

UNIVERSIDADE FEDERAL FLUMINENSE

ÍTALO GOMES SANTANA

**IMPROVING A STATE-OF-THE-ART HEURISTIC
FOR THE MINIMUM LATENCY PROBLEM
WITH DATA MINING**

Thesis presented to the Computing Graduate Program of the Universidade Federal Fluminense in partial fulfillment of the requirements for the degree of Master of Science.
Topic Area: Algorithms and Optimization.

Advisor:

ALEXANDRE PLASTINO DE CARVALHO

Co-advisor:

ISABEL CRISTINA MELLO ROSSETI

NITERÓI

2018

Ficha catalográfica automática - SDC/BEE

S231i Santana, Italo Gomes
Improving a State-of-the-Art Heuristic for the Minimum Latency Problem with Data Mining / Italo Gomes Santana ; Alexandre Plastino, orientador ; Isabel Cristina Mello Rosseti, coorientadora. Niterói, 2018.
90 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Niterói, 2018.

1. Heurística. 2. Mineração de dados (Computação). 3. Otimização combinatória. 4. Problema da Mínima Latência. 5. Produção intelectual. I. Título II. Plastino, Alexandre, orientador. III. Rosseti, Isabel Cristina Mello, coorientadora. IV. Universidade Federal Fluminense. Escola de Engenharia.

CDD -


ÍTALO GOMES SANTANA

IMPROVING A STATE-OF-THE-ART HEURISTIC FOR THE MINIMUM
LATENCY PROBLEM WITH DATA MINING

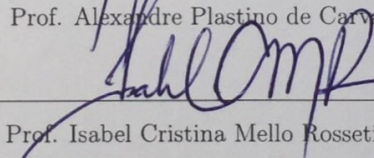
Thesis presented to the Computing Graduate Program of the Universidade Federal Fluminense in partial fulfillment of the requirements for the degree of Master of Science.
Topic Area: Algorithms and Optimization.

Approved on April 13th, 2018.

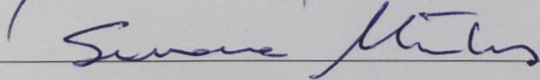
APPROVED BY



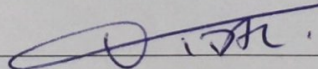
Prof. Alexandre Plastino de Carvalho, Advisor, UFF



Prof. Isabel Cristina Mello Rosseti, Co-advisor, UFF



Prof. Simone de Lima Martins, UFF



Prof. Thibaut Victor Gaston Vidal, PUC-RIO

Niterói

2018

To my parents.

Acknowledgements

I want to thank my parents for their support during all this time.

I want to thank my advisors, Professor Plastino and Rosseti, for their outstanding collaboration in all aspects of this work.

Special thanks to Marcos Melo, and Professors Thibaut, Satoru, and Anand for granting me the original source-code and the essential resources to re-execute the original experiments that was no doubt the basis of this work.

A final thanks to my housemates, friends (not only from UFF), and professors of the Computing Institute that made part of my time in Niterói.

Finally, I would like to thank CAPES and CNPq for their financial support.

Resumo

Recentemente, metaheurísticas híbridas têm se tornado uma tendência em pesquisa operacional. Um exemplo bem sucedido combina *Greedy Randomized Adaptive Search Procedures* (GRASP) e técnicas de mineração de dados, onde padrões frequentes encontrados em soluções de alta qualidade podem levar a uma exploração eficiente do espaço de busca, juntamente com uma redução significativa de tempo computacional. Neste trabalho, uma heurística estado-da-arte baseada em GRASP para o Problema da Mínima Latência (PML) é aperfeiçoada por meio de técnicas de mineração de dados em duas variantes do PML. Experimentos computacionais mostraram que as abordagens com mineração de dados igualaram ou melhoraram a qualidade de solução para um número expressivo de instâncias, juntamente com uma redução substancial de tempo de execução. Além disso, 88 novos valores de custos de soluções de ambos problemas são introduzidos na literatura. Para avaliar os resultados reportados, testes de significância estatística, impacto de uso de padrões minerados, comparações de mesmo tempo e gráficos *time-to-target* são apresentados.

Palavras-chave: Problema da Mínima Latência, Metaheurísticas Híbridas, GRASP, Mineração de Dados

Abstract

Recently, hybrid metaheuristics have become a trend in operations research. A successful example combines the Greedy Randomized Adaptive Search Procedures (GRASP) and data mining techniques, where frequent patterns found in high-quality solutions can lead to an efficient exploration of the search space, along with a significant reduction of computational time. In this work, a GRASP-based state-of-the-art heuristic for the Minimum Latency Problem (MLP) is improved by means of data mining techniques for two MLP variants. Computational experiments showed that the approaches with data mining were able to match or improve the solution quality for a large number of instances, together with a substantial reduction of running time. In addition, 88 new cost values of solutions are introduced into the literature. To support our results, tests of statistical significance, impact of using mined patterns, equal time comparisons and time-to-target plots are provided.

Keywords: Minimum Latency Problem, Hybrid Metaheuristics, GRASP, Data Mining

List of Figures

2.1	An MLP feasible solution. Adapted from [37]	5
2.2	An MLP feasible solution considering Hamiltonian path	5
3.1	A double-bridge move applied to a solution	12
4.1	Use of mined frequent arcs to generate initial solutions	16
5.1	Cost values per iteration - TRP-S1000-R1 instance	31
5.2	Cost values per iteration - TRP-S1000-R7 instance	31
5.3	TTT plots for kroA200 instance	32
5.4	TTT plots for pr299 instance	33
7.1	Cost values versus Iteration - TRP-S1000-R1 instance	56
7.2	Cost values versus Iteration - TRP-S1000-R7 instance	56
7.3	TTT plots for kroA200 instance	57
7.4	TTT plots for pr299 instance	57

List of Tables

4.1	Dataset D composed of five transactions	14
5.1	Results for instances selected from TSPLib in [1, 2]	24
5.2	Results on the 56-instances set selected from TSPLib	25
5.3	Results for TSPLib instances selected in [39]	26
5.4	Results for instances generated in [39] considering 10, 20 and 50 customers	26
5.5	Results on the 100-customers instances generated in [39]	27
5.6	Results on the 200-customers instances generated in [39]	28
5.7	Results on the 500-customers instances generated in [39]	29
5.8	Results on the 1000-customers instances generated in [39]	30
5.9	Results for TSPLib instances selected in [1, 2]	34
5.10	Results on the 56-instances set selected from TSPLib	35
5.11	Results for TSPLib instances selected in [39]	35
5.12	Results on the 100-customers instances generated in [39]	36
5.13	Results on the 200-customers instances generated in [39]	36
5.14	Results on the 500-customers instances generated in [39]	37
5.15	Results on the 1000-customers instances generated in [39]	37
7.1	Results on the instances selected from TSPLib in [1, 2]	43
7.2	Computational time for TSPLib instances selected in [1, 2]	44
7.3	Summary for Tables 7.1 and 7.2	45
7.4	Results on the 56-instances set selected from TSPLib	45
7.5	Computational time for the 56-instances set selected from TSPLib	46
7.6	Summary for Tables 7.4 and 7.5	46

7.7	Results for TSPLib instances selected in [39]	47
7.8	Computational time for TSPLib instances selected in [39]	47
7.9	Summary for Tables 7.7 and 7.8	47
7.10	Results for instances generated in [39] considering 10, 20 and 50 customers	48
7.11	Results on the 100-customers set generated in [39]	48
7.12	Computational time for the 100-customers set generated in [39]	49
7.13	Summary for Tables 7.11 and 7.12	49
7.14	Results for the 200-customers set generated in [39]	50
7.15	Computational time for the 200-customers set generated in [39]	51
7.16	Summary for Tables 7.14 and 7.15	51
7.17	Results for the 500-customers set generated in [39]	52
7.18	Computational time for the 500-customers set generated in [39]	53
7.19	Summary for Tables 7.17 and 7.18	53
7.20	Results on the 1000-customers set generated in [39]	54
7.21	Computational time for the 1000-customers set generated in [39]	54
7.22	Summary for Tables 7.20 and 7.21	55
7.23	Results for instances selected from TSPLib in [1, 2]	58
7.24	Results on the 56-instances set selected from TSPLib	59
7.25	Results for TSPLib instances selected in [39]	60
7.26	Results on the 100-customers set generated in [39]	60
7.27	Results on the 200-customers set generated in [39]	61
7.28	Results on the 500-customers set generated in [39]	61
7.29	Results on the 1000-customers set generated in [39]	62
A.1	Results for instances selected from TSPLib in [1, 2]	69
A.2	Results on the 56-instances set selected from TSPLib	70
A.3	Results for TSPLib instances selected in [39]	71

A.4	Results on the 100-customers set generated in [39]	71
A.5	Results on the 200-customers set generated in [39]	72
A.6	Results on the 500-customers set generated in [39]	72
A.7	Results on the 1000-customers set generated in [39]	73
A.8	New cost values for the MLP version of Hamiltonian circuits	74
A.9	New cost values for the MLP version of Hamiltonian paths	75

List of Abbreviations and Acronyms

BKS	:	Best Known Solution;
DM	:	Data Mining;
FIM	:	Frequent Itemset Mining;
GRASP	:	Greedy Randomized Adaptive Search Procedures;
ILS	:	Iterated Local Search;
MLP	:	Minimum Latency Problem;
RVND	:	Variable Neighborhood Descent with Random ordering;
TSP	:	Traveling Salesman Problem;
TRP	:	Traveling Repairman Problem.

Contents

1	Introduction	1
2	The Minimum Latency Problem	4
3	GILS-RVND: the State-of-the-Art Algorithm	7
4	DM-GILS-RVND: the First Proposal	13
4.1	Data Mining Concepts	13
4.2	Hybridization of Heuristics with Data Mining	14
4.3	Hybridization of GILS-RVND with Data Mining	15
5	Computational Experiments for DM-GILS-RVND	21
5.1	Comparison Methodology	21
5.2	Experiments for Hamiltonian Circuit	23
5.3	Experiments for Hamiltonian Path	24
5.4	Complementary Analyses	29
5.4.1	Impact of the Usage of Mined Patterns	29
5.4.2	Analyses of Time Convergence	31
5.4.3	Complementary Experiments	32
5.4.3.1	Experiments for Hamiltonian Circuit	34
5.4.3.2	Experiments for Hamiltonian Path	35

6	MDM-GILS-RVND: the Second Proposal	38
7	Computational Evaluation for MDM-GILS-RVND	41
7.1	Comparison Methodology	41
7.2	Experiments for Hamiltonian Circuit	42
7.3	Experiments for Hamiltonian Path	43
7.4	Complementary Analyses	52
7.4.1	Impact of the Usage of Mined Patterns	55
7.4.2	Analyses of Time Convergence	56
7.4.3	Complementary Experiments	57
7.4.3.1	Experiments for Hamiltonian Circuit	58
7.4.3.2	Experiments for Hamiltonian Path	59
8	Conclusion and Future Works	63
	References	65
	Appendix A - COMPLEMENTARY EXPERIMENTS	69
A.1	Experiments for Hamiltonian Circuit	69
A.2	Experiments for Hamiltonian Path	71
A.3	New Best Known Solutions for the Minimum Latency Problem	73

Chapter 1

Introduction

In operations research, metaheuristics are widely used to solve combinatorial optimization problems, providing near-optimal solutions in practical execution time. Proposed in [15], Greedy Randomized Adaptive Search Procedures (GRASP) is a well-known metaheuristic successfully applied to several combinatorial optimization problems. Briefly, GRASP is a simple iterative process composed of a constructive phase and a local search phase. Its mechanism works, at each iteration, constructing a feasible solution by a semi-greedy procedure to, then, be improved by exploring its neighbor search space.

Combining components from different metaheuristics, even from distinct aspects and paradigms, has become a trend in combinatorial optimization, hence, named as hybrid metaheuristics [43]. To extend their efficiency and flexibility, hybrid metaheuristics are also able to incorporate concepts from other fields of study, such as Data Mining (DM). For instance, a hybrid GRASP with data mining techniques, named Data Mining GRASP, or simply DM-GRASP, was presented in [35, 36]. This approach succeeded to achieve promising results not only in terms of solution quality but also in execution time.

DM refers to the automatic extraction of knowledge from datasets, which can be represented by means of rules or patterns [19]. In heuristics, the basic idea of incorporating data mining consists in the extraction of patterns from high-quality solutions to efficiently guide the local search phase. This hybrid approach first appeared tackling the Set Packing Problem [35, 36]. The adopted strategy in those works divided the original GRASP into two parts. The first one is responsible for obtaining a set of high-quality solutions (elite set). Next, a data mining process is applied on this elite set in order to extract a set of patterns. The second part uses, at each iteration, an extracted pattern from the set of patterns to generate an initial solution by means of an adapted constructive method. By its promising results, this approach was also applied to the Maximum Diversity Problem [41],

the Server Replication for Reliable Multicast Problem [40], the p-Median Problem [33], the 2-Path Network Design Problem [8] and the One-Commodity Pickup-and-Delivery Traveling Salesman Problem [18].

Afterward, an adaptive version of DM-GRASP, named Multi DM-GRASP (MDM-GRASP), was proposed in [32] for the Server Replication for Reliable Multicast Problem. Different from the DM-GRASP strategy, where the data mining process is performed once, the approach presented in MDM-GRASP performs the data mining process whenever the elite set becomes stable. This stability refers to a given number of iterations without any changes in the elite set. In practical terms, the general quality of the elite set is progressively enhanced throughout the execution, yielding, at each time, better patterns. Indeed, the results from the MDM version were superior when compared to those from the original heuristics and reasonably better than those from the DM version [8, 18, 32], both in terms of solution quality and computational time.

In this work, we propose the hybridization with data mining of a state-of-the-art heuristic for the Minimum Latency Problem (MLP), called GILS-RVND. This robust heuristic was developed in [42] joining GRASP [15], Iterated Local Search (ILS) [23] and Variable Neighborhood Descent with Random neighborhood ordering (RVND) [29]. Another contribution present in GILS-RVND refers to an evaluation framework for moves of neighborhood structures that requires $O(1)$ amortized operations, which was originally proposed for several variants of the Vehicle Routing Problems [22, 44]. Results reported in [42] showed that for instances up to 50 customers, optimality was found in all cases, and for instances up to 1000 customers, this heuristic was extremely efficient, thus, outperforming all existing heuristics approaches at that time regarding solution quality and computational time.

Due to the successful results presented by data mining approaches in the literature, the contribution of this work regards the improvement of GILS-RVND by incorporating data mining techniques, adopting both DM-GRASP and MDM-GRASP approaches, named DM-GILS-RVND and MDM-GILS-RVND, respectively, for the two MLP versions tackled in [42]. These two MLP versions consider either a Hamiltonian path or a Hamiltonian circuit as solution, and were divided, according to [42], into 150 instances for the Hamiltonian path version and 23 instances for the Hamiltonian circuit version.

To extend our experiments for the Hamiltonian circuit version, a 56-instance set is introduced into this work, including all instances from 120 to 1379 customers found in

TSPLib¹. This new set aims at better evaluating all heuristics, once for the 23-instance set, whose instances vary from 42 to 107 customers, as there is no clear difference among the results.

It is noteworthy that the original source codes, as well as the values of the initial seeds of all experiments from [42], were provided by the authors, which allowed us to entirely reproduce the original experiments. Submitted to the same prior conditions, the proposed hybrid heuristics were evaluated and their results were compared to the original version, using the same set of parameters defined in [42] and following a fair and strict evaluation. To support all computational experiments reported in this work, tests of statistical significance, analysis of mined patterns, equal time comparisons and time-to-target plots were also provided.

From these analyses, the hybrid data mining approaches applied to GILS-RVND demonstrated their efficiency not only in the improvement of solution quality, but also in reduction of computational time. An aftermath of these results provided 88 new solution cost values, or Best Known Solutions (BKS), for the MLP literature.

The remaining of this work is organized as follows. In Chapter 2, the Minimum Latency Problem is introduced along with its related literature. Chapter 3 presents the GILS-RVND heuristic. In Chapter 4, the first proposed heuristic with data mining, the DM-GILS-RVND, is showed and its computational evaluation is displayed in Chapter 5. The second hybrid heuristic, the MDM-GILS-RVND, and its computational evaluation are presented in Chapters 6 and 7, respectively. Concluding remarks and future works are pointed out in Chapter 8.

¹Instances collected from <http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/index.html>

Chapter 2

The Minimum Latency Problem

Firstly studied as the Traveling Repairman Problem (TRP) in [3] and afterward treated as the Minimum Latency Problem (MLP) [10], this problem is a Traveling Salesman Problem (TSP) variant and can be defined as follows. Let $G = (V, A)$ be a complete directed graph, where $V = \{v_0, \dots, v_n\}$ stands for the vertices set, being v_0 the depot and the remainder ones representing the customers, and $A = \{(i, j) : i, j \in V, i \neq j\}$ is the set of arcs, where each one is associated with a travel time between vertices i and j . The goal of the MLP is to find a Hamiltonian circuit in G that minimizes the total waiting time (latency) of the customers. The latency of the i -th customer, or $l(i)$, is the sum of all travel times from the depot to the i -th customer present in the Hamiltonian circuit. Therefore, the latency of all customers from the Hamiltonian circuit aimed to be minimized in MLP is defined as $\sum_{k=0}^n l(k)$.

Let S be an MLP feasible solution presented in Figure 2.1(a) and its sequence of visited customers in Figure 2.1(b). Thus, the total latency (cost value) of S , or $f(S)$, is the sum of all 11 latencies of each customers of S , which, in short, has cumulative costs, while in a TSP solution, the costs are a simple sum of all arcs of the solution.

Regarded in [39], another MLP approach does not account the return to the depot as part of the solution, thus, it searches for a Hamiltonian path instead. An illustrative example, following the same format displayed in Figure 2.1, is shown in Figure 2.2 for this MLP version, where the same scenario was used to clearly indicate the differences between the two MLP versions.

In a general context, MLP is recognized much harder than TSP, since minor changes in the sequence of visited customers of an MLP solution can propagate major non-local changes in its general structure [10]. MLP is proven as \mathcal{NP} -Hard for general metric

spaces or for Euclidian spaces in [38]. In the literature, the MLP is also known as Traveling Repairman Problem [3], Delivery Man Problem [16], Cumulative Traveling Salesman Problem [9] and School Bus Driver Problem [12].

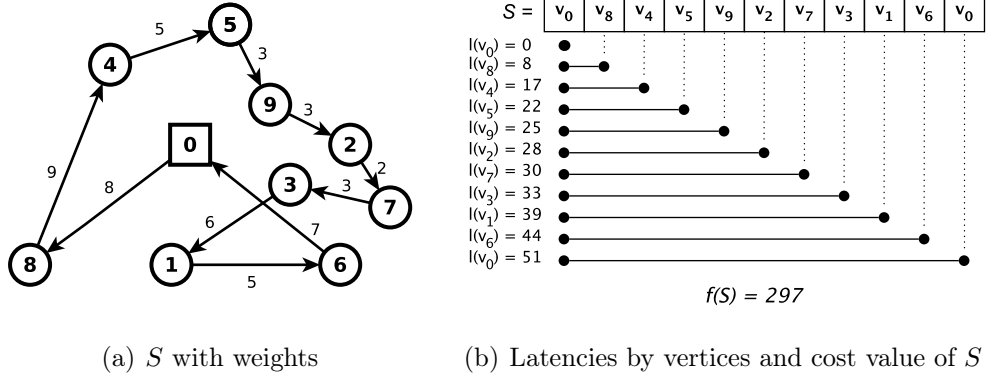


Figure 2.1: An MLP feasible solution. Adapted from [37]

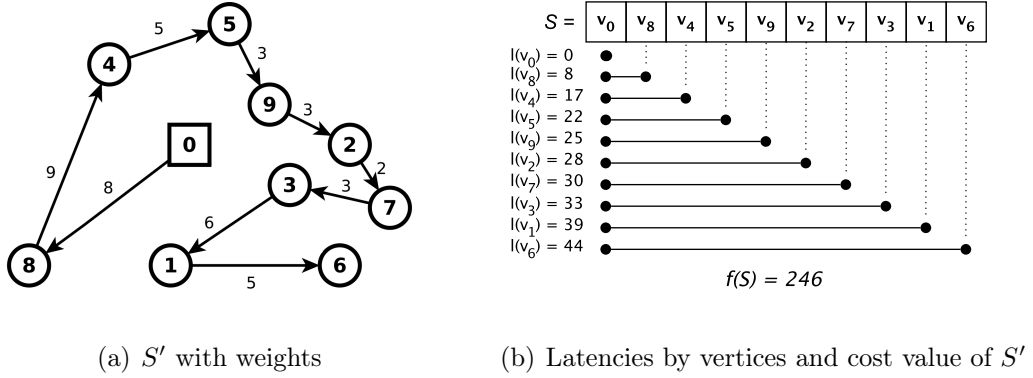


Figure 2.2: An MLP feasible solution considering Hamiltonian path

Many real-life applications are associated to the MLP because of its customer viewpoint aspect, different from the server viewpoint aspect present in the TSP. Several examples of MLP instances can be found in daily life, such as delivery of goods [16], disaster situations [11], data retrieval in networks [14] and disk-head scheduling [10].

A lot of research in the MLP literature has been developed and notable approaches for solving this problem can be divided into exact algorithms, approximation algorithms, and heuristics approaches.

Many exact algorithms have been proposed in the literature for the MLP, and the most important ones are cited as follows. In [24], a branch-and-bound algorithm based on a lower bound scheme was developed and tested on instances between 15 and 30 nodes. A matroidal structure to obtain lower bounds for an enumerative algorithm was used in [16], involving instances up to 60 vertices. Two exact algorithms that incorporate

lower bounds provided by Lagrangian relaxations were reported in [9]. Both researches in [13] and in [28] presented new mixed integer programming formulations, where lower bounds were obtained by linear programming relaxations. Later on, two new linear integer formulations, based on previous formulations found in the literature, were proposed in [14], which could deal with instances with no more than 29 vertices. A branch-and-cut-and-price algorithm, introduced in [1, 2], succeed to handle instances up to 107 vertices, which remains nowadays as the largest instance-size treated in the literature. Based on branch-and-bound, there are also two formulations for the MLP with time windows approach [21]. Finally, two new integers formulations, considering asymmetrical instances, were developed in [6].

Non-exact approaches, such as approximation algorithms, also appear in the literature for the MLP. Reported in [10], a polynomial time algorithm reached an approximation factor (ratio) of 144. Later on, known for their relevancy, approximation algorithms with ratio of 3.59 and 3.03 were reported in [12] and [7], respectively.

Finally, several important heuristics have been proposed to solve the MLP. Apart from an enumerative algorithm, a heuristic approach was also designed in [16]. Heuristics for MLP, VRP and TSP were introduced in [11], where the reported experiments used different objective functions. In [31], a memetic algorithm to obtain upper bounds was developed. Beyond two exact formulations presented in [21] for the MLP with time windows, two heuristic algorithms with and without a tabu list component were developed. Two GRASP-based heuristics with Variable Neighborhood Search (VNS) and another one with Variable Neighborhood Descent (VND) were elaborated and presented in [39].

Developed and published in [42], the GILS-RVND, a simple and effective algorithm for the MLP based on GRASP, ILS and RVND, holds nowadays the best results for a set of 173 instances, including instances up to 1000 customers, being 23 instances tested in [1, 2] and 150 instances from [39]. One year later, these instances were also adopted in [30] to test two General Variable Neighborhood Search (GVNS) heuristics proposed on the basis of different strategies for the local search phase. The results reported in [30] were only compared to results reported in works that introduced these instances [1, 2, 39], where some improvement on these instances was observed. Although the GVNS heuristics and GILS-RVND were not directly compared, analysing their reported results, GILS-RVND presented a general better performance in terms of solution quality. Thus, based on these evidences, GILS-RVND can be considered as a state-of-the-art heuristic for the MLP. Recently, parallel versions based on GILS-RVND were also proposed in [37].

Chapter 3

GILS-RVND: the State-of-the-Art Algorithm

GILS-RVND is a heuristic that joins components of Greedy Randomized Adaptive Search Procedures (GRASP) [15], Iterated Local Search (ILS) [23], and Variable Neighborhood Descent with Random neighborhood ordering (RVND) [29]. Beyond these components, a framework capable of evaluating moves of neighborhood structures in $O(1)$, developed in [22] and extended in [44], was, for the first time, applied into a heuristic for the MLP.

In [15], GRASP is presented as a simple iterative process composed of two phases in each iteration, a constructive phase, and a local search phase. This iterative process ends when the stopping criterion is met, and then, the best solution found is returned. Responsible for generating feasible initial solutions, the constructive phase of GRASP uses a random component to control its greediness, allowing distinct starts for the local search phase at each iteration. In the constructive phase, a partial solution (initially empty) is built by inserting an element at a time until it becomes complete. The initial solution is probably not near-optimal, so, for its improvement, it is submitted to the second phase of GRASP, the local search phase. This phase systematically explores the search space of the problem, in order to replace the current solution to a better one considering the neighborhood. When the current solution cannot be further improved by neighbor solutions, the local search phase terminates. The GRASP terminates when its stopping criterion is reached, such as a predefined number of iterations.

Iterated Local Search (ILS) is a simple and flexible metaheuristic for optimization problems [23]. Basically, this metaheuristic builds its initial solution once, and submit it to an iterative process composed of a local search phase, a perturbation mechanism and an acceptance criterion for solutions. Considering a basic ILS, the initial solution s

can be entirely random or constructed by a semi-greedy procedure. Next, the loop of the ILS starts with the local search component working to improve s by browsing in its neighborhood. After the local search, s is submitted to a perturbation mechanism, allowing to avoid a drawback of being trapped into local optima that may happen to occur during the algorithm execution. This mechanism modifies the solution in such way to escape from a local optima, hence, leading the local search phase to explore another search space area. Finally, ILS uses an acceptance criterion to replace the best global solution by s , where s is chosen not only by the usual cost value factor, but also from factors involving random acceptance and diversification, as suggested in [23]. The iterative process of ILS terminates when the stopping criterion of the loop is reached, returning the best global solution found so far.

Branching from the Variable Neighborhood Descent (VND) [29], the VND with Random neighborhood ordering, or RVND, follows the same principles of VND, except for the deterministic ordering present in VND. The VND mechanism works applying a set of neighborhood structures onto a solution, considering a previously defined order. On the other hand, the ordering of these structures is randomized in RVND.

