is 96 (12 possible rankings starting from each vertex).

From this scenario, to prove Conjecture 1 for $n = 3$ without an exhaustive verification, it remains to be proved that for any coefficient values a_{12}, a_{13}, a_{23}, the addition of these coefficients to any ranking generated by a pseudo-Boolean function of degree 1 does not generate a Dyck Word. To do so, the new words generated by adding the remaining coefficients one by one are analyzed.

The influence of the coefficients a_{ij} is analogous to the formal definition of the relative orders defined by the coefficients a_i, a_j. For example, for $k \neq i, j$, if $a_i > a_i + a_j + a_{ij}$ ($a_i < a_i + a_j + a_{ij}$), then $a_i + a_k > a_i + a_j + a_k + a_{ij}$ ($a_i + a_k < a_i + a_j + a_k + a_{ij}$).

The first case is to consider any 1-degree pseudo-Boolean function f and to add a coefficient a_{ij}: $g(x) = f(x) + a_{ij}x_ix_j$. In this scenario, the fitness function values of g that differ from f are $g(111)$ (specifically, $g(111) = f(111) + a_{ij}$) and $g(x)$ such that $x_i = x_j = x_k + 1 = 1$ (specifically, $g(x) = f(x) + a_{ij}$). Particularly, the "critical" values in which the addition of the coefficient a_{ij} changes the ranking of solutions are $-a_i$, $-a_j$, $-a_i - a_j$, $a_k - a_i$, $a_k - a_j$ and $a_k - a_i - a_j$. For the first values, a_{ij} causes two swaps in the ranking, whereas the last value implies one swap. It can be observed that all the rankings generated by g have no Dyck Words.

The next step (and the most difficult one) is to consider any 1-degree pseudo-Boolean function f and to add two coefficients a_{ij}, a_{ik}: $g(x) = f(x) + a_{ij}x_ix_j + a_{ik}x_ix_k$. The difficulty of this case, in comparison to the previous case, is that there exist some adjacent swaps in the rankings that are influenced by the sum $a_{ij} + a_{ik}$ and consequently by the value $g(111)$. So, some relative orders might be lost. The final step is to add the three coefficients $\{a_{12}, a_{13}, a_{23}\}$ and combine the previous analyses. This combination would prove that the rankings whose word is not a Dyck Word can be generated by a 2-degree pseudo-Boolean function.

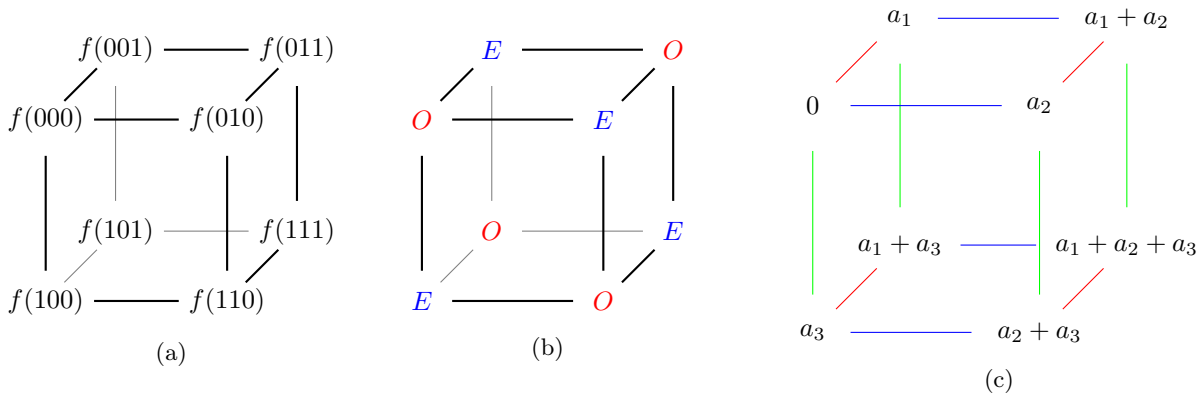


Fig. A.1: Graphical representation for the case $n = 3$. (a) Fitness function values and edges defined by the Hamming distance; (b) Parity of the solutions; (c) Exact fitness function values when $m = 1$.

A.2 Construction of rankings without Dyck Words

Let $n = 3$. Let f be a 2-degree pseudo-Boolean function defined by $\sum_{i=1}^{2}\binom{3}{i} = 6$ real coefficients and $f(111) = \sum_{x\in\mathcal{O}} f(x) - \sum_{x\in\mathcal{E}\setminus\{111\}} f(x)$. By definition, a ranking r without the solution 111 can always be generated by an appropriate selection of the coefficients (each fitness function value apart from 0 is defined with an independent coefficient that allows the solution to be fixed in the desired position):

x	$f(x)$
111	$f(110) + f(101) + f(011) - f(100) - f(010) - f(001)$
110	$f(100) + f(010) + a_{23}$
101	$f(100) + f(001) + a_{13}$
100	a_3
011	$f(010) + f(001) + a_{12}$
010	a_2
001	a_1
000	0

(A.2)

Moreover, because multiplying all the coefficients by any positive real value keeps the ranking invariant, each ranking can be generated by infinite possible selections of the coefficients. Furthermore, it can be ensured that the difference between the fitness function values of two adjacent solutions (with respect to the ranking)

to be higher or lower than a real value. Let us denote r^* as the ranking r without the solution 111 and W^* as the word generated by r^*. Proposition 2 is proved if and only if, for any r^*, by an appropriate selection of the coefficients, the solution 111 can be inserted at any position that generates a ranking r whose word is not a Dyck Word.

Starting from the word W^* of a ranking r^*, first we observe in which positions of r^* the insertion of the solution 111 generates a ranking r such that W is not a Dyck Word. Specifically, the study of the sequence $\Delta_1, \ldots, \Delta_7$ in W^* allows us to know the exact positions where the solution 111 can be inserted to generate r. For any W^*, there are four possible scenarios.

(a) If $\Delta_i < 0$, for all $i \in \{1, \ldots, 7\}$, then inserting the solution 111 at the top of the ranking r^* and defining r, the word W is not a Dyck Word.

(b) If $W^* = [O \;\; E \;\; O \;\; E \;\; O \;\; E \;\; O]^T$, then inserting the solution 111 at any position except for the top and the bottom of the ranking r^* generates a ranking r such that W is not a Dyck Word.

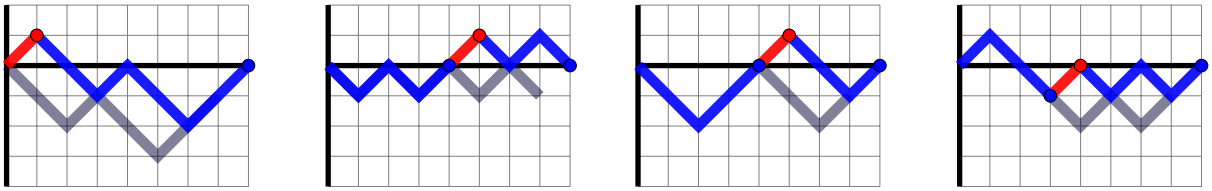
(c) If $W^* \neq [O \;\; E \;\; O \;\; E \;\; O \;\; E \;\; O]^T$ and there exists an integer i such that $\Delta_i = -1$ and $\Delta_{i+1} = 0$, then inserting the solution 111 at any position $j \leq i + 2$ of the ranking r^* generates a ranking r such that W is not a Dyck Word.

(d) If there exists at least one value $i \in \{1, \ldots, 5\}$ such that $\Delta_i = 1$, $\Delta_{i+1} = 0$ and $\Delta_{i+2} = -1$, then inserting the solution 111 at any position $j \geq i + 2$ of the ranking r^* generates a ranking r whose word is not a Dyck Word.

In Figure A.2, one example of each of the mentioned scenarios is displayed. In Figure A.2a, the Δ_i values of r^* are negative values, and inserting the solution 111 at the top of the ranking, we generate r whose word is not a Dyck Word. In Figure A.2b, inserting the solution 111 in the fifth position we generate r such that W is not a Dyck Word. In Figure A.2c, $\Delta_3 = -1$ and $\Delta_4 = 0$, so inserting the solution 111 in the fifth position we generate r whose word is not a Dyck Word. Finally, in Figure A.2d, $\Delta_1 = 1$, $\Delta_2 = 0$ and $\Delta_3 = -1$, which implies that inserting the solution 111 in the fourth position, the word of the generated ranking r is not a Dyck Word.

Consequently, for any r^*, there always exists a position to insert 111 and to generate a ranking r whose word is not a Dyck Word. In addition, if inserting the solution 111 at the top or at the bottom of the ranking r^* generates a ranking r such that W is not a Dyck Word, then the solution 111 can be inserted at any position of the ranking r^* and will generate a ranking without a Dyck Word.

Finally, it remains to be explained why the solution 111 can be inserted in the desired position to generate the ranking r (such that W is not a Dyck Word): that is to say, why the ranking r is possible to be generated. As previously mentioned, the facts that multiplying the coefficients keeps the same ranking and that we can ensure a minimum distance between two adjacent values allow us to increase or decrease some specific coefficients without changing r^* and to increase or decrease the value $f(111)$.



(a) $EOOEOOEE$ (b) $OEOEEOEO$ (c) $OOEEEOOE$ (d) $EOOEOEOE$

Fig. A.2: Posible r^* scenarios and how to generate a ranking r without a Dyck Word.

B Study of Equation (5.17) and $g(\theta)$

In this appendix, several observations and properties about the right-hand side of Equation (5.17) (presented in Chapter 5) are studied and commented.