Furthermore, a notable contribution included into GILS-RVND refers to a simple evaluation framework that requires $O(1)$ amortized operations to evaluate neighbor solutions. This approach was firstly presented in [22] and extended in [44] for many Vehicle Routing Problem variants.

This framework allows calculating cost values of neighbor solutions using preprocessed data structures, which turns to be much faster than the traditional evaluation of considering all customers one by one. At first, a solution s is decomposed into many subsequences, and each subsequence cost value is stored in specific data structures. Next, when the neighborhood of s is explored for better solutions, each neighbor solution can be reached using these data structures, where a specific order of subsequences will generate a neighbor solution. Each specific order is associated with a neighborhood structure. Since five classic TSP neighborhood structures were used in [42], five different orders of subsequences were defined. Therefore, the use of preprocessed cost values of subsequences to calculate cost values of neighbor solutions made GILS-RVND a very efficient algorithm.

Therefore, the combination of components from GRASP, ILS, RVND, and the above framework culminated in the development of GILS-RVND, for both MLP versions already mentioned in Chapter 2. Reported in [42], this heuristic found optimal solutions for instances up to 50 customers and was effective to find good solutions for instances up

to 1000 customers. All previous heuristic approaches for the MLP were outperformed by GILS-RVND at that time. This heuristic also opened new challenges for the MLP literature, since its computational time and solution quality improved other heuristic approaches for the MLP. Thus, GILS-RVND is regarded as the state-of-art proposal for both MLP variants.

The pseudocode of GILS-RVND is detailed in Algorithm 1. Firstly, the cost of the current best solution s^* is initialized (line 1). In the outer loop, for each one of the I_{Max} iterations, an initial solution s is generated by a constructive procedure requiring an $\alpha \in R$ to control the greediness level of this procedure (line 3). Given as input data, the R set contains I_{Max} random values from the interval $[0, 1)$. Before the inner loop, s is copied to s' and $iter_{ILS}$ counter is set to zero (lines 4-5). From this point, s' refers to the best solution of the local search phase (lines 6-14). The basic idea of this phase prevails on the exploration of the neighborhood of s using a RVND procedure (line 7) and a perturbation mechanism over s' (line 12) until I_{ILS} consecutive attempts without any improvement of s' . This perturbation avoids a solution drawback of being trapped into local optima, which will be later detailed. If the cost value of s , or $f(s)$, is lower than $f(s')$ (as MLP is a minimization problem), then, s' is updated and $iter_{ILS}$ is reset. After this checking, s' is perturbed and $iter_{ILS}$ is increased by one unit (lines 12-13). Before the outer loop ending, s^* may be updated according to s' by means of their cost values (lines 15-17). Once all I_{Max} iterations are performed, s^* is returned (line 19).

In order to show how initial solutions are generated, the constructive procedure of GILS-RVND is explained in Algorithm 2. At first, the depot (vertex 0) is straightaway inserted into the partial solution s whilst the remaining ones fill the Customer List (CL) (lines 1-2). For simplification purposes, a variable r is declared to always hold the last inserted customer into s throughout this algorithm execution, where it receives the depot as initial value (line 3). Next, a systematic process to fill s is executed in a loop (lines 4-11). At each iteration, the customers of CL are firstly sorted in ascending order by the travel time between each one and r (line 5). After this sorting, the best $(\alpha \times 100)\%$ customers from the CL are copied to the Restricted Customer List (RCL), where the less is α , the shorter is RCL, and $\alpha = 0$ stands for the best customer of RCL (line 6). Since RCL gets filled, a random customer from this list, or c , is selected to, then, be appended at the end of s (lines 7-8). As c is now the last customer of s , it updates the r variable to the sorting of CL at the next loop iteration and, finally, gets removed from CL (lines 9-10). Eventually, all elements from CL are transferred to s , turning the partial solution into a feasible complete solution returned by the algorithm (line 12).

Algorithm 1 GILS-RVND(I_{Max}, I_{ILS}, R)

```

1:  $f(s^*) \leftarrow \infty$ ;
2: for  $i = 1, \dots, I_{Max}$  do
3:    $s \leftarrow \text{ConstructiveProcedure}(\alpha \in R)$ ;
4:    $s' \leftarrow s$ ;
5:    $iterILS \leftarrow 0$ ;
6:   while  $iterILS < I_{ILS}$  do
7:      $s \leftarrow RVND(s)$ ;
8:     if  $f(s) < f(s')$  then
9:        $s' \leftarrow s$ ;
10:       $iterILS \leftarrow 0$ ;
11:     end if
12:      $s \leftarrow \text{Perturb}(s')$ ;
13:      $iterILS \leftarrow iterILS + 1$ ;
14:   end while
15:   if  $f(s') < f(s^*)$  then
16:      $s^* \leftarrow s'$ ;
17:   end if
18: end for
19: return  $s^*$ 

```

Next, the RVND procedure and the perturbation mechanism used in the local search of GILS-RVND are described. In a general view of RVND, given a set of neighborhood structures (NS), the RVND randomly selects a neighborhood structure $nh \in NS$ to try the improvement of the best current solution s . If s is improved by nh , s is updated and NS is restored to its original state. Otherwise, nh is removed from NS . In other words, at the next iteration, NS will be either equal to the NS given as input or the current NS without nh . Once all neighborhood structures, in sequence, fail to improve s , this procedure returns s . Five TSP traditional neighborhood structures were used in this local search phase and they are described below:

- Swap: two customers are interchanged.
- 2-opt: two non-adjacent arcs are removed and two others are inserted.
- Reinsertion: one customer is relocated to another position.
- Or-opt-2: one arc is relocated to another position.
- Or-opt-3: two adjacent arcs are relocated to another position.

In algorithmic terms, the RVND described before is detailed in Algorithm 3. At first, a neighborhood structures list (NS) with all neighborhood structures (line 1) and the data structures on subsequences of s are initialized (line 2). These data structures

Algorithm 2 ConstructiveProcedure(α)

```

1:  $s \leftarrow \{0\}$ ;
2:  $CL \leftarrow \text{GenerateCL}()$ ;
3:  $r \leftarrow 0$ ;
4: while  $CL \neq \emptyset$  do
5:    $CL \leftarrow \text{SortCL}(CL, r)$ ;
6:    $RCL \leftarrow \text{FillRCL}(CL, \alpha)$ ;
7:    $c \leftarrow \text{SelectRandomClient}(RCL)$ ;
8:    $s \leftarrow s \cup \{c\}$ ;
9:    $r \leftarrow c$ ;
10:   $CL \leftarrow CL - \{c\}$ 
11: end while
12: return  $s$ 

```

are components of the evaluation framework, which divides s into many subsequences to evaluate its neighborhood in $O(1)$ amortized operations. In the inner loop (lines 3-13), a neighborhood structure is randomly selected from NS (line 4) to try the improvement of s (line 5). This improvement is made with a best improvement strategy, where the returned solution s' is either better than s or s itself. If s is improved by s' , then s is updated, NS is reset and the data structures on subsequences are updated using s as reference (lines 6-9), otherwise, the current neighborhood structure is removed from NS (lines 10-11). Once all neighborhood structures fail to improve s , this solution is returned (line 14).

Algorithm 3 RVND(s)

```

1:  $NS \leftarrow \text{SetNeighborhoodList}()$ ;
2:  $\text{InitializeDataStructuresOnSubsequences}(s)$ ;
3: while  $NS \neq \emptyset$  do
4:    $nh \leftarrow \text{SelectRandomNeighborhood}(NS)$ ;
5:    $s' \leftarrow \text{FindBestNeighbor}(nh, s)$ ;
6:   if  $f(s') < f(s)$  then
7:      $s \leftarrow s'$ ;
8:      $NS \leftarrow \text{SetNeighborhoodList}()$ ;
9:      $\text{UpdateDataStructuresOnSubsequences}(s)$ ;
10:  else
11:     $NS \leftarrow NS - \{nh\}$ ;
12:  end if
13: end while
14: return  $s$ 

```

The perturbation mechanism used in the local search is implemented by a move known as Double-bridge [26], and it was initially proposed for the TSP to escape from local optimum. Considering a solution to be perturbed, as the pictorial example shown in Figure 3.1, this move consists in a plain removal of four arcs and insertion of four other arcs, keeping the feasibility of the solution.

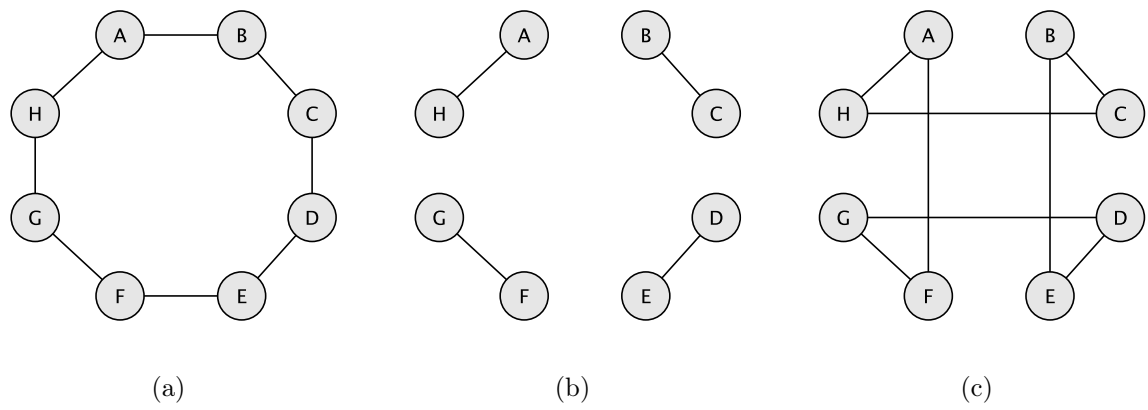


Figure 3.1: A double-bridge move applied to a solution

Chapter 4

DM-GILS-RVND: the First Proposal

Divided into three sections, this chapter aims at presenting the process of incorporating data mining techniques into GILS-RVND. In Section 4.1, data mining concepts that base the hybridization of heuristics are introduced. Next, Section 4.2 demonstrates how these concepts are incorporated into heuristics, presenting a general framework seen so far in the literature. Finally, Section 4.3 details our first hybrid GILS-RVND with data mining, which is explained by means of algorithms and illustrative examples.

4.1 Data Mining Concepts

Data Mining (DM) is the process of discovering knowledge from datasets, which can be described in terms of rules and patterns [19]. Mining this knowledge requires specific techniques to transform bulks of dataset registers into useful information. A common application of data mining techniques can be seen in e-commerce datasets, where these techniques are used to extract interesting frequent itemsets representing customer preferences, in such way that new business strategies can be developed to increase the sales.

In a dataset composed of transactions, where each transaction corresponds to a set of elements from an application domain, different relationships among data can be mined in terms of frequent itemsets (patterns). An itemset, which is a subset of items of the application domain, is mined if its support, a percentage indicator of its occurrence in the dataset, is greater or equal to a given minimum support, so known as frequent itemset. Therefore, the frequent itemsets mining process consists in identifying all frequent itemsets in the dataset regarding a given minimum support. Besides that, maximal frequent itemsets can also be mined, where a maximal frequent itemset corresponds to a frequent itemset that is not a subset of any frequent itemset.

Having these concepts stated, let $E = \{e_1, e_2, e_3, \dots, e_n\}$ be a set of the elements from the application domain. A transaction t is a subset of E and a dataset D is a set of transactions. A frequent itemset F , with support sup , is a subset of E which occurs in at least $(sup \times 100)\%$ of the transactions in D . Thus, identifying all frequent itemsets in D with a minimum sup (sup_{min}) specified as a parameter is the Frequent Itemset Mining (FIM) problem, where several efficient algorithms have been proposed, such as Apriori [4] and FP-Growth [20]. The algorithm FP-Max* was proposed to mine efficiently maximal frequent itemsets [17].

For exemplification purposes, Table 4.1 provides a dataset D composed of five transactions. Let $E = \{1, 2, 3, 4, 5\}$ be a set of all elements of the application domain and sup_{min} equal to 80%. By applying a FIM technique in D , the frequent itemsets extracted are $\{2\}$, $\{3\}$, $\{5\}$, $\{2, 3\}$ and $\{2, 5\}$, since they occur in, at least, 80% of all transactions of D . Note that, in case of mining maximal frequent itemsets, only $\{2, 3\}$ and $\{2, 5\}$ are extracted as these itemsets are not subsets of any frequent itemsets.

#	Transaction
t_1	$\{1, 2, 3, 4, 5\}$
t_2	$\{1, 2, 3, 5\}$
t_3	$\{2, 3, 4, 5\}$
t_4	$\{1, 2, 5\}$
t_5	$\{2, 3, 4\}$

Table 4.1: Dataset D composed of five transactions

The following sections display, respectively, the general framework of the hybridization of heuristics with data mining done so far in the literature, and how this hybridization is employed in GILS-RVND.

4.2 Hybridization of Heuristics with Data Mining

The process of incorporating data mining techniques into heuristics comes from the idea of extracting patterns of near-optimal solutions which can be used to lead the search for better solutions. Proposed in [35, 36], the Data Mining GRASP (DM-GRASP) is a simple two-phase GRASP separated by a data mining process. In its first phase, which lasts for a significant number of GRASP iterations, the best d visited solutions are stored in a set D , which is named elite set. Indeed, each solution of D represents a transaction t . After the first phase, the data mining process, which is an application of a FIM technique, extracts a list of patterns P from D using a predefined sup_{min} .

Next, the second phase of DM-GRASP executes the remaining GRASP iterations, where, at each iteration, a pattern $p \in P$ is given as input to a hybrid constructive method. This method generates initial solutions based on patterns, which will guide the local search phase. It is noteworthy to highlight that this hybridization process does not modify the local search phase in any aspect.

Initially, the DM-GRASP was successfully applied to optimization problems that represent solutions as sets of elements [8, 27, 33, 40, 41]. The mining of itemsets for this category of problems was straightforward, since itemsets are sets of elements. However, this technique cannot be directly employed when the ordering of the elements is relevant in the solution, in this case named as permutation.

In [18], a DM-GRASP-based strategy for a Traveling Salesman Problem (TSP) variant was proposed to deal with permutations as follows. Instead of the intuitive mining of vertices, which are the solution elements, this technique considered the mining of arcs – which naturally holds the ordering of the vertices. Basically, each pair of vertices (i, j) , an arc of a solution, is mapped to an identifier that represents it $(a_{i \rightarrow j})$. This mapping is performed to all solutions from the elite set, i.e., transforming the solutions into sets of elements, which are then easily dealt by a FIM technique.

As the MLP solutions are also represented by permutations, this mapping approach was adopted in this work. In Section 4.3, the hybridization with data mining applied into GILS-RVND, the hybrid constructive method and the adopted mapping of arcs are specified.

4.3 Hybridization of GILS-RVND with Data Mining

This section describes the first hybrid heuristic with data mining proposed in this work, which is named DM-GILS-RVND and is based on DM-GRASP strategy [35]. The DM-GILS-RVND is divided into two phases with half of the total number of iterations each, separated by the data mining process. In the first phase, which is identical to the original heuristic, the best solutions found are stored in the elite set D of size d . A solution s is eligible to be inserted into D either s is better than the worst solution of D , or D is not full. Also, D does not allow identical solutions. After the first phase, the data mining process is performed and outputs a set of patterns P with support greater or equal to a given minimum support sup_{min} . Then, the second phase uses, at each iteration, a pattern p from P to construct an initial solution by means of a hybrid constructive procedure. At

the end, the best solution found is returned.

Figure 4.1 wraps the main idea behind the hybridization applied to GILS-RVND in an illustrative scenario. Assume two solutions, S1 and S2, stored in an elite set of size 2, represented by Figures 4.1(a) and 4.1(b), and sup_{min} equal to 100%. Since MLP solutions are permutations, the mapping approach developed in [18] is adopted for constructing pattern-based solutions for this problem. As a result, a pattern, i.e., a set of mined arcs, is extracted from the elite set. This pattern is highlighted in dashed lines and gray backgrounds in Figure 4.1(c). Based on this pattern, a solution, shown in Figure 4.1(d), is then generated by the hybrid constructive procedure, which will be later explained. Note that all mined arcs are inserted into the resulting solution.

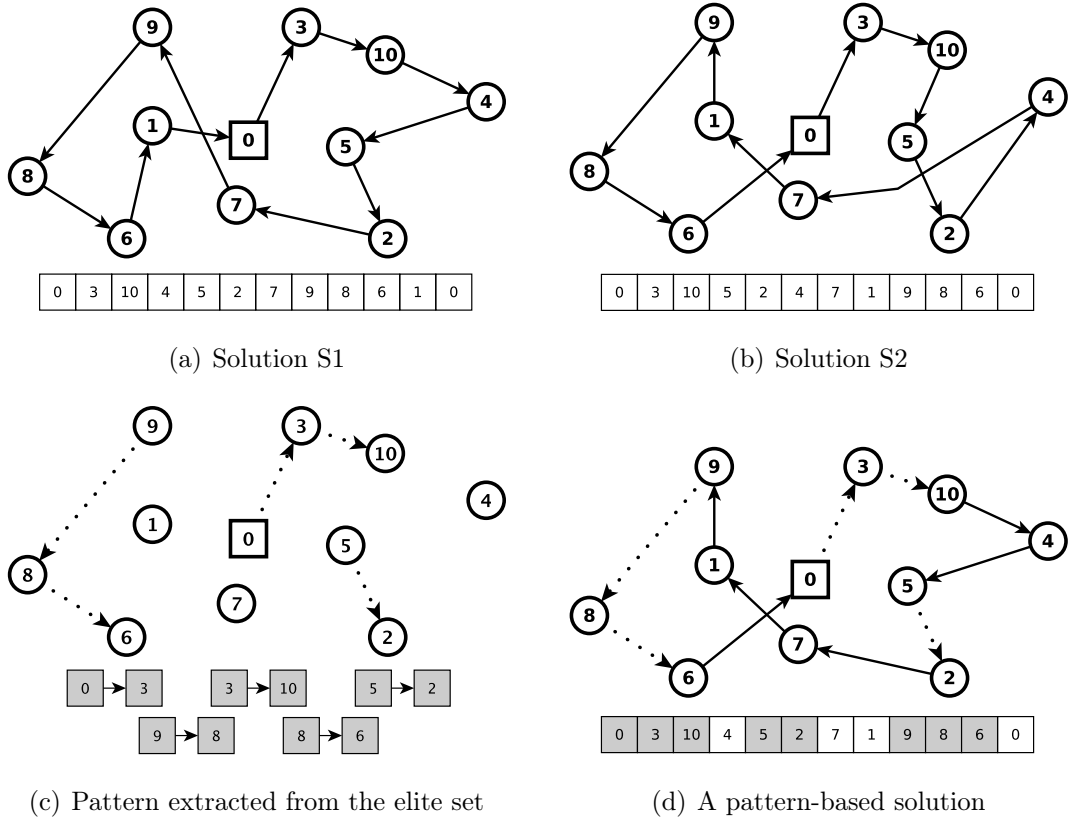


Figure 4.1: Use of mined frequent arcs to generate initial solutions

Specifically, this hybridization with data mining is presented in Algorithm 4. At the beginning, the cost of the best global solution s^* is initialized (line 1). In the first phase of the strategy (lines 2-19), which comprises the first half of the iterations, the algorithm remains the same as GILS-RVND, except for the presence of the $UpdateEliteSet(D, d, s)$ function (line 8). This function inserts s , which is a solution returned by the RVND procedure, in D if s is unique in D and its cost value is better than the worst solution cost value in D . After the first phase, the data mining process is executed, requiring

D , d , sup_{min} and $MaxP$ as input parameters, where $MaxP$ is an integer threshold for the number of selected patterns (line 20). In our hybridization, the list of patterns P (of size $MaxP$) is composed of the largest patterns found by the FIM technique. These patterns are kept in P in increasing order by the number of arcs. In the second phase of the algorithm (lines 21-37), which executes the second half of iterations, a hybrid constructive procedure replaces the original constructive procedure (line 22). This hybrid procedure, which is later explained, requires $\alpha \in R$ and a pattern p selected in a round-robin way from P . Next, the remainder lines of the second part are equal to their respective part of GILS-RVND. At the end of all multi-start iterations, s^* is returned (line 38).

Regarding the construction of the elite set, it is important to highlight that the inserted solutions can come from different multi-start iterations or, eventually, from an unique iteration. Experiments using insertion criteria that increase the degree of diversification of the elite set solutions, such as limiting the insertion of at most two solutions per iteration into it, have been executed, obtaining worse results.

In this work, the extraction of the patterns is performed by the FPM^{*} algorithm¹, which is a efficient implementation for extracting maximal frequent itemsets [17].

For generating initial solutions using patterns, the hybrid constructive procedure of the DM-GILS-RVND tries to insert segments of solution when possible, where a segment of solution is a sequence of consecutive arcs found in the given pattern. Considering the example showed in Figure 4.1(c), three segments of solution were found in the extracted pattern, which were used to generate a solution, illustrated in 4.1(d). It is important to emphasize that during this construction, only the first vertex of a segment is available to be chosen, so that, when the first vertex is selected, the entire segment is inserted at once.

This hybrid constructive procedure is detailed in Algorithm 5. Firstly, the depot (vertex 0) is placed into the partial solution s (line 1). Then, the Consecutive Arcs Lists (CAL) are generated using the pattern p as input (line 2). Each list in CAL is basically a sequence of adjacent arcs found in p . Based on CAL, the Customers List (CL) of this algorithm is then created with those that do not exist in any list of CAL and those that are the very first customer of each list in CAL (line 3). Next, if the vertex 0 is the first vertex of a list in CAL, then this list is appended to s and removed from CAL (lines 4-7). After, the variable r receives the last inserted vertex in s (line 8). In the inner loop (lines 9-21), r is used as reference in the sorting of CL (line 10), as already explained in Algorithm 2. Next, the RCL is filled with the $(\alpha \times 100)\%$ best customers from CL

¹Available at <http://fimi.ua.ac.be/>.

to, then, a customer c be randomly selected from RCL (lines 10-12). The customer c is checked in all first elements of each CAL list and, if found, the entire segment is inserted into s and removed from CAL (lines 13-15), otherwise, only c is inserted into s (lines 16-17). After this conditional block, r is updated and c is removed from CL (lines 19-20). When CL becomes empty, s is returned by the algorithm (line 22).

Algorithm 4 DM-GILS-RVND($I_{Max}, I_{ILS}, R, D, d, Sup, MaxP$)

```

1:  $f(s^*) \leftarrow \infty$ ;
2: for  $i = 1, \dots, I_{Max}/2$  do
3:    $s \leftarrow ConstructiveProcedure(\alpha \in R)$ ;
4:    $s' \leftarrow s$ ;
5:    $iterILS \leftarrow 0$ ;
6:   while  $iterILS < I_{ILS}$  do
7:      $s \leftarrow RVND(s)$ ;
8:      $UpdateEliteSet(D, d, s)$ ;
9:     if  $f(s) < f(s')$  then
10:       $s' \leftarrow s$ ;
11:       $iterILS \leftarrow 0$ ;
12:     end if
13:      $s \leftarrow Perturb(s')$ ;
14:      $iterILS \leftarrow iterILS + 1$ ;
15:   end while
16:   if  $f(s') < f(s^*)$  then
17:      $s^* \leftarrow s'$ ;
18:   end if
19: end for
20:  $P \leftarrow MinePatterns(D, d, Sup, MaxP)$ ;
21: for  $i = 1, \dots, I_{Max}/2$  do
22:    $s \leftarrow HybridConstructiveProc(\alpha \in R, p \in P)$ ;
23:    $s' \leftarrow s$ ;
24:    $iterILS \leftarrow 0$ ;
25:   while  $iterILS < I_{ILS}$  do
26:      $s \leftarrow RVND(s)$ ;
27:     if  $f(s) < f(s')$  then
28:        $s' \leftarrow s$ ;
29:        $iterILS \leftarrow 0$ ;
30:     end if
31:      $s \leftarrow Perturb(s')$ ;
32:      $iterILS \leftarrow iterILS + 1$ ;
33:   end while
34:   if  $f(s') < f(s^*)$  then
35:      $s^* \leftarrow s'$ ;
36:   end if
37: end for
38: return  $s^*$ 

```

Algorithm 5 HybridConstructiveProcedure(α, p)

```

1:  $s \leftarrow \{0\}$ ;
2:  $CAL \leftarrow \text{GenerateCAL}(p)$ ;
3:  $CL \leftarrow \text{GenerateCLFromCAL}(CAL)$ ;
4: if  $\{\exists cal \in CAL \mid 0 \text{ is the first customer in } cal\}$  then
5:    $s \leftarrow s \cup cal$ ;
6:    $CAL \leftarrow CAL - cal$ ;
7: end if
8:  $r \leftarrow \text{SelectLastCustomer}(s)$ ;
9: while  $CL \neq \emptyset$  do
10:   $CL \leftarrow \text{SortCL}(CL, r)$ ;
11:   $RCL \leftarrow \text{FillRCL}(CL, \alpha)$ ;
12:   $c \leftarrow \text{SelectRandomClient}(RCL)$ ;
13:  if  $\{\exists cal \in CAL \mid c \text{ is the first customer in } cal\}$  then
14:     $s \leftarrow s \cup cal$ ;
15:     $CAL \leftarrow CAL - cal$ ;
16:  else
17:     $s \leftarrow s \cup \{c\}$ ;
18:  end if
19:   $r \leftarrow \text{SelectLastCustomer}(s)$ ;
20:   $CL \leftarrow CL - \{c\}$ 
21: end while
22: return  $s$ 

```

Chapter 5

Computational Experiments for DM-GILS-RVND

This chapter reports several computational experiments used to compare GILS-RVND and DM-GILS-RVND performances and it is divided into four sections, described as follows. Section 5.1 presents the methodology used in our experiments. Based on this methodology, Sections 5.2 and 5.3 report experiments for the MLP variants, respectively for Hamiltonian circuits and for Hamiltonian paths. At last, Section 5.4 extends the evaluation of these two heuristics with extra analysis.

5.1 Comparison Methodology

The original GILS-RVND source-code was granted by the authors to develop our proposed DM-GILS-RVND. This source-code was implemented in C++ and compiled with the g++ 4.4.3, in single thread of a Intel®Core™ i7 3.40GHz with 16GB of RAM under GNU/Linux Ubuntu 14.04 (64-bits).