Proposition 3.

$$g^*(\theta) = \frac{n-1}{e^\theta - 1} - \sum_{i=1}^{n-1} \frac{n-i+1}{e^{(n-i+1)\theta} - 1} = \frac{1}{e^\theta - 1}\left(n - 1 - \sum_{i=1}^{n-1} \frac{n-i+1}{e^{(n-i)\theta} + e^{(n-i-1)\theta} + \cdots + 1}\right) \tag{B.3}$$

is a continuous function defined in $\mathbb{R}\backslash\{0\}$. *Moreover,*

$$\lim_{\theta \to 0} g^*(\theta) = \frac{1}{2}\binom{n}{2}. \tag{B.4}$$

Proof. First of all, the function g^* is not defined when $\theta = 0$. Moreover, the continuity of the function is trivial (combination of scalar and exponential functions and the denominator is never zero).

Let us show $\lim_{\theta\to 0} g^*(\theta) = \binom{n}{2}/2$. Let us prove the limit by means of L'Hôpital's rule.

$$\begin{aligned}
&\lim_{\theta \to 0}\left(\frac{n-1}{e^\theta - 1} - \sum_{i=1}^{n-1} \frac{n-i+1}{e^{(n-i+1)\theta} - 1}\right) = \lim_{\theta \to 0}\left(\frac{1}{e^\theta - 1}\left(n - 1 - \sum_{i=1}^{n-1} \frac{n-i+1}{e^{(n-i)\theta} + e^{(n-i-1)\theta} + \cdots + 1}\right)\right) \\
\overset{\text{L'Hôpital}}{=}\ &\lim_{\theta \to 0} \frac{1}{e^\theta}\left(\sum_{i=1}^{n-1} \frac{(n-i+1)\cdot((n-i)e^{(n-i)\theta} + \cdots + e^\theta)}{(e^{(n-i)\theta} + e^{(n-i-1)\theta} + \cdots + 1)^2}\right) \\
=\ &\sum_{i=1}^{n-1} \frac{(n-i) + (n-i-1) + \cdots + 1}{n-i+1} = \sum_{i=1}^{n-1} \frac{n-i}{2} = \frac{1}{2}\binom{n}{2}.
\end{aligned} \tag{B.5}$$

□

Therefore, we have a function which is defined in $\mathbb{R}\backslash\{0\}$ and has a limit when θ tends to 0. So, the extension of the right-hand side of Equation (5.17) can be defined in the following way:

$$g(\theta) = \begin{cases} g^*(\theta), & \text{if } \theta \neq 0 \\ \frac{1}{2}\binom{n}{2}, & \text{if } \theta = 0. \end{cases} \tag{B.6}$$

Proposition 4. *$g(\theta)$ is a continuous decreasing function, $g(\theta) + g(-\theta) = \binom{n}{2}$ and*

$$\lim_{\theta \to +\infty} g(\theta) = 0. \tag{B.7}$$

Proof. By definition of $g(\theta)$ and Proposition 3, it is trivial to observe that $g(\theta)$ is a continuous function. Moreover, for any value $\theta \neq 0$,

$$g'(\theta) = \frac{(1-n)e^{\theta}}{(e^{\theta}-1)^2} + \sum_{i=1}^{n-1} \left(\frac{(n-i+1)^2 (e^{(n-i+1)\theta})}{(e^{(n-i+1)\theta}-1)^2} \right). \tag{B.8}$$

To prove $g'(\theta)$ is always a negative value ($\theta \neq 0$), we will prove the following inequality:

$$-\frac{e^{\theta}}{(e^{\theta}-1)^2} + \frac{(n-i+1)^2 (e^{(n-i+1)\theta})}{(e^{(n-i+1)\theta}-1)^2} < 0 \ , \ \forall i = 1, \ldots, n-1. \tag{B.9}$$

Developing the expression,

$$\begin{aligned} & e^{\theta} \left(e^{(n-i)\theta} + e^{(n-i-1)\theta} + \cdots + 1 \right)^2 - (n-i+1)^2 (e^{(n-i+1)\theta}) > 0 \ , \ \forall i = 1, \ldots, n-1 \\ \iff & \sum_{k_1+k_2+\cdots+k_{n-i+1}=2} \binom{2}{k_1, k_2, \ldots, k_{n-i+1}} (e^{(n-i)\theta})^{k_1} (e^{(n-i-1)\theta})^{k_2} \cdots 1^{k_{n-i+1}} - (n-i+1)^2 (e^{(n-i)\theta}) > 0 \ , \ \forall i = 1, \ldots, n-1 \\ \iff & 1 + 2e^{\theta} + \cdots + (n-i)e^{(n-i-1)\theta} - (n-i+1)(n-i)e^{(n-i)\theta} + (n-i)e^{(n-i+1)\theta} + \cdots + e^{2(n-i)\theta} > 0 \ , \ \forall i = 1, \ldots, n-1. \end{aligned} \tag{B.10}$$

and this is always true (bear in mind that the exponential function is always positive). A direct way to see that the previous inequality holds is considering the next inequality:

$$\begin{aligned} & e^{k\theta} + e^{(2(n-i)-k)\theta} - 2e^{(n-i)\theta} > 0 \ , \ \forall k = 0, \ldots, n-i-1 \\ \iff & 1 + e^{(2(n-i-k))\theta} > 2e^{(n-i-k)\theta} \ , \ \forall k = 0, \ldots, n-i-1 \\ \iff & \frac{e^{-(n-i-k)\theta} + e^{(n-i-k)\theta}}{2} = \cosh(n-i-k) > 1 \ , \ \forall k = 0, \ldots, n-i-1. \end{aligned} \tag{B.11}$$

In Proposition 6 *(v)*, a similar result is mentioned.

Now let us prove that $g(\theta) + g(-\theta) = \binom{n}{2}$. By definition of $g(\theta)$, the case $\theta = 0$ is trivial, so let us calculate for the rest of values.

$$g(\theta)+g(-\theta) = n-1\left(\frac{1}{e^{\theta}-1}+\frac{1}{e^{-\theta}-1}\right)-\sum_{i=1}^{n-1}(n-i+1)\left(\frac{1}{e^{(n-i+1)\theta}-1}+\frac{1}{e^{-(n-i+1)\theta}-1}\right)$$
$$= n-1(-1)-\sum_{i=1}^{n-1}(n-i+1)(-1)$$
$$= 1-n+\sum_{i=1}^{n-1}(n-i+1)=\sum_{i=1}^{n-1}(n-i)=\frac{n(n-1)}{2}=\binom{n}{2}. \tag{B.12}$$

Finally, the limit $\lim_{\theta\to+\infty} g(\theta)=0$ is trivial. □

C Sequence $m_n^1(d)$

In this appendix, several properties of the sequence $m_n^1(d)$ defined in Definition 18 (presented in Chapter 5) are shown, where $n\in\mathbb{N}$ and $d=0,\ldots,D=n(n-1)/2$. The first values are shown in Table C.1.

Table C.1: Number of permutations of Σ_n at Kendall tau distance d of permutation σ and at Kendall tau distance $d+1$ of permutation τ, where $d(\sigma,\tau)=1$, for $n=1,\ldots,6$.

n	$m_n^1(0),\ldots,m_n^1(D)$
1	1
2	1, 0
3	1, 1, 1, 0
4	1, 2, 3, 3, 2, 1, 0
5	1, 3, 6, 9, 11, 11, 9, 6, 3, 1, 0
6	1, 4, 10, 19, 30, 41, 49, 52, 49, 41, 30, 19, 10, 4, 1, 0

Proposition 5. *For a fixed $n\in\mathbb{N}$, the sequence $(m_n^1(0),...,m_n^1(D))$ satisfies the following properties:*

(i) For any distance $d\in\{0,\ldots,D\}$,

$$m_n^1(d)=\sum_{i=0}^{d}(-1)^{d-i}m_n(i). \tag{C.13}$$

(ii) For any distance $d\in\{0,\ldots,D\}$,

$$m_n(d)=m_n^1(d)+m_n^1(d-1). \tag{C.14}$$

(iii) For any distance $d\in\{0,\ldots,D\}$,

$$m_n^1(d)=m_n^1(D-d-1). \tag{C.15}$$

(iv) For any distance $d\in\{0,\ldots,D\}$ and $n>3$,

- *If D is even, let us define $d^1_{max} = (D/2) - 1$ and $d^2_{max} = D/2$. Then,*

$$\begin{cases} m^1_n(d) < m^1_n(d+1) & \text{when } d = 0, \ldots, d^1_{max} \\ m^1_n(d) > m^1_n(d+1) & \text{when } d = d^2_{max}, \ldots, D-1. \end{cases} \tag{C.16}$$

- *If D is odd, let us define $d_{max} = \lfloor D/2 \rfloor$. Then,*

$$\begin{cases} m^1_n(d) < m^1_n(d+1) & \text{when } d = 0, \ldots, d_{max} - 1 \\ m^1_n(d) > m^1_n(d+1) & \text{when } d = d_{max}, \ldots, D-1. \end{cases} \tag{C.17}$$

(v) For any distance $d \in \{0, \ldots, D\}$ and $n \neq 1$,

$$m^1_n(d) = \sum_{k=0}^{d} \sum_{j=0}^{n-1} m_{n-1}(k-j) \cdot (-1)^{d-k}. \tag{C.18}$$

(vi) For any distance $d \in \{0, \ldots, D\}$,

$$m^1_n(d) \leq m^1_n(d-1) + m^1_n(d+1). \tag{C.19}$$

Proof. Properties $(i) - (v)$ can be easily derived from Definition 18 and the characteristics of the sequence $m_n(d)$. Finally, let us prove Property (vi), which states that $m^1_n(i) < m^1_n(i-1) + m^1_n(i+1)$. There exist three cases:

a) If $m^1_n(i) \leq m^1_n(i-1)$, the inequality is trivial.

b) If $m^1_n(i) \leq m^1_n(i+1)$, the inequality is trivial.

c) If $m^1_n(i) > m^1_n(i-1)$ and $m^1_n(i) > m^1_n(i+1)$, then $m^1_n(i)$ is a single maximum (Note that this case appears the first time when $n = 6$).