All computational experiments reported in [42] were fully re-executed in our research with the same set of parameters, the same g++ compiler and the same seeds of pseudo-random numbers. For these experiments, reported in Sections 5.2 and 5.3, $I_{Max} = 10$, $I_{ILS} = \min\{100, n\}$ and $R = \{0.00, 0.01, \dots, 0.25\}$ are given as input parameters. Also, as indicated in [42] and reproduced in our work, each heuristic was executed 10 times for each instance, which used a distinct seed of pseudo-random numbers. Exclusively for DM-GILS-RVND, $sup_{min} = 70\%$, $d = 10$ and $MaxP = 5$ were defined. In fact, $sup_{min} = 70\%$ was chosen based on the best trade-off between solution quality and computational time found in our experiments, different from the usual $sup_{min} = 20\%$, adopted in [27, 32,

40, 41], that presented impracticable averages of computational time due to the mining process. The elite set size $d = 10$ was based on the promising results of DM-GRASP seen in [27, 32, 40, 41], and $MaxP = 5$ was naturally determined by the five remaining multi-start iterations assigned to the second phase of DM-GILS-RVND.

For Sections 5.2 and 5.3, tables are used to report the results of the experiments. In these tables, the first column represents the tested instance, and the following one stands for its best known solution (BKS) in the literature. The next three columns show, respectively, the Best Solution, the Average Solution and the Average Time for each instance submitted to GILS-RVND in the already mentioned 10 executions. Likewise, the same is applied in the next three columns regarding DM-GILS-RVND. The last column demonstrates the percentage time gap of average time between the strategies. This value is calculated using the Equation 5.1, where *DMHeuristic* and *BaseHeuristic* mean, respectively, the value of the Average Time column of DM-GILS-RVND and GILS-RVND. Additionally, shown at the bottom of the table, the general Average of all time gaps and a counter of Better results are provided, being this last one a number of winnings of a heuristic over the another one, eligible only for Best Solution, Average Solution and Average Time columns.

$$Gap(\%) = (100(DMHeuristic - BaseHeuristic)) / BaseHeuristic \quad (5.1)$$

In order to support these evaluations, statistical significance tests (SSTs) were carried out on all instances results. By applying these tests, we intend to verify, in statistical terms, whether the results generated by both heuristics implied in a significant difference, reinforcing the application of DM into GILS-RVND, or not.

Fairly, all instances were submitted to paired SSTs with 5% of significance level, where a data sample stands for the 10 solution cost values obtained by a heuristic solving a instance. Therefore, two-sample tests were used to perform each paired SST, either using Wilcoxon test or Student's t-test. To decide which two-sample test had to be used, the normality of the data samples were beforehand verified by the Shapiro-Wilk test. If both data samples followed a normal distribution, the Student's t-test was applied, otherwise, the Wilcoxon test was used. With no exception, all SSTs reported in this work were carried out on R platform [34].

For interpreting the results of a statistical test, two hypothesis must be considered, the null hypothesis and the alternative hypothesis. Moreover, in a SST, it is important to decide between one-tailed and two-tailed tests. One-tailed tests are used when the relation

“ $A < B$ ” or “ $A > B$ ” is suitable, while two-tailed tests are assumed when the relation “ $A \neq B$ ” is relevant. Seeing this, as we intended to verify the performance of DM-GILS-RVND over GILS-RVND, one-tailed tests were chosen. Thus, the null hypothesis in our tests considered the algorithm A is better or equal to algorithm B, whereas the alternative hypothesis meant that the algorithm B performed better than algorithm A.

The p-value (numeric result from a SST) is used to determinate if the null hypothesis will be rejected or not. If the p-value is lower than the significance level, which was defined to 5%, the null hypothesis must be rejected and, hence, the alternative hypothesis is accepted, meaning that there was statistical significance. Otherwise, a p-value greater or equal to 5% means that the null hypothesis should not be rejected.

Broadening our evaluation between DM-GILS-RVND and GILS-RVND, Section 5.4 extends the comparison of both strategies towards different analysis.

5.2 Experiments for Hamiltonian Circuit

This section presents the computational results of GILS-RVND and DM-GILS-RVND for the MLP variant that considers Hamiltonian circuits as solutions, where two instance sets were tested. A set of 23 instances varying from 42 to 107 customers, originally selected in [1, 2] from TSPLib, and another set composed of all 56 instances with 120 to 1379 customers from TSPLib, which has not been previously tested in the literature.

Table 5.1 reports the results for the set of 23 instances. In this scenario, both heuristics found the optimal solutions in all 21 instances and matched the best known solutions in the remainder two instances, with DM-GILS-RVND performing in average of 5.06% faster than GILS-RVND. Considering the Better counter, GILS-RVND got two better average solutions over DM-GILS-RVND, whereas DM-GILS-RVND was faster than GILS-RVND in 17 instances. Although DM-GILS-RVND did not match or improve two average solutions, no statistical significance was observed in this set.

Regarding the 56-instance set for this MLP variant, Table 5.2 displays its computational results. It is noteworthy that no instance of this set has been recorded in the literature for this MLP variant, thus, BKS column is omitted. Seen at the GAP(%) Time column, the performance enhancement of DM-GILS-RVND is confirmed as the problem gets harder, achieving 21.09% less computational time in general average. For the Better counter, DM-GILS-RVND outperformed the original strategy in all aspects, being 19 wins against 5 loses for best solution, 30 wins against 14 loses for average solution, and 55

wins against one lose in average time. In statistical terms, five instances for the Student's t-test and two instances for the Wilcoxon test, where DM-GILS-RVND was better than GILS-RVND, were statistically significant. Beyond that, one instance was statistically significant for the Student's t-test where GILS-RVND was better than DM-GILS-RVND.

Instance	OPT/ BKS	GILS-RVND			DM-GILS-RVND			GAP(%) Time
		Best Solution	Average Solution	Average Time (s)	Best Solution	Average Solution	Average Time (s)	
dantzig42	12528	12528	12528.0	0.17	12528	12528.0	0.17	0.00
swiss42	22327	22327	22327.0	0.16	22327	22327.0	0.16	0.00
att48	209320	209320	209320.0	0.29	209320	209320.0	0.29	0.00
gr48	102378	102378	102378.0	0.31	102378	102378.0	0.29	-6.45
hk48	247926	247926	247926.0	0.28	247926	247926.0	0.28	0.00
eil51	10178	10178	10178.0	0.40	10178	10178.0	0.40	0.00
berlin52	143721	143721	143721.0	0.39	143721	143721.0	0.39	0.00
brazil58	512361	512361	512361.0	0.55	512361	512361.0	0.52	-5.45
st70	20557	20557	20557.0	0.99	20557	20557.0	0.94	-5.05
eil76	17976	17976	17976.0	1.52	17976	17976.0	1.44	-5.26
pr76	3445242	3445242	3445242.0	1.35	3455242	3455242.0	1.33	-1.48
pr76r	345427	345427	345427.0	1.38	345427	345427.0	1.28	-7.25
gr96	2097170	2097170	2097170.0	2.84	2097170	2097682.3	2.75	-3.17
rat99	57986*	57986	57986.0	5.30	57986	57986.0	5.02	-5.28
kroA100	983128	983128	983128.0	4.21	983128	983128.0	3.58	-14.96
kroB100	986008	986008	986008.0	4.13	986008	986008.0	4.01	-2.91
kroC100	961324	961324	961324.0	3.95	961324	961324.0	3.62	-8.35
kroD100	976965	976965	976965.0	4.06	976965	976965.0	3.47	-14.53
kroE100	971266	971266	971266.0	4.00	971266	971266.0	3.69	-7.75
rd100	340047	340047	340047.0	4.15	340047	340047.0	3.99	-3.86
eil101	27513*	27513	27513.0	5.79	27513	27519.8	5.00	-13.64
lin105	603910	603910	603910.0	3.57	603910	603910.0	3.44	-3.64
pr107	2026626	2026626	2026626.0	4.33	2026626	2026626.0	4.01	-7.39
Average	-	-	-	-	-	-	-	-5.06
Better	-	23	23	6	23	21	23	-

* - Optimality is not proven

Table 5.1: Results for instances selected from TSPLib in [1, 2]

5.3 Experiments for Hamiltonian Path

For the MLP variant, which considers Hamiltonian paths as solutions, we used 150 instances, divided into: a set of 10 instances varying between 70 and 532 customers selected from TSPLib, introduced in [39], and seven groups of 20 generated instances each with dimensions of 10, 20, 50, 100, 200, 500, and 1000 customers by the author of [39].

In Table 5.3, which reports for the group of instances that ranges from 70 to 532 customers, the computational time was reduced in all instances, achieving a general average of 12.87% faster than the original strategy. In terms of solution quality, DM-GILS-RVND only improved the average of att532 instance, the hardest instance of this set. No statistical significance was observed in this group.

Instance	GILS-RVND			DM-GILS-RVND			GAP(%) Time
	Best Solution	Average Solution	Average Time (s)	Best Solution	Average Solution	Average Time (s)	
gr120	363454	363569.5	9.54	363454	363584.8	8.24	-13.63
pr124	3154346	3154346.0	5.39	3154346	3154346.0	5.14	-4.64
bier127	4545005	4546378.8	9.25	4545005	4545005.0	7.80	-15.68
ch130	349874	349891.7	9.23	349874	349903.5	8.46	-8.34
pr136	6199268	6199805.4	17.30	6199268	6200032.6	14.11	-18.44
gr137	4061498	4061498.0	8.11	4061498	4061498.0	7.10	-12.45
pr144	3846137	3846137.0	9.11	3846137	3846137.0	9.06	-0.55
ch150	444424	444424.0	13.06	444424	444424.0	10.80	-17.30
kroA150	1825769	1825769.0	19.84	1825769	1825769.0	15.51	-21.82
kroB150	1786546	1786546.0	16.27	1786546	1786546.0	14.68	-9.77
pr152	5064566	5064566.0	11.23	5064566	5064566.0	10.45	-6.95
u159	2972030	2972204.2	14.21	2972030	2972291.3	12.88	-9.36
si175	1808532	1808532.0	19.14	1808532	1808532.0	14.92	-22.05
brg180	174750	174750.0	16.79	174750	174750.0	16.00	-4.71
rat195	218632	218763.2	44.69	218632	218760.6	37.06	-17.07
d198	1186049	1186098.6	38.28	1186049	1186086.2	31.55	-17.58
kroA200	2672437	2672444.2	42.23	2672437	2672437.0	33.70	-20.20
kroB200	2669515	2674486.0	42.00	2669515	2675761.6	36.48	-13.14
gr202	2909247	2914644.2	35.95	2909247	2912564.8	31.62	-12.04
ts225	13240046	13240046.0	26.60	13240046	13240533.0	27.28	2.56
tsp225	402783	403080.2	53.89	402783	402970.5	43.67	-18.96
pr226	7196869	7196869.0	34.29	7196869	7196869.0	28.86	-15.84
gr229	10725914	10729883.8	53.66	10725914	10729943.9	43.78	-18.41
gil262	285060	285527.1	96.12	285043	285343.5	76.34	-20.58
pr264	5471615	5471615.0	47.02	5471615	5471615.0	38.74	-17.61
a280	346989	347125.9	107.18	346989	347009.6	83.61	-21.99
pr299	6556628	6557983.4	104.92	6556628	6558164.9	78.64	-25.05
lin318	5619810	5629995.9	117.98	5619810	5630556.9	100.63	-14.71
rd400	2768830	2776672.7	350.84	2767608	2775101.2	278.61	-20.59
fl417	1874242	1874242.8	382.64	1874242	1874242.0	263.61	-31.11
gr431	21159702	21239150.9	336.98	21143311	21210280.6	264.24	-21.59
pr439	17829541	17887107.0	285.56	17829541	17876876.9	199.04	-30.30
pcb442	10301705	10323539.7	413.41	10290913	10321804.2	313.07	-24.27
d493	6684190	6691057.1	608.47	6680997	6688669.0	410.33	-32.56
att532	5613010	5632753.5	988.04	5622905	5630730.4	744.64	-24.63
ali535	31870389	31904676.6	880.76	31870389	31902870.9	587.16	-33.33
si535	12247211	12250679.7	498.76	12246397	12252151.6	340.83	-31.66
pa561	658870	661211.6	1155.32	660249	662216.9	918.00	-20.54
u574	9314596	9344178.4	1234.19	9313459	9350198.1	893.26	-27.62
rat575	1848869	1859221.1	1739.46	1847411	1856382.8	1345.04	-22.67
p654	7827273	7827639.2	1755.28	7827273	7827919.4	1263.97	-27.99
d657	14159477	14220133.3	2615.66	14125530	14188813.5	1868.86	-28.55
gr666	63571693	63731966.5	2296.23	63546987	63663647.1	1699.61	-25.98
u724	13506660	13558605.3	4651.76	13491605	13546178.5	3505.29	-24.65
rat783	3282794	3296069.6	7044.52	3272226	3290521.7	4740.52	-32.71
dsj1000	7646018508	7685887300.0	18068.70	7640607124	7671314634.0	12612.84	-30.20
dsj1000ceil	7646519008	7683329486.0	18543.76	7644298506	7680652520.0	12885.48	-30.51
pr1002	115550770	116178260.2	11963.29	115507699	115975798.5	8817.55	-26.29
si1032	46896355	46897662.4	2402.72	46896355	46896783.6	1975.55	-17.78
u1060	102508056	102759766.0	15680.50	102558414	102821622.2	10471.80	-33.22
vm1084	94760440	95053081.2	13894.43	94705227	94982553.5	9673.12	-30.38
pcb1173	30926325	31032128.8	20508.89	30891188	30972619.2	13903.73	-32.21
d1291	29383346	29477239.4	12171.21	29392621	29511969.3	8189.00	-32.72
rl1304	144886001	145596878.7	18617.53	144803181	145558912.3	12967.80	-30.35
rl1323	155697857	156360364.3	22758.06	155749119	156332300.1	15938.27	-29.97
nrv1379	35360407	35519379.7	49624.72	35327900	35475906.4	34547.99	-30.38
Average	-	-	-	-	-	-	-21.09
Better	37	25	1	51	43	55	-

Table 5.2: Results on the 56-instances set selected from TSPLib

Instance	BKS	GILS-RVND			DM-GILS-RVND			GAP(%) Time
		Best Solution	Average Solution	Average Time (s)	Best Solution	Average Solution	Average Time (s)	
st70	19215	19215	19215.0	0.94	19215	19215.0	0.91	-3.19
rat99	54984	54984	54984.0	5.81	54984	54984.0	5.22	-10.15
kroD100	949594	949594	949594.0	4.21	949594	949594.0	3.75	-10.93
lin105	585823	585823	585823.0	3.79	585823	585823.0	3.63	-4.22
pr107	1980767	1980767	1980767.0	4.96	1980767	1980767.0	4.50	-9.27
rat195	210191	210191	210335.9	44.99	210191	210385.6	37.84	-15.89
pr226	7100308	7100308	7100308.0	34.35	7100308	7100308.0	30.10	-12.37
lin318	5560679	5560679	5569819.5	124.15	5560679	5570488.8	97.39	-21.55
pr439	17688561	17688561	17734922.0	275.12	17702120	17735797.8	225.26	-18.12
att532	5577965	5581240	5597866.8	923.03	5581240	5596611.1	710.32	-23.04
Average	-	-	-	-	-	-	-	-12.87
Better	-	10	9	0	9	7	10	-

Table 5.3: Results for TSPLib instances selected in [39]

Table 5.4 reports the results for instance sets with 10, 20 and 50 customers at once. In these sets, both DM-GILS-RVND and GILS-RVND behaved similarly, achieving the optimal values of each instance within an average of 0.01, 0.02 and 0.05 seconds, respectively, for the sets of 10, 20 and 50 customers. Furthermore, for simplification, only average solutions are shown, since they match with their respective best solution.

Instance	GILS-RVND			DM-GILS-RVND		
	S10	S20	S50	S10	S20	S50
TRP-Sn-R1	1303	3175	12198	1303	3175	12198
TRP-Sn-R2	1517	3248	11621	1517	3248	11621
TRP-Sn-R3	1233	3570	12139	1233	3570	12139
TRP-Sn-R4	1386	2983	13071	1386	2983	13071
TRP-Sn-R5	978	3248	12126	978	3248	12126
TRP-Sn-R6	1477	3328	12684	1477	3328	12684
TRP-Sn-R7	1163	2809	11176	1163	2809	11176
TRP-Sn-R8	1234	3461	12910	1234	3461	12910
TRP-Sn-R9	1402	3475	13149	1402	3475	13149
TRP-Sn-R10	1388	3359	12892	1388	3359	12892
TRP-Sn-R11	1405	2916	12103	1405	2916	12103
TRP-Sn-R12	1150	3314	10633	1150	3314	10633
TRP-Sn-R13	1531	3412	12115	1531	3412	12115
TRP-Sn-R14	1219	3297	13117	1219	3297	13117
TRP-Sn-R15	1087	2862	11986	1087	2862	11986
TRP-Sn-R16	1264	3433	12138	1264	3433	12138
TRP-Sn-R17	1058	2913	12176	1058	2913	12176
TRP-Sn-R18	1083	3124	13357	1083	3124	13357
TRP-Sn-R19	1394	3299	11430	1394	3299	11430
TRP-Sn-R20	951	2796	11935	951	2796	11935
Better	20	20	20	20	20	20

Table 5.4: Results for instances generated in [39] considering 10, 20 and 50 customers

For the 100-customers set results, shown in Table 5.5, DM-GILS-RVND outperformed, in terms of computational time, the original strategy in all situations, with a general average of 10.54% less computational time. Considering solution quality, the heuristic based on DM and the original strategy matched all BKS, however, four average solutions of DM-GILS-RVND did not reach or improve their respective results by GILS-RVND. Although DM-GILS-RVND lost in these four situations, no instance was statistically significant in this set.

Instance	BKS	GILS-RVND			DM-GILS-RVND			GAP(%) Time
		Best Solution	Average Solution	Average Time (s)	Best Solution	Average Solution	Average Time (s)	
TRP-S100-R1	32779	32779	32779.0	4.33	32779	32779.0	3.75	-13.39
TRP-S100-R2	33435	33435	33435.0	4.61	33435	33435.0	4.20	-8.89
TRP-S100-R3	32390	32390	32390.0	4.33	32390	32390.0	4.03	-6.93
TRP-S100-R4	34733	34733	34733.0	4.45	34733	34733.0	3.89	-12.58
TRP-S100-R5	32598	32598	32598.0	5.43	32598	32598.0	4.48	-17.50
TRP-S100-R6	34159	34159	34159.0	4.83	34159	34159.0	4.31	-10.77
TRP-S100-R7	33375	33375	33375.0	5.34	33375	33375.0	4.31	-19.29
TRP-S100-R8	31780	31780	31780.0	4.29	31780	31780.0	3.57	-16.78
TRP-S100-R9	34167	34167	34167.0	4.56	34167	34167.5	4.18	-8.33
TRP-S100-R10	31605	31605	31605.0	4.13	31605	31605.0	3.67	-11.14
TRP-S100-R11	34188	34188	34198.5	4.72	34188	34219.5	4.48	-5.08
TRP-S100-R12	32146	32146	32146.0	4.38	32146	32146.0	4.03	-7.99
TRP-S100-R13	32604	32604	32604.0	4.75	32604	32604.0	4.41	-7.16
TRP-S100-R14	32433	32433	32433.0	3.89	32433	32433.5	3.54	-9.00
TRP-S100-R15	32574	32574	32574.0	4.36	32574	32574.0	3.90	-10.55
TRP-S100-R16	33566	33566	33566.0	4.87	33566	33566.0	4.39	-9.86
TRP-S100-R17	34198	34198	34198.0	5.65	34198	34198.0	5.41	-4.25
TRP-S100-R18	31929	31929	31929.0	4.34	31929	31929.0	3.95	-8.99
TRP-S100-R19	33463	33463	33463.0	4.91	33463	33463.0	4.35	-11.41
TRP-S100-R20	33632	33632	33632.2	5.35	33632	33632.4	4.77	-10.84
Average	-	-	-	-	-	-	-	-10.54
Better	-	20	20	0	20	16	20	-

Table 5.5: Results on the 100-customers instances generated in [39]

Considering Table 5.6, which reports the results on the 200-customers set, the total reduction of effort by DM-GILS-RVND achieved an average of 18.19% less computational time. In terms of solution quality, no strategy behaved with a superior performance: both achieved all BKS, and DM-GILS-RVND improved nine average solutions, drawing in one instance, and losing in 10 instances. When SSTs were applied onto these heuristics, only one instance, where DM-GILS-RVND was better than GILS-RVND, was statistically significant.

For the set of 500 customers, Table 5.7 shows that DM-GILS-RVND outperformed the original strategy in aspects of best solution, average solution and computational time. In terms of best solution, DM-GILS-RVND improved 15 instance results achieved by GILS-RVND, matched four results and lost in only one situation. For average solution, the DM

Instance	BKS	GILS-RVND			DM-GILS-RVND			GAP(%) Time
		Best Solution	Average Solution	Average Time (s)	Best Solution	Average Solution	Average Time (s)	
TRP-S200-R1	88787	88787	88794.6	44.08	88787	88812.1	33.10	-24.91
TRP-S200-R2	91977	91977	92013.1	41.02	91977	92009.8	35.24	-14.09
TRP-S200-R3	92568	92568	92631.2	40.50	92568	92612.1	32.60	-19.51
TRP-S200-R4	93174	93174	93192.3	43.48	93174	93180.4	34.91	-19.71
TRP-S200-R5	88737	88737	88841.2	42.65	88737	88861.9	36.31	-14.87
TRP-S200-R6	91589	91589	91601.9	42.43	91589	91596.3	34.61	-18.43
TRP-S200-R7	92754	92754	92763.2	43.75	92754	92777.6	34.91	-20.21
TRP-S200-R8	89048	89048	89049.0	45.13	89048	89051.0	39.14	-13.27
TRP-S200-R9	86326	86326	86326.0	39.51	86326	86326.0	32.44	-17.89
TRP-S200-R10	91552	91552	91596.5	44.63	91552	91602.6	35.84	-19.70
TRP-S200-R11	92655	92655	92700.6	43.90	92655	92702.5	39.15	-10.82
TRP-S200-R12	91457	91457	91504.1	46.08	91457	91521.7	38.64	-16.15
TRP-S200-R13	86155	86155	86181.4	43.88	86155	86184.7	35.59	-18.89
TRP-S200-R14	91882	91882	91929.1	42.60	91882	91892.7	35.74	-16.10
TRP-S200-R15	88912	88912	88912.4	42.28	88912	88912.0	36.58	-13.48
TRP-S200-R16	89311	89311	89364.7	47.08	89311	89316.6	36.46	-22.56
TRP-S200-R17	89089	89089	89118.3	42.92	89089	89097.2	35.54	-17.19
TRP-S200-R18	93619	93619	93676.6	46.36	93619	93641.5	37.09	-20.00
TRP-S200-R19	93369	93369	93401.6	42.75	93369	93504.1	34.48	-19.35
TRP-S200-R20	86292	86292	86292.0	42.42	86292	86296.9	31.14	-26.59
Average	-	-	-	-	-	-	-	-18.19
Better	-	20	11	0	20	10	20	-

Table 5.6: Results on the 200-customers instances generated in [39]

strategy improved 15 average solutions and lost in five cases. Computational times from GILS-RVND were reduced by DM-GILS-RVND in average of 25.55%. Moreover, four instances were statistically significant for the Student’s t-test, where DM-GILS-RVND performed better than the original heuristic.

Table 5.8 reports the computational results for the set of 1000 customers. Regarding the best solution aspect, DM-GILS-RVND improved 18 instance results of GILS-RVND, while, for average solution, the DM strategy outperformed 15 instance results. Computational times were improved by DM-GILS-RVND with average of 31.23% less computational time than GILS-RVND. Interestingly, the best general improvement – both in terms of solution quality and computational time – was attained by DM-GILS-RVND over GILS-RVND in this set, which is the hardest one for this MLP variant. For SSTs, seven instances for the Student’s t-test and two instances for the Wilcoxon test were statistically significant, where DM-GILS-RVND performed better than GILS-RVND.

As also seen in Section 5.2, DM-GILS-RVND had outstanding performances against GILS-RVND when the instance size gets larger, i.e., the problem gets harder, chiefly for instances above 500 customers. The following section presents different comparative analysis between these two heuristics.

Instance	BKS	GILS-RVND			DM-GILS-RVND			GAP(%) Time
		Best Solution	Average Solution	Average Time (s)	Best Solution	Average Solution	Average Time (s)	
TRP-S500-R1	1841386	1841386	1856018.7	830.85	1841386	1852238.0	620.18	-25.36
TRP-S500-R2	1815664 [†]	1816568	1823196.9	724.17	1817288	1821667.9	502.99	-30.54
TRP-S500-R3	1826855 [†]	1833044	1839254.2	761.86	1831357	1838557.2	580.34	-23.83
TRP-S500-R4	1804894 [†]	1809266	1815876.4	810.63	1808563	1816765.0	589.33	-27.3
TRP-S500-R5	1821250 [†]	1823975	1834031.7	734.32	1823135	1831575.2	565.36	-23.01
TRP-S500-R6	1782731 [†]	1786620	1790912.4	796.28	1785217	1789675.6	596.65	-25.07
TRP-S500-R7	1847999	1847999	1857926.6	781.01	1847089	1854324.8	609.98	-21.9
TRP-S500-R8	1819636	1820846	1829257.3	769.33	1820639	1829831.8	580.50	-24.54
TRP-S500-R9	1733819	1733819	1737024.9	693.82	1730561	1736278.3	545.16	-21.43
TRP-S500-R10	1761174 [†]	1762741	1767366.3	784.64	1762197	1767912.7	587.06	-25.18
TRP-S500-R11	1797881	1797881	1801467.9	741.50	1797881	1802957.9	536.77	-27.61
TRP-S500-R12	1774452	1774452	1783847.1	766.23	1774452	1781588.2	564.99	-26.26
TRP-S500-R13	1863905 [†]	1873699	1878049.4	797.54	1867803	1875973.1	556.95	-30.17
TRP-S500-R14	1799171	1799171	1805732.9	835.86	1798130	1803379.2	585.61	-29.94
TRP-S500-R15	1785263 [†]	1791145	1797532.9	800.69	1790524	1795858.3	587.39	-26.64
TRP-S500-R16	1804392 [†]	1810188	1816484.0	761.93	1808183	1814941.6	590.35	-22.52
TRP-S500-R17	1825748	1825748	1834443.2	738.57	1824674	1832635.8	533.61	-27.75
TRP-S500-R18	1825615 [†]	1826263	1833323.7	780.31	1826263	1831998.5	636.26	-18.46
TRP-S500-R19	1776855 [†]	1779248	1782763.9	773.39	1774846	1785575.3	553.04	-28.49
TRP-S500-R20	1820813	1820813	1830483.3	726.50	1820713	1829740.0	544.20	-25.09
Average	-	-	-	-	-	-	-	-25.55
Better	-	5	5	0	15	19	20	-

[†] cost values from [37]

Table 5.7: Results on the 500-customers instances generated in [39]

5.4 Complementary Analyses

This section presents three distinct complementary assessments that consider different viewpoints of analyses to better understand the behaviors of GILS-RVND and DM-GILS-RVND. The first one, presented in Subsection 5.4.1, illustrates the impact of the usage of patterns into initial solutions compared to initial solutions built by the constructive method of GILS-RVND. In the next analysis, Subsection 5.4.2 shows experiments of time convergence of GILS-RVND and DM-GILS-RVND to targets (cost values of solutions). Subsection 5.4.3 presents fair comparisons between DM-GILS-RVND and GILS-RVND, where the same amount of time is given to both heuristics as stopping criterion instead of the number of multi-start iterations, as done in Sections 5.2 and 5.3.

5.4.1 Impact of the Usage of Mined Patterns

In this subsection, an execution using the TRP-S1000-R1 instance and another one with TRP-S1000-R7 instance were selected to demonstrate the impact of the usage of mined patterns into initial solutions. For each experimented instance, two figures show the results of each heuristic phase: the constructive phase and the local search phase, where the multi-start iterations were arranged at the abscissa axis and cost values of solutions

Instance	BKS	GILS-RVND			DM-GILS-RVND			GAP(%) Time
		Best Solution	Average Solution	Average Time (s)	Best Solution	Average Solution	Average Time (s)	
TRP-S1000-R1	5107395	5107395	5133698.3	19889.15	5099593	5122436.1	14123.42	-28.99
TRP-S1000-R2	5106161	5106161	5127449.4	19218.18	5084762	5112406.6	13532.12	-29.59
TRP-S1000-R3	5096977	5096977	5113302.9	18798.18	5089701	5111390.5	12553.89	-33.22
TRP-S1000-R4	5112465 [†]	5118006	5141392.6	18493.11	5110184	5142219.9	12404.32	-32.92
TRP-S1000-R5	5097991 [†]	5103894	5122660.7	18906.29	5085136	5116167.7	13124.07	-30.58
TRP-S1000-R6	5109946 [†]	5115816	5143087.1	19143.87	5102671	5132201.1	13862.07	-27.59
TRP-S1000-R7	4995703 [†]	5021383	5032722.0	17681.02	4984950	5014796.3	13154.44	-25.60
TRP-S1000-R8	5109325	5109325	5132722.6	18065.24	5109325	5129942.9	12781.96	-29.25
TRP-S1000-R9	5046566 [†]	5052599	5073245.3	17979.62	5045262	5070469.7	11758.44	-34.60
TRP-S1000-R10	5060019 [†]	5078191	5093592.6	17596.33	5070109	5089275.6	12274.40	-30.24
TRP-S1000-R11	5031455 [†]	5041913	5066161.5	18307.69	5052459	5066612.8	11826.31	-35.40
TRP-S1000-R12	5029792	5029792	5051235.2	19149.54	5030837	5043866.1	12940.67	-32.42
TRP-S1000-R13	5102520	5102520	5131437.5	19604.19	5098034	5123032.4	13895.03	-29.12
TRP-S1000-R14	5092861 [†]	5099433	5118980.6	18974.58	5089565	5116989.7	13171.29	-30.58
TRP-S1000-R15	5131013 [†]	5142470	5174493.2	18889.53	5123240	5166046.9	13687.70	-27.54
TRP-S1000-R16	5064094 [†]	5073972	5090280.5	18206.27	5070422	5091420.9	11962.50	-34.29
TRP-S1000-R17	5052283 [†]	5071485	5084450.4	18571.62	5065952	5082717.2	11847.47	-36.21
TRP-S1000-R18	5005789 [†]	5017589	5037094.0	19745.37	5003047	5038269.6	12966.44	-34.33
TRP-S1000-R19	5064873 [†]	5076800	5097167.6	19790.69	5065743	5087261.2	13365.27	-32.47
TRP-S1000-R20	4977262	4977262	5002920.6	18715.65	4970735	5003599.3	13150.60	-29.73
Average	-	-	-	-	-	-	-	-31.23
Better	-	2	5	0	18	15	20	-

[†] cost values from [37]

Table 5.8: Results on the 1000-customers instances generated in [39]

at the ordinate axis. For simplification, as DM-GILS-RVND and GILS-RVND have the same behavior in the first half of the multi-start iterations, one line is used to represent both heuristics.

To picture the analyses of the TRP-S1000-R1 instance, Figure 5.1(a) demonstrates that the obtained cost values from the data mining heuristics drop dramatically due to the usage of a mined pattern, whereas the cost values from GILS-RVND roughly remain at the same baseline. Being specific at this point, the used pattern contained 940 arcs, which was directly employed in the construction of the initial solution. For the local search phase, displayed in Figure 5.1(b), from the DM process onwards, the DM-GILS-RVND always presented better cost values compared to their respective values obtained by GILS-RVND. We believe that better constructed solutions from patterns allowed the local search to be more effective. Considering both figures, the best global solution was found in the local search phase of DM-GILS-RVND at the tenth iteration.

In the second scenario, which regards the TRP-S1000-R7 instance, the usage of a mined pattern was again significant in generating high-quality initial solutions. As shown in Figure 5.2(a), four out of five initial solutions based on patterns had better cost values than those constructed by the original constructive method. For the local search phase, Figure 5.2(b) demonstrates that again DM-GILS-RVND achieved better cost values com-

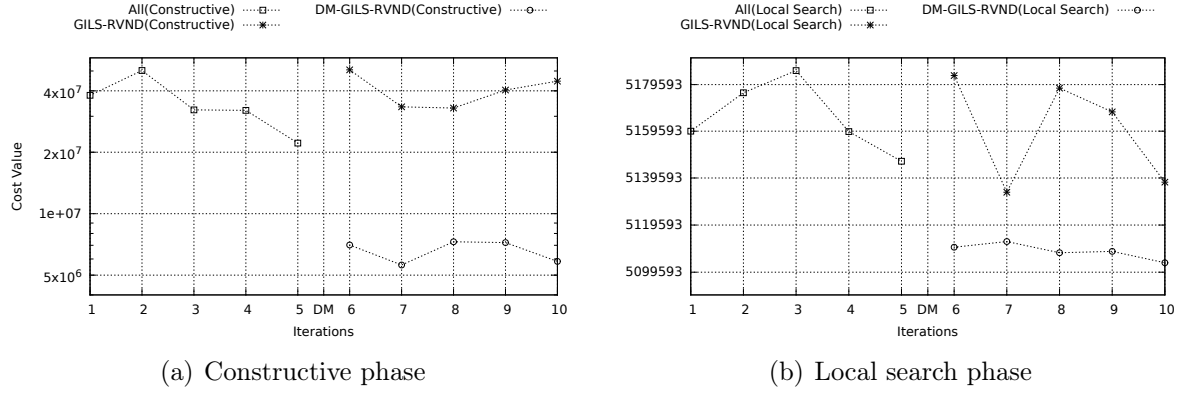


Figure 5.1: Cost values per iteration - TRP-S1000-R1 instance

pared to their respective values in GILS-RVND, achieving its best global at the tenth iteration.

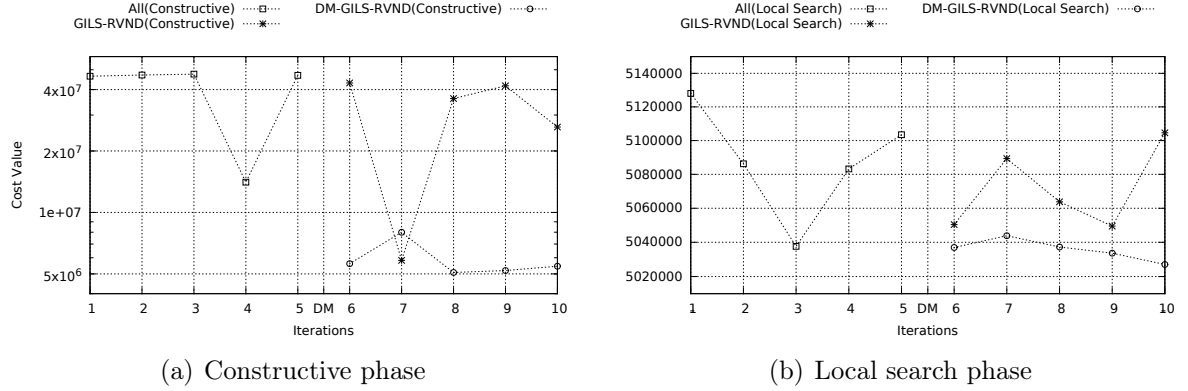


Figure 5.2: Cost values per iteration - TRP-S1000-R7 instance

5.4.2 Analyses of Time Convergence

In order to evaluate algorithms with random components, Time-to-target (TTT) plots are used to analyse their behaviors [5]. Particularly, a TTT plot displays the probability (ordinate axis) of an algorithm achieve a solution at least as good as a given target within a running time (abscissa axis).

For the TTT experiments reported in this subsection, two targets were chosen for the kroA200 instance, and other two targets for the pr299 instance, regarding the MLP version of Hamiltonian circuits. The first target 2672445 for the kroA200 instance was chosen based on the average solution obtained by GILS-RVND, as reported in Table 5.2, and the second target 2677290 stands for the worst solution cost value reached by GILS-RVND. The targets for the pr299 instance were chosen based on the average solution (6557983) achieved by GILS-RVND and the average solution (6558164) reached by DM-

GILS-RVND, both reported in Table 5.2. For each tested target, 100 executions using distinct seeds of pseudo-random numbers were used in the experiments.

Figure 5.3 illustrates the results for the kroA200 instance. Having all points for the easy target (2677290) plotted in Figure 5.3(a), we can observe that the behaviors of DM-GILS-RVND and GILS-RVND were very similar. On the other hand, the plot used to represent the TTT plot for the hard target (2672445), shown in Figure 5.3(b), demonstrated distinct performances between the heuristics. For example, the GILS-RVND presented a probability around 65% to reach the given target in 25 seconds, while DM-GILS-RVND presented a probability near 90% to reach the target within the same time.

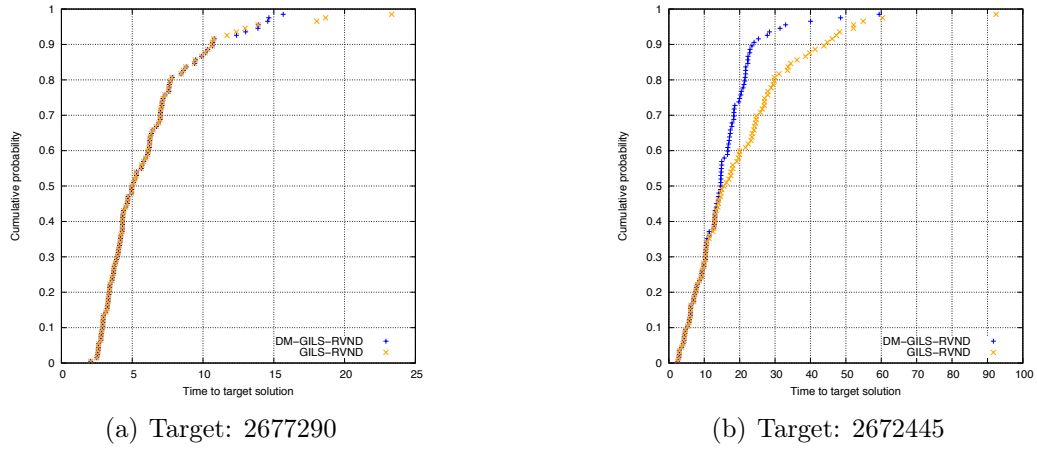


Figure 5.3: TTT plots for kroA200 instance

Figure 5.4 presents the TTT plots for the pr299 instance. Considering the easy target (6558165), illustrated in Figure 5.4(a), both heuristics virtually performed the same. Looking at the TTT plot for the hard target (6557983), shown in Figure 5.4(b), again we can observe a better performance of DM-GILS-RVND. For example, the GILS-RVND strategy presented a probability around 80% to reach the given target in 200 seconds, whereas the probability of DM-GILS-RVND reaching the target within the same time is about 92%.

Considering the TTT plots showed in this subsection, DM-GILS-RVND demonstrated to converge faster than GILS-RVND when hard targets are given as input, whilst, for easy targets, both heuristics remained with similar performances.

5.4.3 Complementary Experiments

Shown in Sections 5.2 and 5.3, the stopping criterion defined for the heuristics was defined as the number of multi-start iterations, specifically, when the tenth iteration ends.

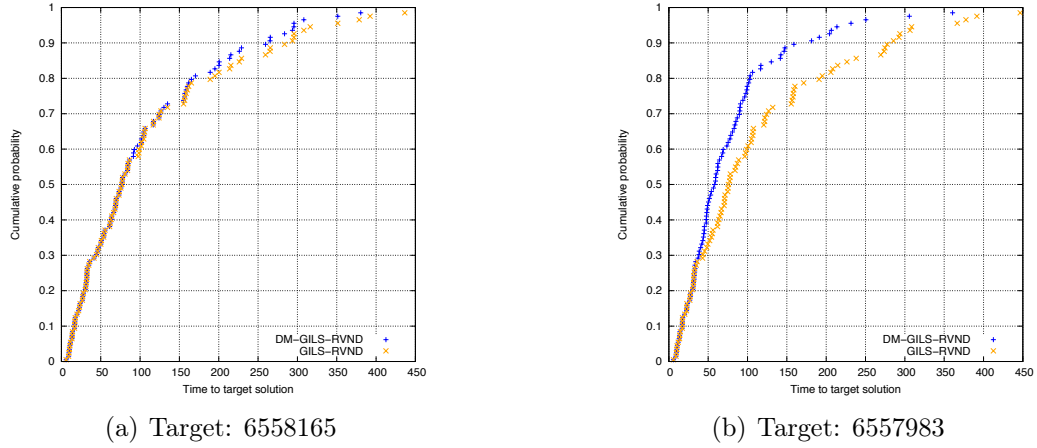


Figure 5.4: TTT plots for pr299 instance

However, as both heuristics performed in different amounts of time, this section presents a fair comparison between DM-GILS-RVND and GILS-RVND involving the same computational time as stopping criterion.

In order to verify the stopping criteria based on running time, two checkpoints were inserted into DM-GILS-RVND. These checkpoints were placed after the hybrid constructive method and after any complete neighborhood evaluation on the RVND method. Once the running time is reached, the best global solution found is forced to return and the algorithm stops.

Preliminary experiments showed the overheads of these checkpoints are worthless, even for large instances, being accurate enough for the purpose established in this analysis. Besides that, all instance sets previously used to compare GILS-RVND and DM-GILS-RVND, in Sections 5.2 and 5.3, have been tested in this complementary evaluation, where their computational results are fully reported in Appendix A. In this subsection, we will only show their summaries in tables.

The reported Tables 5.9 to 5.15 summarize the results in an $A-B-C$ format, where A means the number of results that the heuristic on the row improved the heuristic on the column, B represents a tie, and C stands for the number of results that the heuristic on the column improved the results of the heuristic on the row. For SST (Statistical Significance Tests) results, a $D-E$ format was adopted, where D indicates the number of instances for which the heuristic on the row had a better performance over the heuristic on the column with statistical significance, while E represents the occurrences of statistical significance for the heuristic on the column when performed better than the heuristic on the row. To complement these tables, their respective results from Sections 5.2 and

5.3, which demonstrates direct comparisons between DM-GILS-RVND and GILS-RVND results, are also shown in order to verify the progress of using this equal-time comparison.

The next two subsections show, respectively, the experiments for Hamiltonian circuits and Hamiltonian paths.

5.4.3.1 Experiments for Hamiltonian Circuit

For the experiments considering the Hamiltonian circuit version of MLP, both sets of instances seen in Section 5.2, which are a 23-instance set selected in [39] and a 56-instance set selected from TSPLib, were submitted to these extended experiments.

Table 5.9, which summarizes Table A.1 in Appendix A, represents the results on the 23-instance set for this problem version. Since 21 instances have no further improvement due to the obtained average solutions matched their respective optimal values, only two instances were tested in this subsection. Considering these two remaining ones, DM-GILS-RVND improved the original average solutions previously obtained, so, matching all results found by GILS-RVND in the same computational time.

Summary from Table A.1			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	0-2-0	0-2-0	0-0

Summary from Table 5.1			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	0-2-0	2-0-0	0-0

Table 5.9: Results for TSPLib instances selected in [1, 2]

Submitted to this equal-time comparison, the results for the 56-instances set, summarized in Table 5.10 (fully reported in Table A.2), presented an overall improvement compared to the results reported in Table 5.2. In these new experiments, the DM strategy outperformed even more the results of GILS-RVND, where, for the best solution aspect, DM-GILS-RVND improved 20 instance results, tied in 32 occasions, and lost in four situations. For average solutions, the DM strategy improved 35 instances, tied in 10 instance results, and lost in eleven instances. Considering the SSTs, the depth of enhancement is verified by the number of results statistically significant, which raised from 7 to 13 instances in favor of DM-GILS-RVND. Moreover, a statistically significant result in favor of GILS-RVND, which is reported in Table 5.2, was lost due to an improvement

made by DM-GILS-RVND.

Summary from Table A.2			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	4-32-20	11-10-35	0-13

Summary from Table 5.2			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	5-32-19	14-12-30	1-7

Table 5.10: Results on the 56-instances set selected from TSPLib

5.4.3.2 Experiments for Hamiltonian Path

Regarding the MLP version for Hamiltonian paths, all instance sets assessed in Section 5.3 were submitted to the comparison proposed in this subsection, except for the sets of 10, 20, and 50 customers. For these sets, all average solutions were equal to their respective best solutions, which are also optimal values for the tested instance as shown in Table 5.4.

Presented in Table 5.11, a summary of Table A.3, the results for the heterogeneous set of instances varying from 70 to 532 customers showed a notable enhancement of DM-GILS-RVND performance when the same computational time was given. In this comparison, the DM heuristic improved three average solutions of GILS-RVND, while lost in only one instance and tied in the six remaining ones.

Summary from Table A.3			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	1-9-0	1-6-3	0-0

Summary from Table 5.3			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	1-9-0	3-6-1	0-0

Table 5.11: Results for TSPLib instances selected in [39]

Compared to the results on the 100-customers set, displayed in Table 5.12 (a summary of the results reported in Table A.4), the DM strategy as the original strategy matched all BKS considering the aspect of best solution. In terms of average solution, one instance

result from Table 5.5 was improved by DM-GILS-RVND, resulting in 17 ties and three winnings for GILS-RVND.

Summary from Table A.4			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	0-20-0	3-17-0	0-0

Summary from Table 5.5			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	0-20-0	4-16-0	0-0

Table 5.12: Results on the 100-customers instances generated in [39]

For the 200-customers set, Table 5.13 reports the summarized results from Table A.5. For the best solution aspect, DM-GILS-RVND and GILS-RVND also reached all BKSs, as seen in the direct comparison (reported in Table 5.6). However, DM-GILS-RVND improved its performance against GILS-RVND regarding average solutions, obtaining 14 out of the 20 best average solutions, tying in three instances and losing in other three instances. The extended computational experiments for this instance set demonstrated that giving the same computational time for both heuristics, the DM strategy has been superior than GILS-RVND.

Summary from Table A.5			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	0-20-0	3-3-14	0-1

Summary from Table 5.6			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	0-20-0	10-1-9	0-1

Table 5.13: Results on the 200-customers instances generated in [39]

Considering the set of instances with 500 customers, Table 5.14 reports its results, which is a summary from Table A.6. In terms of average solution, the DM heuristic showed a predominant behavior compared to GILS-RVND, where 15 best average solutions reported in Table 5.7 were increased to 18 in these extended experiments. In this same aspect, no tie and two winnings for GILS-RVND were observed. For the best solution column, the numbers remained the same, where 15 winnings for DM-GILS-RVND, 4 ties and one winning for GILS-RVND were obtained.

The results for the set of the largest and hardest instances of this MLP variant are reported in Table 5.15, which summarizes Table A.7. In these experiments, considering best solution and average solution aspects, the DM strategy outperformed GILS-RVND in all 20 instances of this set, presenting a better performance than the original strategy. Moreover, in terms of statistical significance tests, the number of results statistically significant rose from 7 to 14, where DM-GILS-RVND performed better than GILS-RVND using the Students t-test.

Summary from Table A.6			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	1-4-15	2-0-18	0-4

Summary from Table 5.7			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	1-4-15	5-0-15	0-4

Table 5.14: Results on the 500-customers instances generated in [39]

Summary from Table A.7			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	0-0-20	0-0-20	0-14

Summary from Table 5.8			
DM-GILS-RVND			
	Best	Average	SST
GILS-RVND	2-0-18	5-0-15	0-7

Table 5.15: Results on the 1000-customers instances generated in [39]

Chapter 6

MDM-GILS-RVND: the Second Proposal

Unlike DM-GRASP (Data Mining GRASP) [35, 36], where the mining process is executed once and after a fixed number of multi-start iterations, the MDM-GRASP strategy (Multi Data Mining GRASP), firstly approached in [8], has its mining process executed whenever the elite set becomes stable – which refers to a number of iterations without any changes in the elite set. The idea of MDM-GRASP is to execute the mining process: (a) as soon as the elite set becomes stable and (b) whenever the elite set has been changed and again become stable. Thereby, in practical terms, the elite set quality is progressively enhanced, yielding, at each mining process, more refined patterns.

The second proposal of this work, named as MDM-GILS-RVND, is a hybrid version of GILS-RVND employing the idea of multi data mining. In this hybrid proposal, the data mining procedure has to be executed several times throughout the algorithm execution based on the stability of the elite set. However, this idea proposed so far cannot be directly applied into GILS-RVND, since only 10 multi-start iterations were defined for it in [42], which is an insufficient number of iterations to let the elite set becomes stable. It is worth saying that, on previous works, the MDM-GRASP was applied in heuristics that required 200, 500 and 1000 iterations, respectively reported in [8, 18, 32], showing promising results when compared with DM-GRASP.

Different from the classic MDM-GRASP, the proposed MDM-GILS-RVND has its mining points at: (a) after the first half of multi-start iterations, and (b) whenever the elite set is updated. As DM-GILS-RVND, MDM-GILS-RVND is also an algorithm composed of two main phases, a phase before the first data mining execution and a phase after this mining execution.

The details of MDM-GILS-RVND are displayed in Algorithm 6. Its first phase remains

the same as the first phase of DM-GILS-RVND (lines 1-19). Next, a boolean flag *Updated* is initialized and assigned to *TRUE* to enforce that the data mining process will be executed before the first multi-start iteration of the second phase (line 20). In the second phase itself (lines 20-45), if *D* (elite set) has been changed on the very last iteration, i.e., it was updated (*Updated* is *TRUE*), then, the mining process is performed again, and after that *Updated* is set to *FALSE* (lines 22-25). After, the hybrid constructive procedure is called, which is the constructive method described in Algorithm 5 (line 26). In the inner loop (lines 29-41), likewise the local search from the first phase of this algorithm, *D* can again be updated. If it happens, the mining process will be executed before the next multi-start iteration (*Updated* is assigned to *TRUE*, line 33), generating a list of patterns sorted in decreasing order by the number of mined arcs (line 23). Next, right after the local search phase, the best global solution s^* may be updated (lines 42-44). When all multi-start iterations are executed, the algorithm stops and s^* is returned (line 46).

Algorithm 6 MDM-GILS-RVND($I_{Max}, I_{ILS}, R, D, d, Sup, MaxP$)

```

1:  $f(s^*) \leftarrow \infty$ ;
2: for  $i = 1, \dots, I_{Max}/2$  do
3:    $s \leftarrow ConstructiveProcedure(\alpha \in R)$ ;
4:    $s' \leftarrow s$ ;
5:    $iterILS \leftarrow 0$ ;
6:   while  $iterILS < I_{ILS}$  do
7:      $s \leftarrow RVND(s)$ ;
8:      $UpdateEliteSet(D, d, s)$ ;
9:     if  $f(s) < f(s')$  then
10:       $s' \leftarrow s$ ;
11:       $iterILS \leftarrow 0$ ;
12:     end if
13:      $s \leftarrow Perturb(s')$ ;
14:      $iterILS \leftarrow iterILS + 1$ ;
15:   end while
16:   if  $f(s') < f(s^*)$  then
17:      $s^* \leftarrow s'$ ;
18:   end if
19: end for
20:  $Updated \leftarrow TRUE$ ;
21: for  $i = 1, \dots, I_{Max}/2$  do
22:   if  $Updated$  then
23:      $P \leftarrow MinePatterns(D, d, Sup, MaxP)$ ;
24:      $Updated \leftarrow FALSE$ ;
25:   end if
26:    $s \leftarrow HybridConstructiveProc(\alpha \in R, p \in P)$ ;
27:    $s' \leftarrow s$ ;
28:    $iterILS \leftarrow 0$ ;
29:   while  $iterILS < I_{ILS}$  do
30:      $s \leftarrow RVND(s)$ ;
31:      $UpdateEliteSet(D, d, s)$ ;
32:     if  $D$  was updated then
33:        $Updated \leftarrow TRUE$ ;
34:     end if
35:     if  $f(s) < f(s')$  then
36:        $s' \leftarrow s$ ;
37:        $iterILS \leftarrow 0$ ;
38:     end if
39:      $s \leftarrow Perturb(s')$ ;
40:      $iterILS \leftarrow iterILS + 1$ ;
41:   end while
42:   if  $f(s') < f(s^*)$  then
43:      $s^* \leftarrow s'$ ;
44:   end if
45: end for
46: return  $s^*$ ;

```

Chapter 7

Computational Evaluation for MDM-GILS-RVND

This chapter reports the computational experiments involving MDM-GILS-RVND along with DM-GILS-RVND and GILS-RVND results. Firstly, the methodology applied to carry out these experiments is shown in Section 7.1. Next, Sections 7.2 and 7.3 present the computational evaluation among MDM-GILS-RVND, DM-GILS-RVND and GILS-RVND considering the two MLP variants tackled in this work. At last, Section 7.4 covers extra analysis regarding these heuristics.