In this particular case, D is an odd number, $m^1_n(i) = m^1_n(\lfloor D/2 \rfloor)$ is the maximum value, $m^1_n(i+1) = m^1_n(i-1)$; and $m_n(\lfloor D/2 \rfloor)$ and $m_n(\lceil D/2 \rceil)$ are the maximum values.

We present the properties and observations used to prove the last situation:

(a) For any $n \geq 6$, the maximum distance between two permutations in Σ_n is $D(n) = n(n-1)/2$. So the difference between the maximum values for two permutations in Σ_n and Σ_{n-1} is $D(n) - D(n-1) = n(n-1)/2 - (n-1)(n-2)/2 = n - 1$.

(b) Using the previous property and the sequence $m_n(i)$, we can deduce the following observations:

- If $m_n(i)$ is the first maximum value for any fixed integer $n \geq 6$, then $m_{n-1}(i - \lfloor (n-1)/2 \rfloor)$ is the (first) maximum value.
- If $m_n(i)$ is the maximum value, then $m_{n-1}(i)$ is located in the descending part of the sequence, that is, $m_{n-1}(i-1) > m_{n-1}(i) > m_{n-1}(i+1)$.
- Similarly, if $m_n(i)$ is the maximum value, then $m_{n-1}(i-n)$ is located in the ascending part of the sequence, that is, $m_{n-1}(i-n-1) < m_{n-1}(i-n) < m_{n-1}(i-n+1)$.

(c) For any integer values n and i, $m_n(i) \leq m_n(i+1) + m_n(i-1)$.

Once we bear the previous observations in mind, let us use Property (v).

$$m_n^1(i) = \sum_{k=0}^{i} m_n(k) \cdot (-1)^{i-k} \overset{(n\neq 1)}{=} \sum_{k=0}^{i} \sum_{j=0}^{n-1} m_{n-1}(k-j) \cdot (-1)^{i-k}. \tag{C.20}$$

- If $n-1 \geq i$:

$$m_n^1(i) \overset{\text{(C.20)}}{=} \begin{cases} \sum_{j=0}^{i/2} m_{n-1}(2j), & \text{if } i \text{ is even} \\ \sum_{j=0}^{(i-1)/2} m_{n-1}(2j+1), & \text{if } i \text{ is odd.} \end{cases} \tag{C.21}$$

- If $n-1 < i$,

$$m_n^1(i) \overset{\text{(C.20)}}{=} \begin{cases} \sum_{j=0}^{i/2} m_{n-1}(2j) - \sum_{k=0}^{(i-n-1)/2} m_{n-1}(2k+1), & \text{if } i \text{ is even and } n \text{ is odd} \\ \sum_{j=0}^{n-1} m_{n-1}(i-j) + \sum_{k=0}^{(i-n-2)/2} m_{n-1}(2k+1) - \sum_{l=0}^{(i-1)/2} m_{n-1}(2l), & \text{if } i \text{ is odd and } n \text{ is odd} \\ \sum_{j=0}^{n-1} m_{n-1}(i-j) + \sum_{k=0}^{(i-n-2)/2} m_{n-1}(2k+1) - \sum_{l=0}^{(i-2)/2} m_{n-1}(2l+1), & \text{if } i \text{ is even and } n \text{ is even} \\ \sum_{j=0}^{(i-1)/2} m_{n-1}(2j+1) - \sum_{k=0}^{(i-n-1)/2} m_{n-1}(2k+1) = \sum_{j=(i-n+1)/2}^{(i-1)/2} m_{n-1}(2j+1), & \text{if } i \text{ is odd and } n \text{ is even.} \end{cases} \tag{C.22}$$

In order to extend the sums, let us denote by " $\overset{(even)}{\cdots}$ " and " $\overset{(odd)}{\cdots}$ " the coefficients with even and odd indexes, respectively.

When $n \geq 6$, there are four possible cases depending on the n and i integer parity values (Equations (C.23) — (C.26)).

Let n be an odd number and i an even number.

$$\begin{aligned} 2m_n^1(i-1) - m_n^1(i) &= 3\left[m_{n-1}(1) + \overset{(odd)}{\cdots} + m_{n-1}(i-n-2)\right] + m_{n-1}(i-n) \\ &\quad + 2\left(m_{n-1}(i-n) + \cdots + m_{n-1}(i-1)\right) - 3\left[m_{n-1}(0) + \overset{(even)}{\cdots} + m_{n-1}(i-2)\right] - m_{n-1}(i) \\ &= 3\overline{m}_{n-1}(i-n-2) + m_{n-1}(i-n) + \left[-m_{n-1}(i-n-1) + m_{n-1}(i-n)\right] + \\ &\quad + \left[m_{n-1}(i-n) - m_{n-1}(i-n+1) + m_{n-1}(i-n+2)\right] + \cdots + \\ &\quad + \left[m_{n-1}(i-3) - m_{n-1}(i-2) + m_{n-1}(i-1)\right] + \left[m_{n-1}(i-1) - m_{n-1}(i)\right] \overset{\text{(b) and (c)}}{>} 0. \end{aligned} \tag{C.23}$$

Let n and i be odd numbers.