7.1 Comparison Methodology

The comparison methodology used for DM-GILS-RVND and GILS-RVND experiments, defined in Section 5.1, was also applied to MDM-GILS-RVND, including the same set of parameters, instances, used hardware and statistical significance tests (SSTs). It is noteworthy that MDM-GILS-RVND does not require any new parameter.

In Sections 7.2 and 7.3, we report the computational experiments in three tables: a table for solution quality, another one exclusively for computational time and a third one for summarizing the former two tables. Considering the first two tables, the same format used to compare the heuristics, specified in Section 5.1, was adopted. On the other hand, the third table reports all possible comparisons among MDM-GILS-RVND, DM-GILS-RVND and GILS-RVND, using the *A-B-C* format to arrange the results, defined in Subsection 5.4.3.

7.2 Experiments for Hamiltonian Circuit

For the MLP variant that considers Hamiltonian circuits as solutions, the two already introduced sets for this version were tested on MDM-GILS-RVND: a set of 23 instances selected in [1, 2] and a new set of 56 instances selected for this work. All instances of these sets are found in TSPLib.

Tables 7.1, 7.2, and 7.3 show the results on the 23-instance set selected in [1, 2] from TSPLib. For the tables of this chapter, since more than two algorithms are evaluated, the count Better refers to the number of best results that the respective strategy achieved. The obtained results regarding the best solution showed MDM-GILS-RVND found the optimal solutions for all instances along with the other heuristics. Regarding average solution, GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND obtained, respectively, 23, 21 and 20 best results. Considering computational time, the MDM-GILS-RVND and DM-GILS-RVND obtained in average, respectively, 5.48% and 5.06% of less computational time compared to GILS-RVND. Therefore, being MDM-GILS-RVND the fastest heuristic for this group.

Tables 7.4, 7.5 and 7.6 report the results on the 56-instances set. These results demonstrated MDM-GILS-RVND had the best performance among the heuristics in this group, achieving 36 best average solutions against 24 and 22, respectively, for DM-GILS-RVND and GILS-RVND. Regarding best solution, MDM-GILS-RVND also performed better than the other heuristics with 43 best results, while DM-GILS-RVND and GILS-RVND achieved, respectively, 39 and 36. In terms of running time, the MDM heuristic was better, attaining 43 best running times against 12 and 1, respectively, for DM-GILS-RVND and GILS-RVND. The general average of running time obtained by MDM-GILS-RVND was reduced in 23.36% compared to GILS-RVND, against 21.09% obtained by DM-GILS-RVND compared to GILS-RVND. In SST terms, ten instances, being eight instances using Student's t-test and two instances using Wilcoxon test, were statistically significant, where MDM-GILS-RVND was better than GILS-RVND. One instance was statistically significant when GILS-RVND was better than MDM-GILS-RVND in the Student's t-test. Additionally, where MDM-GILS-RVND was better than DM-GILS-RVND, two instances for Student's t-test and one instance for Wilcoxon test were statistically significant.

Instance	OPT/ BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND	
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution
dantzig42	12528	12528	12528.0	12528	12528.0	12528	12528.0
swiss42	22327	22327	22327.0	22327	22327.0	22327	22327.0
att48	209320	209320	209320.0	209320	209320.0	209320	209320.0
gr48	102378	102378	102378.0	102378	102378.0	102378	102378.0
hk48	247926	247926	247926.0	247926	247926.0	247926	247926.0
eil51	10178	10178	10178.0	10178	10178.0	10178	10184.3
berlin52	143721	143721	143721.0	143721	143721.0	143721	143721.0
brazil58	512361	512361	512361.0	512361	512361.0	512361	512361.0
st70	20557	20557	20557.0	20557	20557.0	20557	20557.0
eil76	17976	17976	17976.0	17976	17976.0	17976	17976.0
pr76	3445242	3445242	3445242.0	3445242	3445242.0	3445242	3445242.0
pr76r	345427	345427	345427.0	345427	345427.0	345427	345427.0
gr96	2097170	2097170	2097170.0	2097170	2097682.3	2097170	2097682.3
rat99	57986*	57986	57986.0	57986	57986.0	57986	57986.0
kroA100	983128	983128	983128.0	983128	983128.0	983128	983128.0
kroB100	986008	986008	986008.0	986008	986008.0	986008	986008.0
kroC100	961324	961324	961324.0	961324	961324.0	961324	961324.0
kroD100	976965	976965	976965.0	976965	976965.0	976965	976965.0
kroE100	971266	971266	971266.0	971266	971266.0	971266	971266.0
rd100	340047	340047	340047.0	340047	340047.0	340047	340047.0
eil101	27513*	27513	27513.0	27513	27519.8	27513	27519.8
lin105	603910	603910	603910.0	603910	603910.0	603910	603910.0
pr107	2026626	2026626	2026626.0	2026626	2026626.0	2026626	2026626.0
Better	-	23	23	23	21	23	20

* - Optimality is not proven

Table 7.1: Results on the instances selected from TSPLib in [1, 2]

7.3 Experiments for Hamiltonian Path

All 150 instances used in the experiments reported in Section 5.3 were also submitted to MDM-GILS-RVND heuristic, and their results are placed as follows.

Tables 7.7, 7.8 and 7.9 report the results on the 10-instances set selected by the authors of [39]. For this set, as seen in Table 7.9, both MDM-GILS-RVND and DM-GILS-RVND outperformed, in terms of computational time, GILS-RVND, whereas comparing DM and MDM heuristics to each other, they obtained nearly the same number of winnings. Furthermore, MDM-GILS-RVND had the best average computational time in this set requiring 14.47% less computational time than GILS-RVND. In terms of best solution, MDM-GILS-RVND and GILS-RVND found all 10 best results, while DM-GILS-RVND found 9. Considering average solution, MDM-GILS-RVND and GILS-RVND achieved 8 best results, whereas DM-GILS-RVND reached 6 best results.

For sets of 10-customers, 20-customers and 50-customers, their computational results are gathered in Table 7.10, which reports only best solutions, since all obtained average

Instance	GILS-RVND	DM-GILS-RVND		MDM-GILS-RVND	
	Average Time (s)	Average Time (s)	Gap(%) Time	Average Time (s)	Gap(%) Time
dantzig42	0.17	0.17	0.00	0.17	0.00
swiss42	0.16	0.16	0.00	0.16	0.00
att48	0.29	0.29	0.00	0.28	-3.45
gr48	0.31	0.29	-6.45	0.31	0.00
hk48	0.28	0.28	0.00	0.27	-3.57
eil51	0.40	0.40	0.00	0.39	-2.50
berlin52	0.39	0.39	0.00	0.39	0.00
brazil58	0.55	0.52	-5.45	0.52	-5.45
st70	0.99	0.94	-5.05	0.96	-3.03
eil76	1.52	1.44	-5.26	1.44	-5.26
pr76	1.35	1.33	-1.48	1.33	-1.48
pr76r	1.38	1.28	-7.24	1.30	-5.80
gr96	2.84	2.75	-3.17	2.79	-1.76
rat99	5.30	5.02	-5.28	4.87	-8.11
kroA100	4.21	3.58	-14.96	3.51	-16.63
kroB100	4.13	4.01	-2.91	3.77	-8.72
kroC100	3.95	3.62	-8.36	3.61	-8.61
kroD100	4.06	3.47	-14.53	3.57	-12.07
kroE100	4.00	3.69	-7.75	3.51	-12.25
rd100	4.15	3.99	-3.86	3.96	-4.58
eil101	5.79	5.00	-13.64	4.88	-15.72
lin105	3.57	3.44	-3.64	3.51	-1.68
pr107	4.33	4.01	-7.39	4.10	-5.31
Average	-	-	-5.06	-	-5.48
Better	3	13	-	16	-

Table 7.2: Computational time for TSPLib instances selected in [1, 2]

solutions were equal to their respective best solution. The results showed that all optimal values were found by the three heuristics in all executions with different seeds. The three strategies achieved the best solutions an average of 0.01, 0.02 and 0.05 seconds, respectively, for the sets of 10, 20 and 50 customers.

For the 100-customers set, Tables 7.11, 7.12 and 7.13 report its computational results. Regarding best solution, all heuristics found the BKSs of this set. On the other hand, for average solution, GILS-RVND achieved 20 best results, while DM-GILS-RVND and MDM-GILS-RVND achieved 16 best results each. In terms of computational time, DM-GILS-RVND became the best heuristic for this group, achieving 11 best results, while the MDM heuristic reached 9 best results, and, for GILS-RVND, no best result. Also in relation to computational time, DM-GILS-RVND achieved the best average for this set, requiring 10.54% less computational time when compared to GILS-RVND, whereas, for

	DM-GILS-RVND				MDM-GILS-RVND			
	Best	Average	Time	SST	Best	Average	Time	SST
GILS-RVND	0-23-0	2-21-0	0-6-17	0-0	0-23-0	3-20-0	0-4-19	0-0
DM-GILS-RVND	-	-	-	-	0-23-0	1-22-0	7-6-10	0-0

Table 7.3: Summary for Tables 7.1 and 7.2

Instance	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND	
	Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution
gr120	363454	363569.5	363454	363584.8	363454	363454.0
pr124	3154346	3154346.0	3154346	3154346.0	3154346	3154346.0
bier127	4545005	4546378.8	4545005	4545005.0	4545005	4545691.9
ch130	349874	349891.7	349874	349903.5	349874	349903.5
pr136	6199268	6199805.4	6199268	6200032.6	6199268	6200041.6
gr137	4061498	4061498.0	4061498	4061498.0	4061498	4061498.0
pr144	3846137	3846137.0	3846137	3846137.0	3846137	3846137.0
ch150	444424	444424.0	444424	444424.0	444424	444424.0
kroA150	1825769	1825769.0	1825769	1825769.0	1825769	1825769.0
kroB150	1786546	1786546.0	1786546	1786546.0	1786546	1786546.0
pr152	5064566	5064566.0	5064566	5064566.0	5064566	5064566.0
u159	2972030	2972204.2	2972030	2972291.3	2972030	2972204.2
si175	1808532	1808532.0	1808532	1808532.0	1808532	1808532.0
brg180	174750	174750.0	174750	174750.0	174750	174750.0
rat195	218632	218763.2	218632	218760.6	218632	218736.6
d198	1186049	1186098.6	1186049	1186086.2	1186049	1186273.3
kroA200	2672437	2672444.2	2672437	2672437.0	2672437	2672444.2
kroB200	2669515	2674486.0	2669515	2675761.6	2669515	2675993.6
gr202	2909247	2914644.2	2909247	2912564.8	2909247	2913368.4
ts225	13240046	13240046.0	13240046	13240533	13240046	13240046.0
tsp225	402783	403080.2	402783	402970.5	402783	402933.3
pr226	7196869	7196869.0	7196869	7196869.0	7196869	7196869.0
gr229	10725914	10729883.8	10725914	10729943.9	10725914	10731249.9
gil262	285060	285527.1	285043	285343.5	285060	285312.6
pr264	5471615	5471615.0	5471615	5471615.0	5471615	5471615.0
a280	346989	347125.9	346989	347009.6	346989	347106.9
pr299	6556628	6557983.4	6556628	6558164.9	6556628	6559030.8
lin318	5619810	5629995.9	5619810	5630556.9	5619810	5630590.5
rd400	2768830	2776672.7	2767608	2775101.2	2762532	2775707.0
fl417	1874242	1874242.8	1874242	1874242.0	1874242	1874242.0
gr431	21159702	21239150.9	21143311	21210280.6	21180562	21214270.9
pr439	17829541	17887107	17829541	17876876.9	17829541	17868632.7
pcb442	10301705	10323539.7	10290913	10321804.2	10301705	10321465.7
d493	6684190	6691057.1	6680997	6688669.0	6677458	6687268.2
att532	5613010	5632753.5	5622905	5630730.4	5617783	5628346.4
ali535	31870389	31904676.6	31870389	31902870.9	31860679	31910477.9
si535	12247211	12250679.7	12246397	12252151.6	12248066	12251841.0
pa561	658870	661211.6	660249	662216.9	660590	661790.6
u574	9314596	9344178.4	9313459	9350198.1	9308820	9333295.3
rat575	1848869	1859221.1	1847411	1856382.8	1847272	1856335.1
p654	7827273	7827639.2	7827273	7827919.4	7827273	7827867.8
d657	14159477	14220133.3	14125530	14188813.5	14112540	14195797.6
gr666	63571693	63731966.5	63546987	63663647.1	63500984	63612943.5
u724	13506660	13558605.3	13491605	13546178.5	13504408	13537514.7
rat783	3282794	3296069.6	3272226	3290521.7	3275858	3293606.1
dsj1000	7646018508	7685887300.0	7640607124	7671314634.0	7642715113	7664531851.0
dsj1000ceil	7646519008	7683329486.0	7644298506	7680652520.0	7646395679	7676973751.0
pr1002	115550770	116178260.2	115507699	115975798.5	115420846	115874237.0
si1032	46896355	46897662.4	46896355	46896783.6	46896355	46896783.6
u1060	102508056	102759766.0	102558414	102821622.2	102539819	102759493.6
vm1084	94760440	95053081.2	94705227	94982553.5	94670122	94960603.3
pcb1173	30926325	31032128.8	30891188	30972619.2	30890385	30957008.7
d1291	29383346	29477239.4	29392621	29511969.3	29389729	29515210.4
rl1304	144886001	145596878.7	144803181	145558912.3	144592447	145398549.2
rl1323	155697857	156360364.3	155749119	156332300.1	155719283	156273365.5
nrl1379	35360407	35519379.7	35327900	35475906.4	35291795	35456093.0
Better	36	22	39	24	43	36

Table 7.4: Results on the 56-instances set selected from TSPLib

Instance	GILS-RVND	DM-GILS-RVND		MDM-GILS-RVND	
	Average Time (s)	Average Time (s)	Gap(%) Time	Average Time (s)	Gap(%) Time
gr120	9.54	8.24	-13.63	8.10	-15.09
pr124	5.39	5.14	-4.64	5.15	-4.45
bier127	9.25	7.80	-15.68	7.73	-16.43
ch130	9.23	8.46	-8.34	8.88	-3.79
pr136	17.3	14.11	-18.44	14.82	-14.34
gr137	8.11	7.10	-12.45	7.16	-11.71
pr144	9.11	9.06	-0.55	8.80	-3.40
ch150	13.06	10.80	-17.30	10.67	-18.3
kroA150	19.84	15.51	-21.82	15.68	-20.97
kroB150	16.27	14.68	-9.77	14.49	-10.94
pr152	11.23	10.45	-6.95	10.20	-9.17
u159	14.21	12.88	-9.36	12.92	-9.08
si175	19.14	14.92	-22.05	14.85	-22.41
brg180	16.79	16.00	-4.71	16.18	-3.63
rat195	44.69	37.06	-17.07	35.57	-20.41
d198	38.28	31.55	-17.58	31.95	-16.54
kroA200	42.23	33.70	-20.20	33.73	-20.13
kroB200	42.00	36.48	-13.14	36.17	-13.88
gr202	35.95	31.62	-12.04	29.68	-17.44
ts225	26.60	27.28	2.56	27.10	1.88
tsp225	53.89	43.67	-18.96	43.43	-19.41
pr226	34.29	28.86	-15.84	28.85	-15.86
gr229	53.66	43.78	-18.41	41.12	-23.37
gil262	96.12	76.34	-20.58	74.72	-22.26
pr264	47.02	38.74	-17.61	38.68	-17.74
a280	107.18	83.61	-21.99	79.05	-26.25
pr299	104.92	78.64	-25.05	75.93	-27.63
lin318	117.98	100.63	-14.71	90.74	-23.09
rd400	350.84	278.61	-20.59	247.61	-29.42
fl417	382.64	263.61	-31.11	250.61	-34.51
gr431	336.98	264.24	-21.59	245.10	-27.27
pr439	285.56	199.04	-30.30	200.02	-29.96
pcb442	413.41	313.07	-24.27	291.64	-29.46
d493	608.47	410.33	-32.56	390.16	-35.88
att532	988.04	744.64	-24.63	760.49	-23.03
ali535	880.76	587.16	-33.33	570.79	-35.19
si535	498.76	340.83	-31.66	319.01	-36.04
pa561	1155.32	918.00	-20.54	873.20	-24.42
u574	1234.19	893.26	-27.62	854.63	-30.75
rat575	1739.46	1345.04	-22.67	1234.05	-29.06
p654	1755.28	1263.97	-27.99	1239.21	-29.40
d657	2615.66	1868.86	-28.55	1779.16	-31.98
gr666	2296.23	1699.61	-25.98	1609.15	-29.92
u724	4651.76	3505.29	-24.65	3132.70	-32.66
rat783	7044.52	4740.52	-32.71	4475.85	-36.46
dsj1000	18068.70	12612.84	-30.20	12233.54	-32.29
dsj1000ceil	18543.76	12885.48	-30.51	11929.94	-35.67
pr1002	11963.29	8817.55	-26.29	8029.58	-32.88
si1032	2402.72	1975.55	-17.78	1926.99	-19.80
u1060	15680.50	10471.80	-33.22	9910.02	-36.80
vm1084	13894.43	9673.12	-30.38	9468.85	-31.85
pcb1173	20508.89	13903.73	-32.21	14037.76	-31.55
d1291	12171.21	8189.00	-32.72	8072.84	-33.67
rl1304	18617.53	12967.80	-30.35	12407.04	-33.36
rl1323	22758.06	15938.27	-29.97	15115.63	-33.58
nrw1379	49624.72	34547.99	-30.38	32038.65	-35.44
Average	-	-	-21.09	-	-23.36
Better	1	12	-	43	-

Table 7.5: Computational time for the 56-instances set selected from TSPLib

	DM-GILS-RVND				MDM-GILS-RVND			
	Best	Average	Time	SST	Best	Average	Time	SST
GILS-RVND	5-32-19	15-11-30	1-0-55	1-5	7-33-16	12-14-30	1-0-55	1-10
DM-GILS-RVND	-	-	-	-	9-31-16	16-26-14	12-0-44	0-3

Table 7.6: Summary for Tables 7.4 and 7.5

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND	
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution
st70	19215	19215	19215.0	19215	19215.0	19215	19215.0
rat99	54984	54984	54984.0	54984	54984.0	54984	54984.0
kroD100	949594	949594	949594.0	949594	949594.0	949594	949594.0
lin105	585823	585823	585823.0	585823	585823.0	585823	585823.0
pr107	1980767	1980767	1980767.0	1980767	1980767.0	1980767	1980767.0
rat195	210191	210191	210335.9	210191	210385.6	210191	210388.2
pr226	7100308	7100308	7100308.0	7100308	7100308.0	7100308	7100308.0
lin318	5560679	5560679	5569819.5	5560679	5570488.8	5560679	5568258.8
pr439	17688561	17688561	17734922.0	17702120	17735797.8	17688561	17737442.9
att532	5577965	5581240	5597866.8	5581240	5596611.1	5581240	5594679.7
Better	-	10	8	9	6	10	8

Table 7.7: Results for TSPLib instances selected in [39]

Instance	GILS-RVND	DM-GILS-RVND		MDM-GILS-RVND	
	Average Time (s)	Average Time (s)	Gap(%) Time	Average Time (s)	Gap(%) Time
st70	0.94	0.91	-3.19	0.91	-3.19
rat99	5.81	5.22	-10.15	5.17	-11.02
kroD100	4.21	3.75	-10.93	3.74	-11.16
lin105	3.79	3.63	-4.22	3.72	-1.85
pr107	4.96	4.50	-9.27	4.54	-8.47
rat195	44.99	37.84	-15.89	33.78	-24.92
pr226	34.35	30.10	-12.37	30.45	-11.35
lin318	124.15	97.39	-21.55	101.50	-18.24
pr439	275.12	225.26	-18.12	201.54	-26.74
att532	923.03	710.32	-23.04	667.12	-27.72
Average	-	-	-12.87	-	-14.47
Better	0	5	-	6	-

Table 7.8: Computational time for TSPLib instances selected in [39]

	DM-GILS-RVND				MDM-GILS-RVND			
	Best	Average	Time	SST	Best	Average	Time	SST
GILS-RVND	1-9-0	3-6-1	0-0-10	0-0	0-10-0	2-6-2	0-0-10	0-0
DM-GILS-RVND	-	-	-	-	0-9-1	2-6-2	4-1-5	0-0

Table 7.9: Summary for Tables 7.7 and 7.8

MDM-GILS-RVND, the reduction was 10.41%.

Tables 7.14, 7.15 and 7.16 report the results on the 200-customers set. Regarding the results of best solutions shown in Table 7.14, all heuristics found the BKSs of this set. Considering average solutions, GILS-RVND, DM-GILS-RVND, and MDM-GILS-RVND achieved, respectively, 9, 8 and 7 best results, which, in this case, they presented similar behaviors. In relation to computational time, MDM-GILS-RVND attained 18 best running times, against 2 best running times obtained by DM-GILS-RVND. MDM-

Instance	GILS-RVND			DM-GILS-RVND			MDM-GILS-RVND		
	S10	S20	S50	S10	S20	S50	S10	S20	S50
TRP-Sn-R1	1303	3175	12198	1303	3175	12198	1303	3175	12198
TRP-Sn-R2	1517	3248	11621	1517	3248	11621	1517	3248	11621
TRP-Sn-R3	1233	3570	12139	1233	3570	12139	1233	3570	12139
TRP-Sn-R4	1386	2983	13071	1386	2983	13071	1386	2983	13071
TRP-Sn-R5	978	3248	12126	978	3248	12126	978	3248	12126
TRP-Sn-R6	1477	3328	12684	1477	3328	12684	1477	3328	12684
TRP-Sn-R7	1163	2809	11176	1163	2809	11176	1163	2809	11176
TRP-Sn-R8	1234	3461	12910	1234	3461	12910	1234	3461	12910
TRP-Sn-R9	1402	3475	13149	1402	3475	13149	1402	3475	13149
TRP-Sn-R10	1388	3359	12892	1388	3359	12892	1388	3359	12892
TRP-Sn-R11	1405	2916	12103	1405	2916	12103	1405	2916	12103
TRP-Sn-R12	1150	3314	10633	1150	3314	10633	1150	3314	10633
TRP-Sn-R13	1531	3412	12115	1531	3412	12115	1531	3412	12115
TRP-Sn-R14	1219	3297	13117	1219	3297	13117	1219	3297	13117
TRP-Sn-R15	1087	2862	11986	1087	2862	11986	1087	2862	11986
TRP-Sn-R16	1264	3433	12138	1264	3433	12138	1264	3433	12138
TRP-Sn-R17	1058	2913	12176	1058	2913	12176	1058	2913	12176
TRP-Sn-R18	1083	3124	13357	1083	3124	13357	1083	3124	13357
TRP-Sn-R19	1394	3299	11430	1394	3299	11430	1394	3299	11430
TRP-Sn-R20	951	2796	11935	951	2796	11935	951	2796	11935
Better	20	20	20	20	20	20	20	20	20

Table 7.10: Results for instances generated in [39] considering 10, 20 and 50 customers

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND	
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution
TRP-S100-R1	32779	32779	32779.0	32779	32779.0	32779	32779.0
TRP-S100-R2	33435	33435	33435.0	33435	33435.0	33435	33435.0
TRP-S100-R3	32390	32390	32390.0	32390	32390.0	32390	32390.0
TRP-S100-R4	34733	34733	34733.0	34733	34733.0	34733	34733.0
TRP-S100-R5	32598	32598	32598.0	32598	32598.0	32598	32598.0
TRP-S100-R6	34159	34159	34159.0	34159	34159.0	34159	34159.0
TRP-S100-R7	33375	33375	33375.0	33375	33375.0	33375	33375.0
TRP-S100-R8	31780	31780	31780.0	31780	31780.0	31780	31780.0
TRP-S100-R9	34167	34167	34167.0	34167	34167.5	34167	34167.5
TRP-S100-R10	31605	31605	31605.0	31605	31605.0	31605	31605.0
TRP-S100-R11	34188	34188	34198.5	34188	34219.5	34188	34230.0
TRP-S100-R12	32146	32146	32146.0	32146	32146.0	32146	32146.0
TRP-S100-R13	32604	32604	32604.0	32604	32604.0	32604	32604.0
TRP-S100-R14	32433	32433	32433.0	32433	32433.5	32433	32433.5
TRP-S100-R15	32574	32574	32574.0	32574	32574.0	32574	32574.0
TRP-S100-R16	33566	33566	33566.0	33566	33566.0	33566	33566.0
TRP-S100-R17	34198	34198	34198.0	34198	34198.0	34198	34198.0
TRP-S100-R18	31929	31929	31929.0	31929	31929.0	31929	31929.0
TRP-S100-R19	33463	33463	33463.0	33463	33463.0	33463	33463.0
TRP-S100-R20	33632	33632	33632.2	33632	33632.4	33632	33632.5
Better		20	20	20	16	20	16

Table 7.11: Results on the 100-customers set generated in [39]

Instance	GILS-RVND	DM-GILS-RVND		MDM-GILS-RVND	
	Average Time (s)	Average Time (s)	Gap(%) Time	Average Time (s)	Gap(%) Time
TRP-S100-R1	4.33	3.75	-13.39	3.80	-12.24
TRP-S100-R2	4.61	4.20	-8.89	4.17	-9.54
TRP-S100-R3	4.33	4.03	-6.93	4.02	-7.16
TRP-S100-R4	4.45	3.89	-12.58	3.75	-15.73
TRP-S100-R5	5.43	4.48	-17.50	4.50	-17.13
TRP-S100-R6	4.83	4.31	-10.77	4.22	-12.63
TRP-S100-R7	5.34	4.31	-19.29	4.34	-18.73
TRP-S100-R8	4.29	3.57	-16.78	3.64	-15.15
TRP-S100-R9	4.56	4.18	-8.33	4.27	-6.36
TRP-S100-R10	4.13	3.67	-11.14	3.69	-10.65
TRP-S100-R11	4.72	4.48	-5.08	4.41	-6.57
TRP-S100-R12	4.38	4.03	-7.99	4.15	-5.25
TRP-S100-R13	4.75	4.41	-7.16	4.31	-9.26
TRP-S100-R14	3.89	3.54	-9.00	3.70	-4.88
TRP-S100-R15	4.36	3.90	-10.55	3.91	-10.32
TRP-S100-R16	4.87	4.39	-9.86	4.50	-7.60
TRP-S100-R17	5.65	5.41	-4.25	5.25	-7.08
TRP-S100-R18	4.34	3.95	-8.99	3.94	-9.22
TRP-S100-R19	4.91	4.35	-11.41	4.39	-10.59
TRP-S100-R20	5.35	4.77	-10.84	4.70	-12.15
Average	-	-	-10.54	-	-10.41
Better	0	11	-	9	-

Table 7.12: Computational time for the 100-customers set generated in [39]

	DM-GILS-RVND				MDM-GILS-RVND			
	Best	Average	Time	SST	Best	Average	Time	SST
GILS-RVND	0-20-0	3-17-0	0-0-20	0-0	0-20-0	3-17-0	0-0-20	0-0
DM-GILS-RVND	-	-	-	-	0-20-0	1-19-0	11-0-9	0-0

Table 7.13: Summary for Tables 7.11 and 7.12

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND	
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution
TRP-S200-R1	88787	88787	88794.6	88787	88812.1	88787	88798.4
TRP-S200-R2	91977	91977	92013.1	91977	92009.8	91977	92042.7
TRP-S200-R3	92568	92568	92631.2	92568	92612.1	92568	92648.8
TRP-S200-R4	93174	93174	93192.3	93174	93180.4	93174	93186.8
TRP-S200-R5	88737	88737	88841.2	88737	88861.9	88737	88836.8
TRP-S200-R6	91589	91589	91601.9	91589	91596.3	91589	91596.3
TRP-S200-R7	92754	92754	92763.2	92754	92777.6	92754	92763.0
TRP-S200-R8	89048	89048	89049.0	89048	89051.0	89048	89051.0
TRP-S200-R9	86326	86326	86326.0	86326	86326.0	86326	86326.0
TRP-S200-R10	91552	91552	91596.5	91552	91602.6	91552	91627.5
TRP-S200-R11	92655	92655	92700.6	92655	92702.5	92655	92706.8
TRP-S200-R12	91457	91457	91504.1	91457	91521.7	91457	91507.0
TRP-S200-R13	86155	86155	86181.4	86155	86184.7	86155	86188.6
TRP-S200-R14	91882	91882	91929.1	91882	91892.7	91882	91903.4
TRP-S200-R15	88912	88912	88912.4	88912	88912.0	88912	88912.0
TRP-S200-R16	89311	89311	89364.7	89311	89316.6	89311	89311.0
TRP-S200-R17	89089	89089	89118.3	89089	89097.2	89089	89092.0
TRP-S200-R18	93619	93619	93676.6	93619	93641.5	93619	93650.9
TRP-S200-R19	93369	93369	93401.6	93369	93504.1	93369	93441.0
TRP-S200-R20	86292	86292	86292.0	86292	86296.9	86292	86296.9
Better	-	20	9	20	8	20	7

Table 7.14: Results for the 200-customers set generated in [39]

GILS-RVND reduced in 21.18% the average of computational time when compared to GILS-RVND. For DM-GILS-RVND, the reduction was 18.19%. Furthermore, only one instance, where MDM-GILS-RVND performed better than GILS-RVND, was statistically significant using Wilcoxon test.