$$\begin{aligned}2m_n^1(i-1)-m_n^1(i) &= 3\left[m_{n-1}(0)+\overset{(even)}{\cdots}+m_{n-1}(i-1)\right]\\ &\quad -3\left[m_{n-1}(1)+\overset{(odd)}{\cdots}+m_{n-1}(i-n-1)\right]-[m_{n-1}(i-n+1)+\cdots+m_{n-1}(i)]\\ &= 3\overline{m}_{n-1}(i-n)+[-m_{n-1}(i-n+1)+m_{n-1}(i-n+2)]\\ &\quad +[m_{n-1}(i-n+2)-m_{n-1}(i-n+3)+m_{n-1}(i-n+4)]+\cdots+\\ &\quad +[m_{n-1}(i-3)-m_{n-1}(i-2)+m_{n-1}(i-1)]+[m_{n-1}(i-1)-m_{n-1}(i)]\overset{\text{(b) and (c)}}{>}0.\end{aligned} \tag{C.24}$$

Let n and i be even numbers.

$$\begin{aligned}2m_n^1(i-1)-m_n^1(i) &= 3\left[m_{n-1}(i-n+1)+\overset{(odd)}{\cdots}+m_{n-1}(i-1)\right]-[m_{n-1}(i-n+1)+\cdots+m_{n-1}(i)]\\ &= m_{n-1}(i-n+1)+[m_{n-1}(i-n+1)-m_{n-1}(i-n+2)+m_{n-1}(i-n+3)]+\cdots+\\ &\quad +[m_{n-1}(i-3)-m_{n-1}(i-2)+m_{n-1}(i-1)]+[m_{n-1}(i-1)-m_{n-1}(i)]\overset{\text{(b) and (c)}}{>}0.\end{aligned} \tag{C.25}$$

Let n be an even number and i an odd number.

$$\begin{aligned}2m_n^1(i-1)-m_n^1(i) &= 2\left[m_{n-1}(i-n)+\cdots+m_{n-1}(i-1)\right]+m_{n-1}(i-n)-\\ &\quad -3\left[m_{n-1}(i-n)+\overset{(odd)}{\cdots}+m_{n-1}(i-2)\right]-m_{n-1}(i)\\ &= [m_{n-1}(i-1)-m_{n-1}(i)]+[m_{n-1}(i-3)-m_{n-1}(i-2)+m_{n-1}(i-1)]+\cdots+\\ &\quad +[m_{n-1}(i-n+1)-m_{n-1}(i-n+2)+m_{n-1}(i-n+3)]+m_{n-1}(i-n+1)\overset{\text{(b) and (c)}}{>}0.\end{aligned} \tag{C.26}$$

In all cases, the result is proved. □

D Exponential polynomials

In this appendix, all the properties used in Chapter 5 about the exponential polynomials are shown. Throughout this thesis, the exponential polynomials have integer coefficients and the base used is e. For a fixed value n, the exponential polynomials can be denoted in the following way:

$$Pol(\theta)=\sum_{i=0}^{2D}a_i e^{-i\theta}, \tag{D.27}$$

where D is the maximum Kendall tau distance. The highest value i used in this thesis is $2D$, which is the maximum possible sum of two Kendall tau distance values. By definition, $Pol(0) = \sum_{i=0}^{2D} a_i$, and when θ tends to infinity, $Pol(\theta)$ tends to a_0.

Proposition 6. *The following results are true:*

(i) If $a_i > 0,\ \forall i = 0, \dots, 2D$, then $Pol(\theta)$ is a positive decreasing function.

(ii) If $a_i \geq 0,\ i = 0, \cdots, j$, and $a_i \leq 0,\ i = j+1, \cdots, 2D$, where at least there exists one positive coefficient and one negative, and $\sum_{i=0}^{j} |a_i| < \sum_{i=j+1}^{2D} |a_i|$, then there exists a positive value θ such that $Pol(\theta) = 0$. Analogous with the inverse order.

(iii) Let $a_i \geq 0$, $\forall i = 0, \dots, j_1, j_2, \dots, 2D$ $(j_1 < j_2 - 1)$, and $a_i < 0$, $\forall i = j_1 + 1, \dots, j_2 - 1$, where at least there exists one positive coefficient and one negative. If $\sum_{i=0}^{j_1} |a_i| \geq \sum_{i=j_1+1}^{j_2-1} |a_i|$ and $\sum_{i=j_1+1}^{j_2-1} |a_i| \leq \sum_{i=j_2}^{2D} |a_i|$, then there are no positive roots. Analogous to the opposite order.

(iv) If $a_i = -a_{2D-i}$, $\forall i = 0, \dots, 2D$, then $a_D = 0$ and $Pol(0) = 0$. In addition, there are no θ positive roots (corollary of Property (iii)).

(v) Let $a_i > 0$, $\forall i = 0, \dots, D-1, D+1, \dots, 2D$, $a_i = a_{2D-i}$ and $\sum_{i=0}^{2D} a_i = 0$. Then, $Pol(0) = 0$ and there are no θ positive roots.

Proof. All the properties can be easily proved due to the definition of the exponential function. For Property (v), the argument used in Inequality (B.11) is used. □

D.1 Proving Inequality (5.45)

Proof. To prove Inequality (5.45) at $\hat{\theta}$ value, let us analyze the following functions (for a fixed $n \in \mathbb{N}$ such that $n \geq 3$):

$$
\begin{aligned}
f_1(\theta) &= \sum_{i=0}^{D} \sum_{j=0}^{D} m_n(i) \cdot \left(m_n^1(j) - m_n^1(j-1) \right) e^{-(i+j)\theta}; \\
f_2(\theta) &= \sum_{i=0}^{D} \left((d^*+1) m_n^1(i) + (d^*-1) m_n^1(i-1) \right) e^{-(d^*+i)\theta}.
\end{aligned} \tag{D.28}
$$

$f_1(\theta)$ is an exponential function which fulfills Property (iv) from the exponential polynomials, whereas $f_2(\theta)$ fulfills Property (i). An example of $f_1(\theta) - f_2(\theta)$ is displayed for $n = 5$ and $d = 1, \dots, 4$ in Figure D.3.

In order to prove Inequality (5.45), we have used the following result. At $\theta_0 = \hat{\theta}$:

$$
\begin{aligned}
& d(\sigma^*, \sigma_0) = \sum_{\pi \in \Sigma_n} d(\pi, \sigma_0) p(\pi) = \varphi^{-1}(\hat{\theta}) \sum_{i=0}^{D} m_n(i) \cdot i \cdot e^{-i\hat{\theta}} = -\varphi^{-1}(\hat{\theta}) \cdot \varphi'(\hat{\theta}) \\
\iff & d(\sigma^*, \sigma_0) \cdot \varphi(\hat{\theta}) = -\varphi'(\hat{\theta}) \iff \sum_{i=0}^{D} (d^* - i) \cdot m_n(i) \cdot e^{-i\hat{\theta}} = 0.
\end{aligned} \tag{D.29}
$$

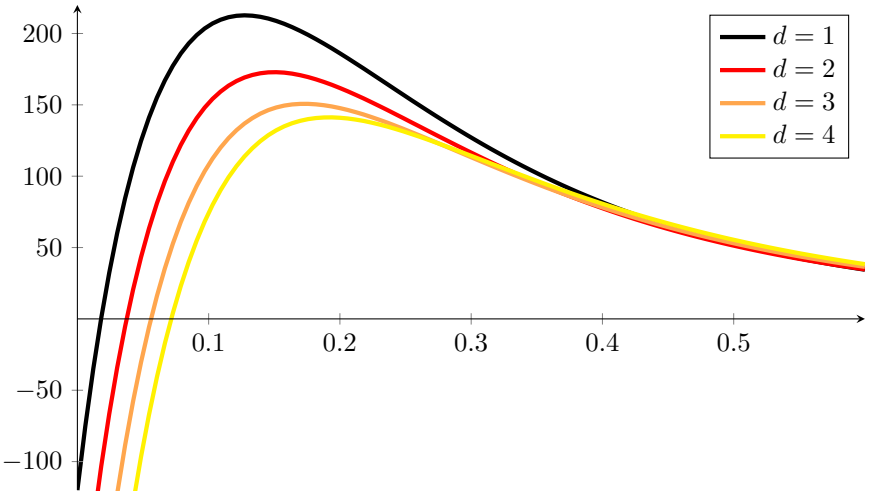


Fig. D.3: The function $f_1(\theta) - f_2(\theta)$, for $n = 5$ and $d = 1, \ldots, 4$.

Hence, let us define a new function:

$$f_3(\theta) = \sum_{i=0}^{D} (d^* - i) \cdot m_n(i) \cdot e^{-i\theta}. \tag{D.30}$$

$f_3(\theta)$ is an exponential polynomial which fulfills Property (ii) due to the fact that $d^* < D/2$, and therefore $\sum_{i=0}^{d^*-1} |a_i| < \sum_{i=d^*+1}^{D} |a_i|$.

After defining the functions f_i for $i = 1, 2, 3$, let us define $F_c(\theta)$ in the following way:

$$F_c(\theta) = f_1(\theta) - f_2(\theta) - c \cdot e^{-D\theta} \cdot f_3(\theta), \tag{D.31}$$

where c is a real positive value. Let us denote $F_c(\theta) = \sum_{i=0}^{2D} b_i^c e^{-i\theta}$.

When $c = 0$, $F_c(\theta)$ is the function associated to Inequality (5.45). At the interval $[0, +\infty)$, $F_0(\theta)$ starts at $-d^* \cdot n!$, and when θ tends to infinity, $F_0(\theta)$ tends to 1. Moreover, by Property (ii), it can be ensured that the equation $F_0(\theta) = 0$ is fulfilled once at $\theta = \theta'$.