The computational results on the 500-customers set are reported in Tables 7.17, 7.18 and 7.19. Regarding best solution, MDM-GILS-RVND obtained the utmost performance in this set with 16 best results, while, for GILS-RVND and DM-GILS-RVND, respectively, 5 and 8 best results were attained. In terms of average solution, the MDM heuristic also obtained the best results of this set, since it achieved 13 best results, while, GILS-RVND and DM-GILS-RVND got 1 and 6 best results, respectively. Furthermore, MDM-GILS-RVND achieved the best results in terms of computational time for this set, since it achieved 15 best results, while DM-GILS-RVND and GILS-RVND, respectively, reached 5 and 0 best results. The average of running time obtained by MDM-GILS-RVND was reduced in 27.74% compared to GILS-RVND, whereas, for DM-GILS-RVND, the reduction was 25.55%. Five instances, where MDM-GILS-RVND was better than GILS-RVND, were statistically significant for Student's t-test. Another five instances, where MDM-GILS-RVND was better than DM-GILS-RVND, were also statistically significant, being

Instance	GILS-RVND	DM-GILS-RVND		MDM-GILS-RVND	
	Average Time (s)	Average Time (s)	Gap(%) Time	Average Time (s)	Gap(%) Time
TRP-S200-R1	44.08	33.10	-24.91	31.78	-27.90
TRP-S200-R2	41.02	35.24	-14.09	34.07	-16.94
TRP-S200-R3	40.50	32.60	-19.51	31.71	-21.70
TRP-S200-R4	43.48	34.91	-19.71	32.48	-25.30
TRP-S200-R5	42.65	36.31	-14.87	35.83	-15.99
TRP-S200-R6	42.43	34.61	-18.43	33.57	-20.88
TRP-S200-R7	43.75	34.91	-20.21	33.08	-24.39
TRP-S200-R8	45.13	39.14	-13.27	36.04	-20.14
TRP-S200-R9	39.51	32.44	-17.89	30.56	-22.65
TRP-S200-R10	44.63	35.84	-19.70	32.97	-26.13
TRP-S200-R11	43.90	39.15	-10.82	37.76	-13.99
TRP-S200-R12	46.08	38.64	-16.15	36.87	-19.99
TRP-S200-R13	43.88	35.59	-18.89	36.03	-17.89
TRP-S200-R14	42.60	35.74	-16.10	34.01	-20.16
TRP-S200-R15	42.28	36.58	-13.48	35.23	-16.67
TRP-S200-R16	47.08	36.46	-22.56	36.19	-23.13
TRP-S200-R17	42.92	35.54	-17.19	32.90	-23.35
TRP-S200-R18	46.36	37.09	-20.00	37.04	-20.10
TRP-S200-R19	42.75	34.48	-19.35	34.98	-18.18
TRP-S200-R20	42.42	31.14	-26.59	30.53	-28.03
Average	-	-	-18.19	-	-21.18
Better	0	2	-	18	-

Table 7.15: Computational time for the 200-customers set generated in [39]

	DM-GILS-RVND				MDM-GILS-RVND			
	Best	Average	Time	SST	Best	Average	Time	SST
GILS-RVND	0-20-0	10-1-9	0-0-20	0-1	0-20-0	10-1-9	0-0-20	0-1
DM-GILS-RVND	-	-	-	-	0-20-0	8-4-8	2-0-18	0-0

Table 7.16: Summary for Tables 7.14 and 7.15

four instances for Student's t-test and one for Wilcoxon test.

Tables 7.20, 7.21 and 7.22 present the results on the 1000-customers set, the hardest instance set of this group. Regarding best solution, MDM-GILS-RVND obtained 15 best results for this set, while, DM-GILS-RVND and GILS-RVND reached 4 and 1 best results, respectively. For average solution, MDM-GILS-RVND presented the best performance for this set, since it achieved 19 out of 20 best results, while DM-GILS-RVND and GILS-RVND, respectively, attained only 1 and 0 best result. Shown in Table 7.21, the computational times achieved by MDM-GILS-RVND improved 17 results, while DM-GILS-RVND and GILS-RVND, respectively, achieved 3 and 0 best results. Considering the average of computational time, MDM-GILS-RVND again reached the best reduction

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND	
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution
TRP-S500-R1	1841386	1841386	1856018.7	1841386	1852238.0	1841386	1850046.4
TRP-S500-R2	1815664 [†]	1816568	1823196.9	1817288	1821667.9	1817057	1822540.1
TRP-S500-R3	1826855 [†]	1833044	1839254.2	1831357	1838557.2	1827550	1837315.8
TRP-S500-R4	1804894 [†]	1809266	1815876.4	1808563	1816765.0	1804005	1813295.1
TRP-S500-R5	1821250 [†]	1823975	1834031.7	1823135	1831575.2	1823135	1832073.0
TRP-S500-R6	1782731 [†]	1786620	1790912.4	1785217	1789675.6	1784189	1789923.1
TRP-S500-R7	1847999	1847999	1857926.6	1847089	1854324.8	1846753	1853596.3
TRP-S500-R8	1819636 [†]	1820846	1829257.3	1820639	1829831.8	1820421	1828768.2
TRP-S500-R9	1733819	1733819	1737024.9	1730561	1736278.3	1731594	1736149.5
TRP-S500-R10	1761174 [†]	1762741	1767366.3	1762197	1767912.7	1761824	1766078.2
TRP-S500-R11	1797881	1797881	1801467.9	1797881	1802957.9	1797881	1802827.0
TRP-S500-R12	1774452	1774452	1783847.1	1774452	1781588.2	1774452	1783638.0
TRP-S500-R13	1863905 [†]	1873699	1878049.4	1867803	1875973.1	1867156	1875908.3
TRP-S500-R14	1799171	1799171	1805732.9	1798130	1803379.2	1796425	1802431.8
TRP-S500-R15	1785263 [†]	1791145	1797532.9	1790524	1795858.3	1785155	1793916.3
TRP-S500-R16	1804392 [†]	1810188	1816484.0	1808183	1814941.6	1808775	1817049.6
TRP-S500-R17	1825748	1825748	1834443.2	1824674	1832635.8	1821971	1830454.8
TRP-S500-R18	1825615 [†]	1826263	1833323.7	1826263	1831998.5	1826263	1831295.2
TRP-S500-R19	1776855 [†]	1779248	1782763.9	1774846	1785575.3	1775023	1781604.6
TRP-S500-R20	1820813	1820813	1830483.3	1820713	1829740.0	1820168	1830222.6
Better	-	5	1	8	6	16	13

[†] cost values from [37]

Table 7.17: Results for the 500-customers set generated in [39]

of computational time in the set, requiring 32.92% less computational time compared to GILS-RVND, while the reduction for DM-GILS-RVND was 31.23%. Regarding SSTs, twelve instances, where MDM-GILS-RVND was better than GILS-RVND, were statistically significant for the Student's t-test. Five instances, where MDM-GILS-RVND was better than DM-GILS-RVND, were also statistically significant, being four instances for Student's t-test and one for Wilcoxon test.

Considering all computational experiments reported in Sections 7.2 and 7.3, MDM-GILS-RVND and DM-GILS-RVND outperformed GILS-RVND not only in terms of computational time, but also in relation to solution quality, mainly in challenging instances, usually with 500 customers or more. We also observe in Tables 7.19 and 7.22 that MDM-GILS-RVND surpassed DM-GILS-RVND, respectively, in instances of 500 customers and 1000 customers, reinforcing that mining the elite set more than once increased the quality of the results.

7.4 Complementary Analyses

In order to better understand the behavior of MDM-GILS-RVND, this section presents three distinct complementary assessments. The first one, presented in Subsection 7.4.1,

Instance	GILS-RVND	DM-GILS-RVND		MDM-GILS-RVND	
	Average Time (s)	Average Time (s)	Gap(%) Time	Average Time (s)	Gap(%) Time
TRP-S500-R1	830.85	620.18	-25.36	623.37	-24.97
TRP-S500-R2	724.17	502.99	-30.54	485.98	-32.89
TRP-S500-R3	761.86	580.34	-23.83	571.46	-24.99
TRP-S500-R4	810.63	589.33	-27.30	579.34	-28.53
TRP-S500-R5	734.32	565.36	-23.01	512.22	-30.25
TRP-S500-R6	796.28	596.65	-25.07	612.53	-23.08
TRP-S500-R7	781.01	609.98	-21.90	620.16	-20.60
TRP-S500-R8	769.33	580.50	-24.54	574.23	-25.36
TRP-S500-R9	693.82	545.16	-21.43	530.72	-23.51
TRP-S500-R10	784.64	587.06	-25.18	564.93	-28.00
TRP-S500-R11	741.50	536.77	-27.61	513.56	-30.74
TRP-S500-R12	766.23	564.99	-26.26	523.90	-31.63
TRP-S500-R13	797.54	556.95	-30.17	540.20	-32.27
TRP-S500-R14	835.86	585.61	-29.94	565.17	-32.38
TRP-S500-R15	800.69	587.39	-26.64	556.92	-30.44
TRP-S500-R16	761.93	590.35	-22.52	555.91	-27.04
TRP-S500-R17	738.57	533.61	-27.75	548.44	-25.74
TRP-S500-R18	780.31	636.26	-18.46	581.67	-25.46
TRP-S500-R19	773.39	553.04	-28.49	559.35	-27.68
TRP-S500-R20	726.50	544.20	-25.09	514.63	-29.16
Average	-	-	-25.55	-	-27.74
Better	0	5	-	15	-

Table 7.18: Computational time for the 500-customers set generated in [39]

	DM-GILS-RVND				MDM-GILS-RVND			
	Best	Average	Time	SST	Best	Average	Time	SST
GILS-RVND	3-2-15	5-0-15	0-0-20	0-4	1-4-15	2-0-18	0-0-20	0-5
DM-GILS-RVND	-	-	-	-	3-4-13	6-0-14	5-0-15	0-5

Table 7.19: Summary for Tables 7.17 and 7.18

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND	
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution
TRP-S1000-R1	5107395	5107395	5133698.3	5099593	5122436.1	5097334	5117118.1
TRP-S1000-R2	5106161	5106161	5127449.4	5084762	5112406.6	5083772	5106199.8
TRP-S1000-R3	5096977	5096977	5113302.9	5089701	5111390.5	5087014	5108822.4
TRP-S1000-R4	5112465 [†]	5118006	5141392.6	5110184	5142219.9	5110766	5137965.2
TRP-S1000-R5	5097991 [†]	5103894	5122660.7	5085136	5116167.7	5088230	5112437.5
TRP-S1000-R6	5109946 [†]	5115816	5143087.1	5102671	5132201.1	5098630	5124817.1
TRP-S1000-R7	4995703 [†]	5021383	5032722.0	4984950	5014796.3	4982461	5019363.8
TRP-S1000-R8	5109325	5109325	5132722.6	5109325	5129942.9	5104125	5127143.8
TRP-S1000-R9	5046566 [†]	5052599	5073245.3	5045262	5070469.7	5044687	5069371.7
TRP-S1000-R10	5060019 [†]	5078191	5093592.6	5070109	5089275.6	5062172	5079835.2
TRP-S1000-R11	5031455 [†]	5041913	5066161.5	5052459	5066612.8	5051685	5061519.8
TRP-S1000-R12	5029792	5029792	5051235.2	5030837	5043866.1	5023000	5038120.8
TRP-S1000-R13	5102520	5102520	5131437.5	5098034	5123032.4	5085356	5114870.4
TRP-S1000-R14	5092861 [†]	5099433	5118980.6	5089565	5116989.7	5087794	5111368.4
TRP-S1000-R15	5131013 [†]	5142470	5174493.2	5123240	5166046.9	5134114	5165423.3
TRP-S1000-R16	5064094 [†]	5073972	5090280.5	5070422	5091420.9	5047856	5088822.7
TRP-S1000-R17	5052283 [†]	5071485	5084450.4	5065952	5082717.2	5063904	5078517.9
TRP-S1000-R18	5005789 [†]	5017589	5037094.0	5003047	5038269.6	4998254	5033070.8
TRP-S1000-R19	5064873 [†]	5076800	5097167.6	5065743	5087261.2	5065623	5086618.9
TRP-S1000-R20	4977262	4977262	5002920.6	4970735	5003599.3	4976158	5000650.3
Better	-	1	0	4	1	15	19

[†] cost values from [37]

Table 7.20: Results on the 1000-customers set generated in [39]

Instance	GILS-RVND	DM-GILS-RVND		MDM-GILS-RVND	
	Average Time (s)	Average Time (s)	Gap(%) Time	Average Time (s)	Gap(%) Time
TRP-S1000-R1	19889.15	14123.42	-28.99	13747.69	-30.88
TRP-S1000-R2	19218.18	13532.12	-29.59	13626.35	-29.10
TRP-S1000-R3	18798.18	12553.89	-33.22	11772.82	-37.37
TRP-S1000-R4	18493.11	12404.32	-32.92	12985.91	-29.78
TRP-S1000-R5	19143.87	13862.07	-27.59	13663.97	-28.62
TRP-S1000-R7	17681.02	13154.44	-25.60	11873.02	-32.85
TRP-S1000-R8	18065.24	12781.96	-29.25	12152.55	-32.73
TRP-S1000-R9	17979.62	11758.44	-34.60	11744.68	-34.68
TRP-S1000-R10	17596.33	12274.40	-30.24	12641.17	-28.16
TRP-S1000-R11	18307.69	11826.31	-35.40	11665.22	-36.28
TRP-S1000-R12	19149.54	12940.67	-32.42	12801.08	-33.15
TRP-S1000-R13	19604.19	13895.03	-29.12	13484.77	-31.21
TRP-S1000-R14	18974.58	13171.29	-30.58	12877.75	-32.13
TRP-S1000-R15	18889.53	13687.70	-27.54	12423.75	-34.23
TRP-S1000-R16	18206.27	11962.50	-34.29	11437.22	-37.18
TRP-S1000-R17	18571.62	11847.47	-36.21	11728.89	-36.85
TRP-S1000-R18	19745.37	12966.44	-34.33	12914.31	-34.60
TRP-S1000-R19	19790.69	13365.27	-32.47	12793.33	-35.36
TRP-S1000-R20	18715.65	13150.60	-29.73	13099.58	-30.01
Average	-	-	-31.23	-	-32.92
Better	0	3	-	17	-

Table 7.21: Computational time for the 1000-customers set generated in [39]

	DM-GILS-RVND				MDM-GILS-RVND			
	Best	Average	Time	SST	Best	Average	Time	SST
GILS-RVND	3-2-15	5-0-15	0-0-20	0-9	1-4-15	2-0-18	0-0-20	0-12
DM-GILS-RVND	-	-	-	-	3-4-13	6-0-14	5-0-15	0-5

Table 7.22: Summary for Tables 7.20 and 7.21

illustrates the impact of the usage of patterns into initial solutions compared to initial solutions built by the constructive method of GILS-RVND. In the next analysis, Subsection 7.4.2 shows experiments of time convergence of GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND to targets (cost values of solutions). Subsection 7.4.3 presents fair comparisons among MDM-GILS-RVND, DM-GILS-RVND and GILS-RVND, where the same amount of time is given to both heuristics as stopping criterion instead of the number of multi-start iterations, as done in Sections 7.2 and 7.3.

7.4.1 Impact of the Usage of Mined Patterns

In this subsection, an execution using the TRP-S1000-R1 instance and another one with TRP-S1000-R7 instance were selected to demonstrate the impact of the usage of mined patterns into initial solutions, as done in Subsection 5.4.1. For each experimented instance, two figures show the results of each heuristic phase: the constructive phase and the local search phase, where the multi-start iterations were arranged at the abscissa axis and cost values of solutions at the ordinate axis. For simplification, as MDM-GILS-RVND, DM-GILS-RVND and GILS-RVND have the same behavior in the first half of the multi-start iterations, one line is used to represent these heuristics.

Shown in Figure 7.1, the first experiment considers the TRP-S1000-R1 instance. As can be observed after the DM process, the constructive and local search phases show close lines for the two data mining strategies, demonstrating, in this way, very similar behaviors. On the other hand, these two data mining heuristics present better cost values of solutions when compared to those obtained by GILS-RVND, indicating that the data mining heuristics were able to explore more efficiently the search space of the problem. The second data mining execution by MDM-GILS-RVND is showed as DM* in both graphs. At last, the better cost value obtained among the evaluated heuristics was found by MDM-GILS-RVND at the 10th multi-start iteration.

Figure 7.2 displays the behaviors of the heuristics on TRP-S1000-R7 instance. The data mining heuristics presented better cost values throughout the algorithm execution when compared to GILS-RVND. This figure shows that the best solution found was ob-

tained by MDM-GILS-RVND after the second mining process, which happens again after the 9th multi-start iteration.

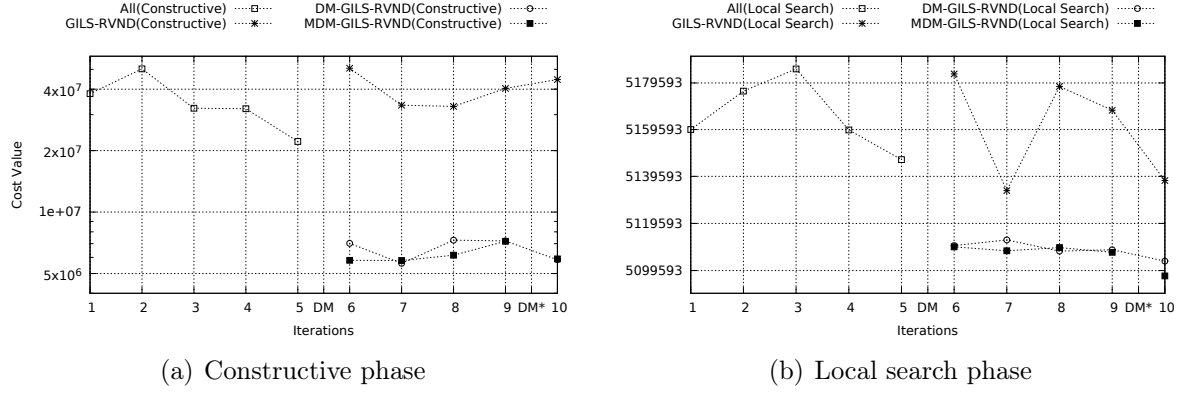


Figure 7.1: Cost values versus Iteration - TRP-S1000-R1 instance

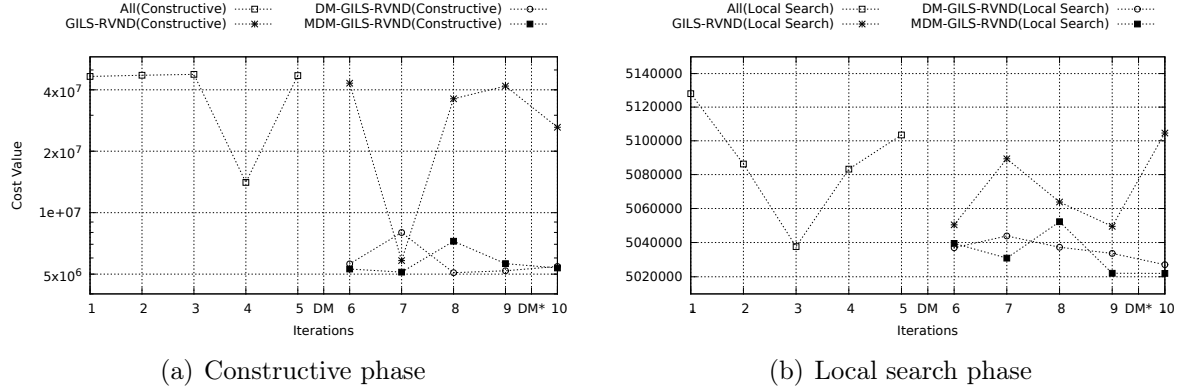


Figure 7.2: Cost values versus Iteration - TRP-S1000-R7 instance

7.4.2 Analyses of Time Convergence

Following the time convergence analysis stated in Subsection 5.4.2, which used kroA200 and pr299 instances, the current subsection aims at to present new plots including the computational results of MDM-GILS-RVND.

Figure 7.3 considers the results of TTT plots on the kroA200 instance. Taking the easy target (2677290) into account, shown in Figure 7.3(a), all heuristics present very similar behaviors, which are indicated by using the overlapped points. On the other hand, the plot used to illustrate the hard target (2672445), shown in Figure 7.3(b), demonstrated distinct performances among the data mining heuristics and the original approach. For example, both MDM-GILS-RVND and DM-GILS-RVND presented a probability of near 90% to reach the given target in 25 seconds, while, within the same computational time, GILS-RVND presents a probability of around 65% to reach the same target.

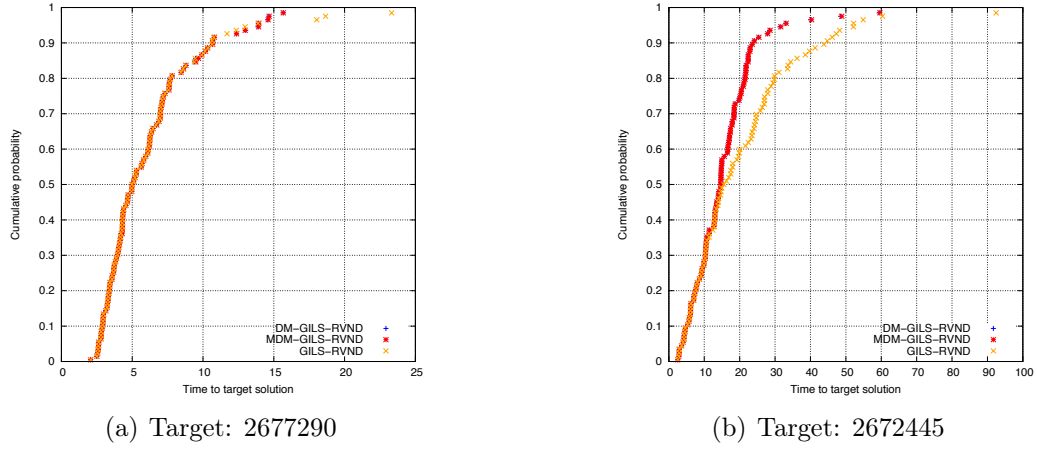


Figure 7.3: TTT plots for kroA200 instance

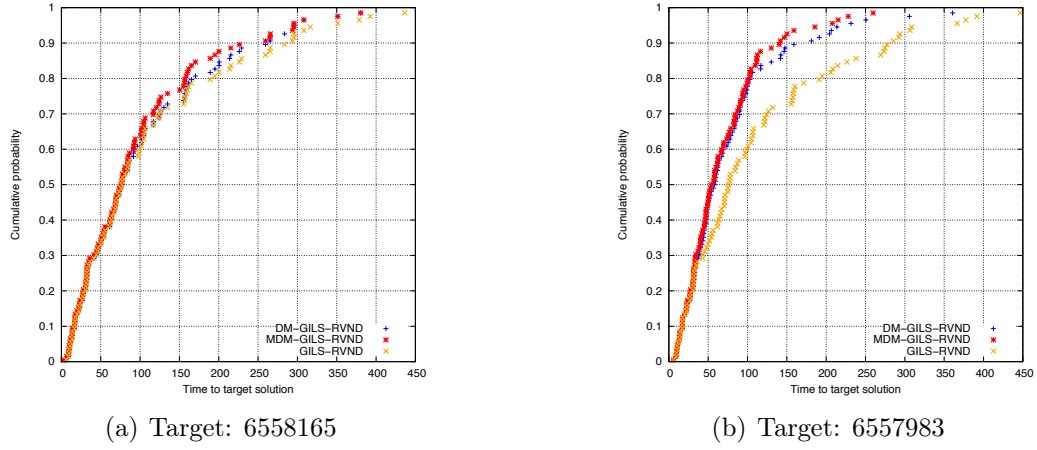


Figure 7.4: TTT plots for pr299 instance

TTT plots defined for pr299 instance are displayed in Figure 7.4. Specifically for Figure 7.4(a), which represents the easy target (6558165), the three evaluated heuristics present marginally different behaviors among each other. For example, to reach the given target within about 160 seconds, GILS-RVND, DM-GILS-RVND, and MDM-GILS-RVND obtained, respectively, probabilities of near 80%, 81%, and 85%. Alternatively, Figure 7.4(b) present the behaviors obtained by the heuristics for the hard target (6557983). For example, GILS-RVND, DM-GILS-RVND, and MDM-GILS-RVND have probabilities of, respectively, 72%, 89%, and 92% to achieve the target solution within 150 seconds.

7.4.3 Complementary Experiments

This subsection reports the computational experiments that involves MDM-GILS-RVND using running time as stopping criterion. The reported results of this subsection includes MDM-GILS-RVND results along with the results reported in Subsection 5.4.3. All

instance sets follow the *A-B-C* format, firstly presented in Subsection 5.4.3, and are divided into Subsections 7.4.3.1 and 7.4.3.2, showing, respectively, the results for the MLP versions of Hamiltonian circuits and Hamiltonian paths. Finally, since this subsection presents the last computational experiments of this work, the number of new BKSs found in each instance set is also reported, whereas the details of each new BKS are reported in Table A.3 (see Appendix A).

7.4.3.1 Experiments for Hamiltonian Circuit

This subsection considers the Hamiltonian circuit version of MLP, where a set of 23 instances selected in [39] and a set of 56 instances selected from TSPLib were submitted to the proposed MDM-GILS-RVND extended experiments, which are reported as follows.

Table 7.23 (union of Tables A.1 and 7.3) reports the results on the 23-instances set, which regards only the instances that do not have their optimal values proven, specifically rat99 and eil101 instances. Considering equal-time comparisons, MDM-GILS-RVND achieved the average solutions found by GILS-RVND, which held the best average solutions of this set. Therefore, giving the same running time to the data mining heuristics, they could match all best computational results of this set.

Summary from Table A.1

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	0-2-0	0-2-0	0-0	0-2-0	0-2-0	0-0
DM-GILS-RVND	-	-	-	0-2-0	0-2-0	0-0

Summary from Table 7.3

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	0-2-0	2-0-0	0-0	0-2-0	2-0-0	0-0
DM-GILS-RVND	-	-	-	0-2-0	0-2-0	0-0

Table 7.23: Results for instances selected from TSPLib in [1, 2]

The results on the 56-instances set displayed in Table 7.24 indicates that MDM-GILS-RVND obtained substantial improvements when compared to the results of GILS-RVND and DM-GILS-RVND from Table 7.6. Indeed, the number of winnings in favor of the MDM heuristic raised in all analyzed elements, including in SST terms. Confronting the DM and MDM heuristics to each other, the MDM version outperformed the DM version when their general behaviors are considered, since in average solution aspect, MDM-GILS-RVND achieved 35 winnings, 15 ties and 6 losses compared to DM-GILS-RVND.

For the SSTs, the enhancement of results provided by MDM-GILS-RVND is verified by the growth of the number of instances statistically significant. Indeed, 16 instance results for the Student's t-test and 4 instance results for the Wilcoxon test were statistically significant when MDM-GILS-RVND performed better than GILS-RVND. Besides that, 13 instance results for the Student's t-test and 2 instance results for the Wilcoxon test were statistically significant, where MDM-GILS-RVND was better than DM-GILS-RVND. Finally, as this instance set is a new one for the MLP literature, all 56 best solutions obtained by the algorithms are regarded as new BKS, which are detailed in Table A.8.

Summary from Table A.2

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	4-32-20	11-10-35	0-13	4-33-19	5-12-39	0-20
DM-GILS-RVND	-	-	-	4-33-19	6-15-35	0-15

Summary from Table 7.6

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	5-32-19	15-11-30	1-7	7-33-16	12-14-30	1-10
DM-GILS-RVND	-	-	-	9-31-16	16-26-14	0-3

Table 7.24: Results on the 56-instances set selected from TSPLib

7.4.3.2 Experiments for Hamiltonian Path

For MLP version of Hamiltonian paths, all instance sets for this problem were submitted to the comparison proposed in this subsection, except for the sets of 10, 20, and 50 customers, since these sets have their average solutions matching their respective optimal values.

Table 7.25 displays the results on the heterogeneous set of instances that varies from 70 to 532 customers. Analyzing this table, we observed that the number of average solutions achieved by both DM heuristics was equal, with 3 winnings, 6 ties and 1 loss when compared to GILS-RVND. Another key point for this set is the improvement of the BKS for the att532 instance, which was achieved by MDM-GILS-RVND. A statistical significance result was found for the Student's t-test when MDM-GILS-RVND outperformed GILS-RVND and DM-GILS-RVND. One instance had its BKS improved by MDM-GILS-RVND, which is pointed in Table A.9.

Regarding the results obtained about best solution on the 100-customers set, displayed in Table 7.26, the two DM strategies matched all BKS. In terms of average solution, both

DM heuristics improved a instance compared to their results in direct comparison, shown in Table 7.13, which resulted in 17 ties and 3 winnings for GILS-RVND.

Summary from Table A.3

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	1-9-0	1-6-3	0-0	0-9-1	1-6-3	0-1
DM-GILS-RVND	-	-	-	0-8-2	0-6-4	0-1

Summary from Table 7.9

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	1-9-0	3-6-1	0-0	0-10-0	2-6-2	0-0
DM-GILS-RVND	-	-	-	0-9-1	2-6-2	0-0

Table 7.25: Results for TSPLib instances selected in [39]

Summary from Table A.4

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	0-20-0	3-17-0	0-0	0-20-0	3-17-0	0-0
DM-GILS-RVND	-	-	-	0-20-0	0-19-1	0-0

Summary from Table 7.13

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	0-20-0	4-16-0	0-0	0-20-0	4-16-0	0-0
DM-GILS-RVND	-	-	-	0-20-0	1-19-0	0-0

Table 7.26: Results on the 100-customers set generated in [39]

Table 7.27 reports the summarized results on the 200-customers set. Considering average solutions, the overall best performance of this set was obtained by DM-GILS-RVND, with 14 winnings, 3 ties and 3 losses against GILS-RVND. For the comparison between the DM heuristics, this statement is also valid, since DM-GILS-RVND obtained 9 winnings, 5 ties and 6 losses against MDM-GILS-RVND. An instance result was statistically significant, where DM-GILS-RVND outperformed MDM-GILS-RVND for the Wilcoxon test.

The computational results on the 500-instances set, shown in Table 7.28, demonstrates considerable improvements of DM-GILS-RVND and MDM-GILS-RVND over GILS-RVND results. Regarding the average solution aspect, MDM-GILS-RVND and DM-GILS-RVND obtained, respectively, 18 out of 20 and 15 out of 20 winnings in instance results when compared with GILS-RVND using the traditional comparison, as reported in Table 7.19.

Summary from Table A.5

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	0-20-0	3-3-14	0-1	0-20-0	5-2-13	0-1
DM-GILS-RVND	-	-	-	0-20-0	9-5-6	1-0

Summary from Table 7.16

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	0-20-0	10-1-9	0-0	0-20-0	10-1-9	0-1
DM-GILS-RVND	-	-	-	0-20-0	8-4-8	0-0

Table 7.27: Results on the 200-customers set generated in [39]

For the scenario using equal-time comparisons, MDM-GILS-RVND and DM-GILS-RVND achieved, respectively, 19 out of 20 and 18 out of 20 winnings in instance results compared with GILS-RVND, displayed in Table A.6. Thus, showing that the data mining heuristics presented better behaviors when compared to the original strategy. Regarding only DM-GILS-RVND and MDM-GILS-RVND results, the multi data mining version achieved the best performance of them, since this version obtained 14 out of 20 best average solutions compared to DM-GILS-RVND. For the SST results of Table A.6, where DM-GILS-RVND performed better than GILS-RVND, 4 instances using Student's t-test were statistically significant. When MDM-GILS-RVND had a greater performance than GILS-RVND, 11 instances using Student's t-test were statistically significant. At last, 5 instances were statistically significant when MDM-GILS-RVND performed better than DM-GILS-RVND for the Student's t-test. Finally, as shown Table A.9, 11 new BKS were obtained by DM-GILS-RVND and MDM-GILS-RVND.

Summary from Table A.6

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	1-4-15	2-0-18	0-4	1-1-18	1-0-19	0-11
DM-GILS-RVND	-	-	-	1-3-16	4-0-16	0-5

Summary from Table 7.19

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	3-2-15	5-0-15	0-4	1-4-15	2-0-18	0-5
DM-GILS-RVND	-	-	-	3-4-13	6-0-14	0-5

Table 7.28: Results on the 500-customers set generated in [39]

Table 7.29 reports the results on the 1000-customers set, which is the hardest one for this MLP variant. Observing the progress from the traditional comparison (Table 7.22)

to the equal-time comparison (Table A.7), the data mining heuristics demonstrated their greatest performance for this MLP variant, where these heuristics won in all instances against GILS-RVND. Comparing DM-GILS-RVND and MDM-GILS-RVND to each other, the multi data mining heuristic have been dominant against DM-GILS-RVND, outperforming all average solutions of DM-GILS-RVND. Regarding SST results, 14 instance results were statistically significant, where DM-GILS-RVND performed better than GILS-RVND, using the Student's t-test. Additionally, all 20 instance results using Student's t-test were statistically significant when MDM-GILS-RVND outperformed GILS-RVND. Moreover, 15 instance results using Student's t-test and 3 instance results using Wilcoxon test were statistically significant when MDM-GILS-RVND surpassed DM-GILS-RVND. Finally, 20 best solutions obtained by DM-GILS-RVND and MDM-GILS-RVND improved the BKSs of this instance set, described in Table A.9.

Summary from Table A.7

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	0-0-20	0-0-20	0-14	0-0-20	0-0-20	0-20
DM-GILS-RVND	-	-	-	3-0-17	0-0-20	0-18

Summary from Table 7.22

	DM-GILS-RVND			MDM-GILS-RVND		
	Best	Average	SST	Best	Average	SST
GILS-RVND	3-2-15	5-0-15	0-9	1-4-15	2-0-18	0-12
DM-GILS-RVND	-	-	-	3-4-13	6-0-14	0-5

Table 7.29: Results on the 1000-customers set generated in [39]

Chapter 8

Conclusion and Future Works

In this work, two hybrid heuristics using data mining techniques were conceived in the basis of a state-of-the-art heuristic (GILS-RVND) for two Minimum Latency Problem (MLP) variants.

The first proposed hybrid heuristic, named DM-GILS-RVND, was developed using DM-GRASP concepts, where, the data mining process is executed once and after the first half of multi-start iterations. Next, the second hybrid heuristic, named MDM-GILS-RVND, consists in an adapted version of the classic MDM-GRASP (Multi DM-GRASP) that performs the data mining process whenever the elite set gets updated.

In order to provide a fair and strict evaluation of the computational experiments, all experiments reported in [42] were entirely reproduced in this work. Additionally, a set of 56 instances selected from TSPLib was introduced into the experiments of the MLP version of Hamiltonian circuits, since the results of the initial 23-instances set of this MLP variant presented similar performances among the evaluated heuristics. In total, 229 instances were tested in this work, being 79 and 150 instances, respectively, for the MLP version of Hamiltonian circuits and Hamiltonian paths.

Moreover, analyses of time convergence (TTT plots), impact of mined patterns into initial solutions, and statistical significance tests were done to support the evaluation of the heuristics. Computational experiments using running time as stopping criterion were also carried out in this work.

Computational results reported in this work demonstrated that as the problem gets harder (i.e., the number of customers increases), the more efficient the data mining heuristics perform. Considering the results obtained from the reproduced experiments, DM-GILS-RVND and MDM-GILS-RVND proved to perform better than GILS-RVND in terms

of solution quality and computational time simultaneously. For example, in hard instances for both MLP variants – in this work, regarded as instances with 500 customers or more –, we observe that the improvements of solution quality made by the data mining heuristics were significant, since their numbers of wins over GILS-RVND were predominant. Regarding the experiments using equal-time comparisons, the data mining heuristics improved even more their results from the reproduced experiments over GILS-RVND, since the running time for the hybrid heuristics were increased, allowing them to further explore the search space of the problem.

It is important to highlight that MDM-GILS-RVND outperformed DM-GILS-RVND in terms of computational time and solution quality. For computational time, MDM-GILS-RVND obtained, in general, better average of running times than DM-GILS-RVND, when both heuristics are compared to GILS-RVND. Regarding these averages, MDM-GILS-RVND surpassed DM-GILS-RVND in all instance sets, except for the 100-customers set. In terms of solution quality, MDM-GILS-RVND started to have distinguishable results over DM-GILS-RVND as the instance size gets larger, as can be observed, for instance, in the results of the 56-instances set (a tested set from the MLP version of Hamiltonian circuits).

Finally, 56 new BKSs for the MLP version of Hamiltonian circuits and 32 new BKSs for the MLP version of Hamiltonian paths obtained in this work are, respectively, reported in Tables A.8 and A.9 of Appendix A.

A future improvement to be further investigated consists in the application of mined patterns to reduce the problem’s size, successfully developed in [25] for two Vehicle Routing Problem variants. Another challenging point that can also be explored regards using MDM-GILS-RVND concepts into other heuristics that require very few multi-start iterations, as GILS-RVND.

References

- [1] ABELEDO, H.; FUKASAWA, R.; PESSOA, A.; UCHOA, E. The Time Dependent Traveling Salesman Problem: Polyhedra and Branch-Cut-and-Price Algorithm. Tech. rep., Universidade Federal Fluminense, 2010.
- [2] ABELEDO, H.; FUKASAWA, R.; PESSOA, A.; UCHOA, E. The Time Dependent Traveling Salesman Problem: Polyhedra and Branch-Cut-and-Price Algorithm. In *Proceedings of the 9th International Symposium on Experimental Algorithms, SEA 2010* (Ischia Island, Italy, 2010), pp. 202–213.
- [3] AFRATI, F.; COSMADAKIS, S.; PAPADIMITRIOU, C. H.; PAPAGEORGIOU, G.; PAKOSTANTINO, N.; APAGEORGIOU, G. E. P. The complexity of the travelling repairman problem. *RAIRO Informatique théorique* 20 (1986), 79–87.
- [4] AGRAWAL, R.; SRIKANT, R. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases* (San Francisco, CA, USA, 1994), VLDB '94, Morgan Kaufmann Publishers Inc., pp. 487–499.
- [5] AIEX, R. M.; RESENDE, M. G. C.; RIBEIRO, C. C. TTT plots: A Perl Program to Create Time-to-target Plots. *Optimization Letters* 1 (2007), 355–366.
- [6] ANGEL-BELLO, F.; ALVAREZ, A.; GARCÍA, I. Two improved formulations for the minimum latency problem. *Applied Mathematical Modelling* 37 (2012), 2257–2266.
- [7] ARCHER, A.; BLASIAK, A. Improved approximation algorithms for the minimum latency problem via prize-collecting strolls. In *Proceedings of the 21th Annual ACM-SIAM Symposium on Discrete Algorithms* (2010), pp. 429–447.
- [8] BARBALHO, H.; ROSSETI, I.; MARTINS, S. L.; PLASTINO, A. A Hybrid Data Mining GRASP with Path-Relinking. *Computers and Operations Research* 40 (2013), 3159–3173.
- [9] BIANCO, L.; MINGOZZI, A.; RICCIARDELLI, S. The Traveling Salesman Problem with Cumulative Costs. *Networks* 23 (1993), 81–91.
- [10] BLUM, A.; CHALASANI, P.; COPPERSMITH, D.; PULLEYBLANK, B.; RAGHAVAN, P.; SUDAN, M. The Minimum Latency Problem. In *Proceedings of the 26th annual ACM symposium on Theory of computing* (Montreal, Canada, 1994), pp. 163–171.
- [11] CAMPBELL, A. M.; VANDENBUSSCHE, D.; HERMANN, W. Routing for Relief Efforts. *Transportation Science* 42 (2008), 127–145.

- [12] CHAUDHURI, K.; GODFREY, B.; RAO, S.; TALWAR, K. Paths, Trees, and Minimum Latency Tours. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2003* (Cambridge, The United States of America, 2003), pp. 36–45.
- [13] EIJL, C. A. V. A Polyhedral Approach to the Delivery Man Problem. Tech. rep., Eindhoven University of Technology, 1995.
- [14] EZZINE, I.; SEMET, F.; CHABCHOUB, H. New formulations for the traveling repairman problem. In *Proceedings of the 8th International Conference of Modeling and Simulation* (Hammamet, Tunisia, 2010).
- [15] FEO, T. A.; RESENDE, M. G. C. Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization* 6 (1995), 109–133.
- [16] FISCHETTI, M.; LAPORTE, G.; MARTELLO, S. The Delivery Man Problem and Cumulative Matroids. *Operations Research* 41 (1993), 1055–1064.
- [17] GRAHNE, G.; ZHU, J. Efficiently Using Prefix-trees in Mining Frequent Itemsets. In *Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations* (Melbourne, The United States of America, 2003), pp. 236–245.
- [18] GUERINE, M.; ROSSETI, I.; PLASTINO, A. Extending the Hybridization of Metaheuristics with Data Mining: Dealing with Sequences. *Intelligent Data Analysis* 20 (2016), 1133–1156.
- [19] HAN, J.; KAMBER, M.; PEI, J. *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann Publishers Inc., San Francisco, The United States of America, 2011.
- [20] HAN, J.; PEI, J.; YIN, Y. Mining Frequent Patterns without Candidate Generation. In *Proceedings of ACM SIGMOD'00* (2000), pp. 1–12.
- [21] HEILPORN, G.; CORDEAU, J. F.; LAPORTE, G. The Delivery Man Problem with time windows. *Discrete Optimization* 7 (2010), 269–282.
- [22] KINDERVATER, G. A.; SAVELSBERGH, M. W. Vehicle routing: handling edge exchanges. *Local Search in Combinatorial Optimization* (1997), 337–360.
- [23] LOURENÇO, H. R.; MARTIN, O. C.; STÜTZLE, T. Iterated local search. In *Handbook of Metaheuristics* (2003), F. Glover and G. A. Kochenberger, Eds., Springer New York, pp. 320–353.
- [24] LUCENA, A. Time-dependent traveling salesman problem - the deliveryman case. *Networks* 20 (1990), 753–763.
- [25] MAIA, M. R. D. H. Heurísticas Híbridadas com Mineração de Dados para o Problema de Roteamento de Veículos com Frota Heterogênea. Master's thesis, Universidade Federal Fluminense, 2015.
- [26] MARTIN, O.; OTTO, S. W.; FELTEN, E. W. Large-step Markov Chains for the Traveling Salesman Problem. *Complex Systems* 5 (1991), 299–326.

- [27] MARTINS, D.; VIANNA, G.; ROSSETI, I.; MARTINS, S. L.; PLASTINO, A. Making a State-of-the-art Heuristic Faster with Data Mining. *Annals of Operations Research* 263 (2018), 141–162.
- [28] MÉNDEZ-DÍAZ, I.; ZABALA, P.; LUCENA, A. A new formulation for the Traveling Deliveryman Problem. *Discrete Applied Mathematics* 156 (2008), 3223–3237.
- [29] MLADENović, N.; HANSEN, P. Variable Neighborhood Search. *Computers and Operations Research* 24 (1997), 1097–1100.
- [30] MLADENović, N.; UROŠEVIĆ, D.; HANAFI, S. Variable neighborhood search for the travelling deliveryman problem. *4OR: A Quarterly Journal of Operations Research* 11 (2013), 57–73.
- [31] NGUEVEU, S. U.; PRINS, C.; WOLFLER CALVO, R. An effective memetic algorithm for the cumulative capacitated vehicle routing problem. *Computers and Operations Research* 37 (2010), 1877–1885.
- [32] PLASTINO, A.; BARBALHO, H.; SANTOS, L. F. M.; FUCHSHUBER, R.; MARTINS, S. L. Adaptive and multi-mining versions of the DM-GRASP hybrid metaheuristic. *Journal of Heuristics* 20 (2014), 39–74.
- [33] PLASTINO, A.; FONSECA, E. R.; FUCHSHUBER, R.; FREITAS, A. A.; LUIS, M.; MARTINS, S. L. A Hybrid Data Mining Metaheuristic for the p -Median Problem. *Statistical Analysis and Data Mining* 4 (2011), 313–335.
- [34] R DEVELOPMENT CORE TEAM. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2008. ISBN 3-900051-07-0.
- [35] RIBEIRO, M. H.; PLASTINO, A.; MARTINS, S. L. Hybridization of GRASP Metaheuristic with Data Mining Techniques. *Journal of Mathematical Modelling and Algorithms* 5 (2006), 23–41.
- [36] RIBEIRO, M. H.; TRINDADE, V. A.; PLASTINO, A.; MARTINS, S. L. Hybridization of GRASP metaheuristics with data mining techniques. In *Hybrid Metaheuristics, First International Workshop, HM 2004, Valencia, Spain, August 22-23, 2004, Proceedings* (2004), pp. 69–78.
- [37] RIOS, E. *Exploração de Estratégias de Busca Local em Ambientes CPU / GPU*. PhD thesis, Institute of Computing, Fluminense Federal University, 2016.
- [38] SAHNI, S.; GONZALEZ, T. P-Complete Approximation Problems. *Journal of the ACM* 23 (1976), 555–565.
- [39] SALEHIPOUR, A.; SÖRENSEN, K.; GOOS, P.; BRÄYSY, O. Efficient GRASP+VND and GRASP+VNS metaheuristics for the traveling repairman problem. *4OR: A Quarterly Journal of Operations Research* 9 (2011), 189–209.
- [40] SANTOS, L. F.; MILAGRES, R.; ALBUQUERQUE, C. V.; MARTINS, S. L.; PLASTINO, A. A Hybrid GRASP with Data Mining for Efficient Server Replication for Reliable Multicast. In *IEEE Globecom 2006* (2006), pp. 1–6.

- [41] SANTOS, L. F. M.; RIBEIRO, M. H.; PLASTINO, A.; MARTINS, S. L. A Hybrid GRASP with Data Mining for the Maximum Diversity Problem. In *Proceedings of the International Workshop on Hybrid Metaheuristics* (2005), pp. 116–127.
- [42] SILVA, M. M.; SUBRAMANIAN, A.; VIDAL, T.; OCHI, L. S. A Simple and Effective Metaheuristic for the Minimum Latency Problem. *European Journal of Operational Research* 221 (2012), 513–520.
- [43] TALBI, E.-G. A Taxonomy of Hybrid Metaheuristics. *Journal of Heuristics* 8 (2002), 541–564.
- [44] VIDAL, T.; CRAININC, T. G.; GENDREAU, M.; PRINS, C. Timing Problems and Algorithms: Time Decisions for Sequences of Activities. *Networks* 62 (2015), 102–128.

APPENDIX A - COMPLEMENTARY EXPERIMENTS

This appendix presents the computational results of the complementary experiments reported in Subsections 5.4.3 and 7.4.3. These results were divided according to the tested MLP version: Section A.1 for Hamiltonian circuits and Section A.2 for Hamiltonian paths. In these sections, each table follows the same table format described in Section 5.1, except for the last column, where the average of the input running time for each execution is shown. Also in these tables, underlined results indicate new cost values of solutions achieved for the literature, and the Better counter stands for the number of best results encountered on the respective column. Finally, presented in Section A.3, all new Best Known Solutions (BKSs) for the MLP literature are compiled in Tables A.8 and A.9, regarding, respectively, BKSs for Hamiltonian circuits and Hamiltonian paths.