We wanted to prove Inequality (5.45): that is to say, $F_0(\hat{\theta}) > 0$. By definition of $F_c(\theta)$, this is equivalent to proving that $F_c(\hat{\theta}) > 0$, for some value c. To prove this, we will choose an appropriate value c which ensures that $F_c(\theta)$ is a positive value for any $\theta \in [0, +\infty)$. First, notice that when c tends to infinity, $F_c(0)$ tends to $+\infty$. Then, we can ensure for $c > M$, being M the smallest positive number such that $\sum_{i=0}^{2D} b_i^M = 0$ is fulfilled, that $F_c(0) > 0$ and when θ tends to infinity, $F_c(\theta)$ tends to 1 because $b_0^c = 1$. Finally, an option to prove that F_c is a positive function for a particular c is to observe that, for a suitable value c, $F_c(\theta)$ fulfills Property (iii). This can be ensured because of the inequality $\sum_{i=0}^{d^*-1} a_i < -\sum_{i=d^*+1}^{D} a_i$ which $f_3(\theta)$ fulfills.

Consequently, $F_c(\theta) > 0$ for any $\theta \in [0, +\infty)$. Particularly, $F_c(\hat{\theta}) > 0$ which implies, by definition of $F_c(\theta)$, that Inequality (5.45) is fulfilled. □

D.2 The function h is a negative function.

Proposition 7. *For any $\theta \geq 0$, let us denote*

$$h(\theta) = \sum_{d=1}^{D} m_n(d) \cdot d \cdot p(\tilde{\sigma}_d) \left(1 - 2 \sum_{i=0}^{d-1} m_n(i) p(\tilde{\sigma}_i) - m_n(d) p(\tilde{\sigma}_d) \right) \tag{D.32}$$

the difference value between $\sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) p^S(\pi)$ and $\sum_{\pi \in \Sigma_n} d(\pi, \sigma^*) p(\pi)$. Then, $h(\theta)$ is a negative value.

Proof.

$$\begin{aligned} h(\theta) < 0 &\iff \sum_{d=1}^{D} m_n(d) \cdot d \cdot \frac{e^{-d\theta}}{\varphi(\theta)} \cdot \left(1 - 2 \sum_{i=0}^{d-1} m_n(i) \frac{e^{-i\theta}}{\varphi(\theta)} - m_n(d) \frac{e^{-d\theta}}{\varphi(\theta)} \right) < 0 \\ &\iff \sum_{d=1}^{D} m_n(d) \cdot d \cdot e^{-d\theta} \cdot \varphi(\theta) < \sum_{d=1}^{D} m_n(d) \cdot d \cdot e^{-d\theta} \cdot \left(2 \sum_{i=0}^{d-1} m_n(i) e^{-i\theta} + m_n(d) e^{-d\theta} \right). \end{aligned} \tag{D.33}$$

The proof is based on developing the sum in two non-positive exponential polynomials with non-negative coefficients and comparing those coefficients one-by-one. On the one hand, let us denote by a_i the coefficient of $e^{-i\theta}$ in the left-hand side of Inequality (D.33).

$$\sum_{i=1}^{2D} a_i e^{-i\theta} = \varphi(\theta) \left(\sum_{d=1}^{D} d \cdot m_n(d) \cdot e^{-d\theta} \right), \tag{D.34}$$

where

$$a_i := \sum_{j=1}^{D} \sum_{k=0}^{D} j \cdot m_n(j) \cdot m_n(k) \cdot \delta_{i,j+k} \tag{D.35}$$

and $\delta_{i,j+k}$ is the Kronecker delta:

$$\delta_{i,j+k} = \begin{cases} 1, & \text{if } j+k = i \\ 0, & \text{otherwise.} \end{cases} \tag{D.36}$$

On the other hand, let us denote by b_i the coefficient of $e^{-i\theta}$ in the right-hand side of Inequality (D.33).

$$\sum_{i=1}^{2D} b_i e^{-i\theta} = \sum_{d=1}^{D} m_n(d) \cdot d \cdot e^{-d\theta} \cdot \left(2 \sum_{i=0}^{d-1} m_n(i) e^{-i\theta} + m_n(d) e^{-d\theta} \right), \tag{D.37}$$

where

$$b_i := \sum_{j=0}^{D} \sum_{k=0}^{D} \beta_{j,k} \cdot m_n(j) \cdot m_n(k) \cdot \delta_{i,j+k} \tag{D.38}$$

and

$$\beta_{j,k} = \begin{cases} 2j, & \text{if } j \geq k \\ 2k, & \text{if } j < k. \end{cases} \tag{D.39}$$

To prove Inequality (D.33), let us demonstrate that $a_i \leq b_i, \forall i = 1, \ldots, 2D$, and $a_i < b_i$ for at least one index i. For any i, note that if $i \neq j + k$, there is no coefficient. Otherwise, when $j > k$, then $2j > i$; when $j = k$, the coefficient of the summation is the same, and when $k > j$, then $2k > i$. Therefore, it is demonstrated that

$$h(\theta) = \frac{1}{\varphi^2(\theta)} \sum_{i=1}^{2D} (a_i - b_i) e^{-i\theta} < 0. \tag{D.40}$$

□