A.1 Experiments for Hamiltonian Circuit

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND		Average Time (s)
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution	
rat99	57986	57986	57986.0	57986	57986.0	57986	57986.0	5.30
eil101	27513	27513	27513.0	27513	27513.0	27513	27513.0	5.79
Better		2	2	2	2	2	2	

Table A.1: Results for instances selected from TSPLib in [1, 2]

Instance	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND		Average Time (s)
	Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution	
gr120	363454	363569.5	363454	363584.8	363454	363454.0	9.54
pr124	3154346	3154346.0	3154346	3154346.0	3154346	3154346.0	5.39
bier127	4545005	4546378.8	4545005	4545005.0	4545005	4545691.9	9.25
ch130	349874	349891.7	349874	349903.5	349874	349903.5	9.23
pr136	6199268	6199805.4	6199268	6200032.6	6199268	6200023.6	17.30
gr137	4061498	4061498.9	4061498	4061498.0	4061498	4061498.0	8.11
pr144	3846137	3846137.0	3846137	3846137.0	3846137	3846137.0	9.11
ch150	444424	444424.0	444424	444424.0	444424	444424.0	13.06
kroA150	1825769	1825769.0	1825769	1825769.0	1825769	1825769.0	19.84
kroB150	1786546	1786546.0	1786546	1786546.0	1786546	1786546.0	16.27
pr152	5064566	5064566.0	5064566	5064566.0	5064566	5064566.0	11.23
u159	2972030	2972204.2	2972030	2972291.3	2972030	2972204.2	14.21
si175	1808532	1808532.0	1808532	1808532.0	1808532	1808532.0	19.14
brg180	174750	174750.0	174750	174750.0	174750	174750.0	16.79
rat195	218632	218763.2	218632	218758.9	218632	218716.7	44.69
d198	1186049	1186098.6	1186049	1186061.4	1186049	1186049.0	38.28
kroA200	2672437	2672444.2	2672437	2672437.0	2672437	2672437.0	42.23
kroB200	2669515	2674486.0	2669515	2675761.6	2669515	2675993.6	42.00
gr202	2909247	2914644.2	2909247	2912564.8	2909247	2913251.2	35.95
ts225	13240046	13240046.0	13240046	13241020.0	13240046	13240046.0	26.60
tsp225	402783	403080.2	402783	402899.3	402783	402854.2	53.89
pr226	7196869	7196869.0	7196869	7196869.0	7196869	7196869.0	34.29
gr229	10725914	10729883.8	10725914	10728571.1	10725914	10730345.7	53.66
gil262	285060	285527.1	285043	285333.6	285060	285295.0	96.12
pr264	5471615	5471615.0	5471615	5471615.0	5471615	5471615.0	47.02
a280	346989	347125.9	346989	347000.1	346989	347106.5	107.18
pr299	6556628	6557983.4	6556628	6557893.5	6556628	6557788.6	104.92
lin318	5619810	5629995.9	5619810	5630371.5	5619810	5629823.8	117.98
rd400	2768830	2776672.7	2767608	2773810.9	2762336	2774472.8	354.00
fl417	1874242	1874242.8	1874242	1874242.0	1874242	1874242.0	382.64
gr431	21159702	21239150.9	21143311	21204699.7	21143311	21200881.7	336.98
pr439	17829541	17887107.0	17829541	17863294.5	17829541	17853186.3	285.56
pcb442	10301705	10323539.7	10290913	10319528.9	10301705	10319181.5	413.41
d493	6684190	6691057.1	6678021	6686588.4	6676086	6684218.8	608.47
att532	5613010	5632753.5	5621535	5628449.1	5613732	5623751.9	988.04
ali535	31904676.6	31904676.6	31870389	31894236.5	31860679	31880980.8	880.76
si535	12247211	12250679.7	12246397	12251565.8	12248003	12249794.2	498.76
pa561	658870	661211.6	659739	661800.0	659720	661022.3	1155.32
u574	9314596	9344178.4	9282414	9339267.3	9272607	9322872.8	1234.19
rat575	1848869	1859221.1	1846650	1855531.0	1845687	1853671.5	1739.46
p654	7827273	7827639.2	7827273	7827807.0	7827273	7827560.6	1755.28
d657	14159477	14220133.3	14112540	14183994.3	14111377	14169719.1	2615.66
gr666	63571693	63731966.5	63508352	63652863.2	63437941	63571619.5	2296.23
u724	13506660	13558605.3	13488521	13535072.2	13490438	13525798.1	4651.76
rat783	3282794	3296069.6	3268938	3286472.1	3266917	3282799.2	7044.52
dsj1000	7646018508	7685887300.0	7639381013	7668905895.3	7636325493	7659087450.1	18068.70
dsj1000ceil	7646519008	7683329486.0	7644298506	7679255707.2	7638241766	7658254573.7	18543.76
pr1002	115550770	116178260.2	115507699	115835876.6	115139627	115535663.8	11963.29
si1032	46896355	46897662.4	46896355	46896355.0	46896355	46896355.0	2402.72
u1060	102508056	102759766.0	102380219	102687446.4	102286639	102508247.6	15680.50
vm1084	94760440	95053081.2	94679736	94950688.9	94665088	94890532.8	13894.43
pcb1173	30926325	31032128.8	30861765	30946242.0	30772276	30896137.6	20508.89
d1291	29383346	29477239.4	29389729	29488806.8	29389729	29480929.9	12171.21
rl1304	144886001	145596878.7	144442951	145392224.6	144340259	145281356.8	18617.53
rl1323	155697857	156360364.3	155749119	156147959.5	155524088	155891209.6	22758.06
nrv1379	35360407	35519379.7	35309003	35421034.4	35185689	35322915.2	49624.72
Better	34	16	36	20	49	46	-

Table A.2: Results on the 56-instances set selected from TSPLib

A.2 Experiments for Hamiltonian Path

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND		Average Time (s)
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution	
st70	19215	19215	19215.0	19215	19215.0	19215	19215.0	0.94
rat99	54984	54984	54984.0	54984	54984.0	54984	54984.0	5.81
kroD100	949594	949594	949594.0	949594	949594.0	949594	949594.0	4.21
lin105	585823	585823	585823.0	585823	585823.0	585823	585823.0	3.79
pr107	1980767	1980767	1980767.0	1980767	1980767.0	1980767	1980767.0	4.96
rat195	210191	210191	210335.9	210191	210376.7	210191	210354.2	44.99
pr226	7100308	7100308	7100308	7100308	7100308.0	7100308	7100308.0	34.35
lin318	5560679	5560679	5569819.5	5560679	5569376.4	5560679	5568258.8	124.15
pr439	17688561	17688561	17734922.0	17702120	17730470.6	17688561	17723943.9	275.12
att532	5577965	5581240	5597866.8	5581240	5594946.4	5572131	5586649.3	923.03
Better	-	9	7	8	6	10	9	-

Table A.3: Results for TSPLib instances selected in [39]

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND		Average Time (s)
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution	
TRP-S100-R1	32779	32779	32779.0	32779	32779.0	32779	32779.0	4.33
TRP-S100-R2	33435	33435	33435.0	33435	33435.0	33435	33435.0	4.61
TRP-S100-R3	32390	32390	32390.0	32390	32390.0	32390	32390.0	4.33
TRP-S100-R4	34733	34733	34733.0	34733	34733.0	34733	34733.0	4.45
TRP-S100-R5	32598	32598	32598.0	32598	32598.0	32598	32598.0	5.43
TRP-S100-R6	34159	34159	34159.0	34159	34159.0	34159	34159.0	4.83
TRP-S100-R7	33375	33375	33375.0	33375	33375.0	33375	33375.0	5.34
TRP-S100-R8	31780	31780	31780.0	31780	31780.0	31780	31780.0	4.29
TRP-S100-R9	34167	34167	34167.0	34167	34167.0	34167	34167.0	4.56
TRP-S100-R10	31605	31605	31605.0	31605	31605.0	31605	31605.0	4.13
TRP-S100-R11	34188	34188	34198.5	34188	34210.6	34188	34209.0	4.72
TRP-S100-R12	32146	32146	32146.0	32146	32146.0	32146	32146.0	4.38
TRP-S100-R13	32604	32604	32604.0	32604	32604.0	32604	32604.0	4.75
TRP-S100-R14	32433	32433	32433.0	32433	32433.5	32433	32433.5	3.89
TRP-S100-R15	32574	32574	32574.0	32574	32574.0	32574	32574.0	4.36
TRP-S100-R16	33566	33566	33566.0	33566	33566.0	33566	33566.0	4.87
TRP-S100-R17	34198	34198	34198.0	34198	34198.0	34198	34198.0	5.65
TRP-S100-R18	31929	31929	31929.0	31929	31929.0	31929	31929.0	4.34
TRP-S100-R19	33463	33463	33463.0	33463	33463.0	33463	33463.0	4.91
TRP-S100-R20	33632	33632	33632.2	33632	33632.4	33632	33632.4	5.35
Better	-	20	20	20	17	20	17	-

Table A.4: Results on the 100-customers set generated in [39]

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND		Average Time (s)
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution	
TRP-S200-R1	88787	88787	88794.6	88787	88794.6	88787	88792.7	44.08
TRP-S200-R2	91977	91977	92013.1	91977	92009.8	91977	92003.5	41.02
TRP-S200-R3	92568	92568	92631.2	92568	92612.1	92568	92613.6	40.50
TRP-S200-R4	93174	93174	93192.3	93174	93180.4	93174	93186.8	43.48
TRP-S200-R5	88737	88737	88841.2	88737	88811.6	88737	88836.8	42.65
TRP-S200-R6	91589	91589	91601.9	91589	91589.0	91589	91596.3	42.43
TRP-S200-R7	92754	92754	92763.2	92754	92754.4	92754	92763.0	43.75
TRP-S200-R8	89048	89048	89049.0	89048	89051.0	89048	89051.0	45.13
TRP-S200-R9	86326	86326	86326.0	86326	86326.0	86326	86326.0	39.51
TRP-S200-R10	91552	91552	91596.5	91552	91598.1	91552	91627.5	44.63
TRP-S200-R11	92655	92655	92700.6	92655	92691.2	92655	92709.1	43.90
TRP-S200-R12	91457	91457	91504.1	91457	91492.3	91457	91483.5	46.08
TRP-S200-R13	86155	86155	86181.4	86155	86178.9	86155	86189.2	43.88
TRP-S200-R14	91882	91882	91929.1	91882	91892.7	91882	91892.7	42.60
TRP-S200-R15	88912	88912	88912.4	88912	88912.0	88912	88912.0	42.28
TRP-S200-R16	89311	89311	89364.7	89311	89316.6	89311	89311.0	47.08
TRP-S200-R17	89089	89089	89118.3	89089	89095.4	89089	89092.0	42.92
TRP-S200-R18	93619	93619	93676.6	93619	93641.5	93619	93648.4	46.36
TRP-S200-R19	93369	93369	93401.6	93369	93474.2	93369	93424.7	42.75
TRP-S200-R20	86292	86292	86292.0	86292	86292.0	86292	86292.0	42.42
Better	-	20	5	20	12	20	9	-

Table A.5: Results on the 200-customers set generated in [39]

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND		Average Time (s)
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution	
TRP-S500-R1	1841386	1841386	1856018.7	1841386	1848689.6	<u>1841210</u>	1848699.1	830.85
TRP-S500-R2	[†] 1815664	1816568	1823196.9	1817229	1821512.9	1817057	1821520.3	724.17
TRP-S500-R3	[†] 1826855	1833044	1839254.2	1831357	1837144.8	1826738	1836040.8	761.86
TRP-S500-R4	[†] 1804894	1809266	1815876.4	1806571	1815525.6	<u>1802921</u>	1811446.1	810.63
TRP-S500-R5	[†] 1821250	1823975	1834031.7	1823135	1831959.2	1823135	1829834.0	734.32
TRP-S500-R6	[†] 1782731	1786620	1790912.4	1785217	1789239.8	1783493	1789068.7	796.28
TRP-S500-R7	1847999	1847999	1857926.6	<u>1845736</u>	1853770.2	1846251	1852613.5	781.01
TRP-S500-R8	[†] 1819636	1820846	1829257.3	1820639	1828861.3	1820421	1827758.7	769.33
TRP-S500-R9	1733819	1733819	1737024.9	1730561	1735771.4	<u>1729796</u>	1734877.4	693.82
TRP-S500-R10	[†] 1761174	1762741	1767366.3	1762197	1767099.4	1761181	1764831.2	784.64
TRP-S500-R11	1797881	1797881	1801467.9	1797881	1802343.8	<u>1797771</u>	1801564.4	741.50
TRP-S500-R12	1774452	1774452	1783847.1	1774452	1780293.1	1774452	1780411.4	766.23
TRP-S500-R13	[†] 1863905	1873699	1878049.4	1867803	1875001.2	1866173	1872955.3	797.54
TRP-S500-R14	1799171	1799171	1805732.9	1797687	1802117.3	<u>1796129</u>	1801236.7	835.86
TRP-S500-R15	[†] 1785263	1791145	1797532.9	1788797	1794667.7	<u>1784919</u>	1791932.7	800.69
TRP-S500-R16	[†] 1804392	1810188	1816484.0	1808183	1814428.6	1807758	1814935.8	761.93
TRP-S500-R17	1825748	1825748	1834443.2	1824674	1831266.9	<u>1819909</u>	1829864.6	738.57
TRP-S500-R18	[†] 1825615	1826263	1833323.7	1826263	1831974.3	1826190	1830939.5	780.31
TRP-S500-R19	[†] 1776855	1779248	1782763.9	<u>1774846</u>	1784623.6	<u>1774846</u>	1781150.0	773.39
TRP-S500-R20	1820813	1820813	1830483.3	1820713	1828928.2	<u>1820168</u>	1827538.5	726.50
Better	-	2	1	4	4	18	15	-

[†] cost values from [37]

Table A.6: Results on the 500-customers set generated in [39]

Instance	BKS	GILS-RVND		DM-GILS-RVND		MDM-GILS-RVND		Average Time (s)
		Best Solution	Average Solution	Best Solution	Average Solution	Best Solution	Average Solution	
TRP-S1000-R1	5107395	5107395	5133698.3	5099593	5119372.4	5097334	5111463.4	19889.15
TRP-S1000-R2	5106161	5106161	5127449.4	5084762	5109703.2	5082922	5095563.3	19218.18
TRP-S1000-R3	5096977	5096977	5113302.9	5089701	5107248.7	5080369	5096720.7	18798.18
TRP-S1000-R4	[†] 5112465	5118006	5141392.6	5103527	5132953.5	5092276	5126208.5	18493.11
TRP-S1000-R5	[†] 5097991	5103894	5122660.7	5085136	5111351.7	5085412	5104769.0	18906.29
TRP-S1000-R6	[†] 5109946	5115816	5143087.1	5098364	5128407.7	5087134	5114959.6	19143.87
TRP-S1000-R7	[†] 4995703	5021383	5032722.0	4984950	5013911.2	4980214	5005857.5	17681.02
TRP-S1000-R8	5109325	5109325	5132722.6	5093854	5126224.1	5096892	5114194.5	18065.24
TRP-S1000-R9	[†] 5046566	5052599	5073245.3	5042296	5066052.5	5022622	5054773.5	17979.62
TRP-S1000-R10	[†] 5060019	5078191	5093592.6	5061767	5081279.2	5053780	5068111.1	17596.33
TRP-S1000-R11	[†] 5031455	5041913	5066161.5	5050689	5060465.6	5022026	5046756.0	18307.69
TRP-S1000-R12	5029792	5029792	5051235.2	5016822	5035312.4	5004216	5023893.7	19149.54
TRP-S1000-R13	5102520	5102520	5131437.5	5085029	5117785.7	5083179	5106627.2	19604.19
TRP-S1000-R14	[†] 5092861	5099433	5118980.6	5089565	5111582.3	5076253	5099226.9	18974.58
TRP-S1000-R15	[†] 5131013	5142470	5174493.2	5118229	5163524.7	5121456	5149904.2	18889.53
TRP-S1000-R16	[†] 5064094	5073972	5090280.5	5063177	5086418.2	5041659	5076827.7	18206.27
TRP-S1000-R17	[†] 5052283	5071485	5084450.4	5064257	5075264.3	5045873	5063241.1	18571.62
TRP-S1000-R18	[†] 5005789	5017589	5037094.0	5001170	5028868.0	4984408	5022960.1	19745.37
TRP-S1000-R19	[†] 5064873	5076800	5097167.6	5063691	5083769.6	5058047	5074667.8	19790.69
TRP-S1000-R20	4977262	4977262	5002920.6	4967954	5000000.7	4964371	4987995.2	18715.65
Better	-	0	0	3	0	17	20	-

[†] cost values from [37]

Table A.7: Results on the 1000-customers set generated in [39]

A.3 New Best Known Solutions for the Minimum Latency Problem

Instance	Heuristic	New BKS
gr120	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	363454
pr124	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	3154346
bier127	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	4545005
ch130	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	349874
pr136	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	6199268
gr137	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	4061498
pr144	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	3846137
ch150	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	444424
kroA150	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	1825769
kroB150	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	1786546
pr152	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	5064566
u159	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	2972030
si175	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	1808532
brg180	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	174750
rat195	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	218632
d198	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	1186049
kroA200	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	2672437
kroB200	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	2669515
gr202	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	2909247
ts225	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	13240046
tsp225	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	402783
pr226	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	7196869
gr229	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	10725914
gil262	DM-GILS-RVND	285043
pr264	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	5471615
a280	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	346989
pr299	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	6556628
lin318	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	5619810
rd400	MDM-GILS-RVND	2762336
fl417	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	1874242
gr431	DM-GILS-RVND and MDM-GILS-RVND	21143311
pr439	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	17829541
pcb442	DM-GILS-RVND	10290913
d493	MDM-GILS-RVND	6676086
att532	GILS-RVND	5613010
ali535	MDM-GILS-RVND	31860679
si535	DM-GILS-RVND	12246397
pa561	GILS-RVND	658870
u574	MDM-GILS-RVND	9272607
rat575	MDM-GILS-RVND	1845687
p654	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	7827273
d657	MDM-GILS-RVND	14111377
gr666	MDM-GILS-RVND	63437941
u724	DM-GILS-RVND	13488521
rat783	MDM-GILS-RVND	3266917
dsj1000	MDM-GILS-RVND	7636325493
dsj1000ceil	MDM-GILS-RVND	7638241766
pr1002	MDM-GILS-RVND	115139627
si1032	GILS-RVND, DM-GILS-RVND and MDM-GILS-RVND	46896355
u1060	MDM-GILS-RVND	102286639
vm1084	MDM-GILS-RVND	94665088
pcb1173	MDM-GILS-RVND	30772276
d1291	GILS-RVND	29383346
rl1304	MDM-GILS-RVND	144340259
rl1323	MDM-GILS-RVND	155524088
nrw1379	MDM-GILS-RVND	35185689

Table A.8: New cost values for the MLP version of Hamiltonian circuits

Instance	Heuristic	Current BKS	New BKS
att532	MDM-GILS-RVND	5577965	5572131
TRP-S500-R1	MDM-GILS-RVND	1841386	1841210
TRP-S500-R3	DM-GILS-RVND	1826855	1826738
TRP-S500-R4	MDM-GILS-RVND	1804894	1802921
TRP-S500-R7	DM-GILS-RVND	1847999	1845736
TRP-S500-R9	MDM-GILS-RVND	1733819	1729796
TRP-S500-R11	MDM-GILS-RVND	1797881	1797771
TRP-S500-R14	MDM-GILS-RVND	1799171	1796129
TRP-S500-R15	MDM-GILS-RVND	1785263	1784919
TRP-S500-R17	MDM-GILS-RVND	1825748	1819909
TRP-S500-R19	DM-GILS-RVND and MDM-GILS-RVND	1776855	1774846
TRP-S500-R20	MDM-GILS-RVND	1820813	1820168
TRP-S1000-R1	MDM-GILS-RVND	5107395	5097334
TRP-S1000-R2	MDM-GILS-RVND	5106161	5082922
TRP-S1000-R3	MDM-GILS-RVND	5096977	5080369
TRP-S1000-R4	MDM-GILS-RVND	5112465	5092276
TRP-S1000-R5	DM-GILS-RVND	5097991	5085136
TRP-S1000-R6	MDM-GILS-RVND	5109946	5087134
TRP-S1000-R7	MDM-GILS-RVND	4995703	4980214
TRP-S1000-R8	DM-GILS-RVND	5109325	5093854
TRP-S1000-R9	MDM-GILS-RVND	5046566	5022622
TRP-S1000-R10	MDM-GILS-RVND	5060019	5053780
TRP-S1000-R11	MDM-GILS-RVND	5031455	5022026
TRP-S1000-R12	MDM-GILS-RVND	5019792	5004216
TRP-S1000-R13	MDM-GILS-RVND	5102520	5083179
TRP-S1000-R14	MDM-GILS-RVND	5092861	5076253
TRP-S1000-R15	DM-GILS-RVND	5131013	5118229
TRP-S1000-R16	MDM-GILS-RVND	5064094	5041659
TRP-S1000-R17	MDM-GILS-RVND	5052283	5045873
TRP-S1000-R18	MDM-GILS-RVND	5005789	4984408
TRP-S1000-R19	MDM-GILS-RVND	5064873	5058047
TRP-S1000-R20	MDM-GILS-RVND	4977262	4964371

Table A.9: New cost values for the MLP version of Hamiltonian paths



Incorporando Mineração de Dados a uma Heurística Estado-da-Arte para o Problema da Mínima Latência*

Ítalo G. Santana, Alexandre Plastino, Isabel Rosseti

Instituto de Computação - Universidade Federal Fluminense

Rua Passo da Pátria, 156, Bloco Computação - CEP 24210-240 - Niterói - RJ - Brasil

{isantana,plastino,rosseti}@ic.uff.br

RESUMO

A hibridização de heurísticas com técnicas de mineração de dados tem fornecido resultados significativos para diversos problemas de otimização combinatória. Neste artigo, uma proposta de incorporação de mineração de dados em uma heurística estado-da-arte para o problema da mínima latência (PML) é apresentada. Experimentos computacionais realizados em 150 instâncias demonstraram que a heurística híbrida com mineração de dados foi capaz de igualar ou superar os resultados originais com menos esforço computacional. Além disso, 19 soluções com valores de custo menores que os da literatura foram obtidas.

PALAVRAS CHAVE. Metaheurísticas, Problema da Mínima Latência, Mineração de Dados.

Tópicos: MH - Metaheurísticas, OC - Otimização Combinatória

ABSTRACT

Hybridization of heuristics with data mining techniques has provided significant results for several combinatorial optimization problems. In this paper, an approach of incorporating data mining into a state-of-the-art heuristic for the minimum latency problem (MLP) is presented. Computational experiments carried out on 150 instances showed that the hybrid heuristic with data mining was able to match or improve the original results with less computational effort. Besides, 19 solutions with cost values less than those from literature were obtained.

KEYWORDS. Metaheuristics, Minimum Latency Problem, Data Mining.

Paper topics: MH - Metaheuristics, OC - Combinatorial Optimization

*Trabalho parcialmente financiado pela CAPES e CNPq.



1. Introdução

Como alternativa aos métodos exatos, metaheurísticas têm sido extensivamente aplicadas a problemas de otimização combinatória, pois oferecem soluções de boa qualidade em tempos de execução aceitáveis. Além disso, o interesse em versões híbridas de metaheurísticas tem aumentado bastante nos últimos anos, uma vez que, para muitos problemas de otimização, os melhores resultados da literatura são encontrados por essas metaheurísticas híbridas [Talbi, 2002]. Um exemplo bem-sucedido de hibridização de metaheurísticas foi empregado a uma heurística baseada em GRASP (*Greedy Randomized Adaptive Search Procedures*) [Feo e Resende, 1995] com técnicas de mineração de dados para o problema de empacotamento de conjuntos [Ribeiro et al., 2006], onde essa versão híbrida proposta foi capaz de alcançar melhores resultados em relação à versão original, tanto em termos de qualidade de solução quanto em tempo computacional.

Mineração de dados (MD), do inglês *Data Mining* (DM), é um processo de extração automática de conhecimento de bases de dados que pode ser representado por meio de padrões e regras [Han et al., 2011]. Esse processo foi primeiramente incorporado ao GRASP em [Ribeiro et al., 2006], nomeado de *Data Mining GRASP* (DM-GRASP). Essa proposta era baseada em identificar padrões em soluções de alta qualidade que poderiam ser utilizados como guia no espaço de busca do problema e, assim, de maneira eficiente, alcançar melhores resultados em menor tempo computacional, comparado com a heurística original. Posteriormente, o DM-GRASP foi aplicado ao problema de diversidade máxima e problema de replicação de servidores para transmissão *multicast* confiável [Santos et al., 2008], problema das *p*-medianas [Plastino et al., 2011], problema de projeto de redes a 2-caminhos [Barbalho et al., 2013] e ao problema do caixeiro viajante com coleta e entrega envolvendo um único tipo de produto [Guerine et al., 2016]. A incorporação de mineração de dados também foi realizada com sucesso na metaheurística *Iterated Local Search* (ILS) para o problema das *p*-medianas [Martins et al., 2014].

Neste trabalho, uma heurística estado-da-arte para o problema da mínima latência (PML), denominada GILS-RVND, desenvolvida por [Silva et al., 2012], é utilizada como heurística-base na proposta de hibridização com mineração de dados deste trabalho. A heurística original reúne componentes de GRASP, ILS e do método de busca local RVND (*Variable Neighborhood Descent with Random Neighborhood Ordering*). Resultados computacionais demonstraram que a versão híbrida com mineração de dados foi capaz de não somente melhorar os resultados da heurística original bem como obter uma redução considerável do tempo de execução, reduzindo, em média, 31.23% no grupo de instâncias de maior dimensão da literatura.

Este trabalho está organizado como se segue. A Seção 2 descreve o problema da mínima latência. As Seções 3 e 4 apresentam, em detalhes, o GILS-RVND e a nova proposta híbrida com mineração de dados, respectivamente. Em seguida, a Seção 5 reporta os experimentos computacionais sobre os conjuntos de instâncias utilizados por [Silva et al., 2012], bem como análises de significância estatística e de comportamento entre as duas heurísticas. Por fim, a Seção 6 aponta as conclusões deste estudo e trabalhos futuros.

2. O Problema da Mínima Latência

O problema da mínima latência, do inglês *Minimum Latency Problem*, é uma variante do problema do caixeiro viajante (PCV) e é definido como se segue. Seja $G = (V, A)$ um grafo completo direcionado, onde $V = \{0, \dots, n\}$ é o conjunto dos vértices, sendo o vértice 0 definido como o ponto de partida (depósito) e os outros definidos como os clientes, e $A = \{(i, j) : i, j \in V, i \neq j\}$ é o conjunto dos arcos, cada um associado a um tempo de viagem t_{ij} . Assim, o PML tem como objetivo determinar um circuito hamiltoniano que minimiza o tempo total de espera de todos os clientes. Esse tempo de espera, também chamado de latência, é definido como sendo $\sum_{i=0}^n l(i)$, onde $l(i)$ representa o tempo de viagem entre o depósito e $i \in V$. O problema ainda considera o depósito como ponto inicial e final do circuito e que $l(0) = 0$. Em alguns trabalhos, a restrição de retorno ao depósito não é considerada, e portanto, a solução se torna um caminho hamiltoniano, sendo dada por $\sum_{i=0}^{n-1} l(i)$. Na literatura, o PML também é conhecido como *Traveling Repairman*



Problem [Tsitsiklis, 1992], *Delivery Man Problem* [Fischetti et al., 1993], *Cumulative Traveling Salesman Problem* [Bianco et al., 1993] e *School Bus Driver Problem* [Chaudhuri et al., 2003].

Apesar de o PML ser uma variante do PCV, ele possui uma orientação voltada aos clientes do problema, e isso o torna mais difícil que o PCV, pois pequenas alterações na sequência de visita dos clientes podem impactar em grandes mudanças na solução. Assim como o PCV, o PML também é \mathcal{NP} -Difícil em sua versão de otimização [Blum et al., 1994]. Um exemplo de uma solução viável do PML é ilustrada na Figura 1. Seja S a solução na Figura 1(a) e sua sequência de clientes na Figura 1(b), a latência total (valor de custo) de S , ou $f(S)$, é denotado pela soma das latências $l(i)$ de cada cliente de S , que possuem custos acumulados.

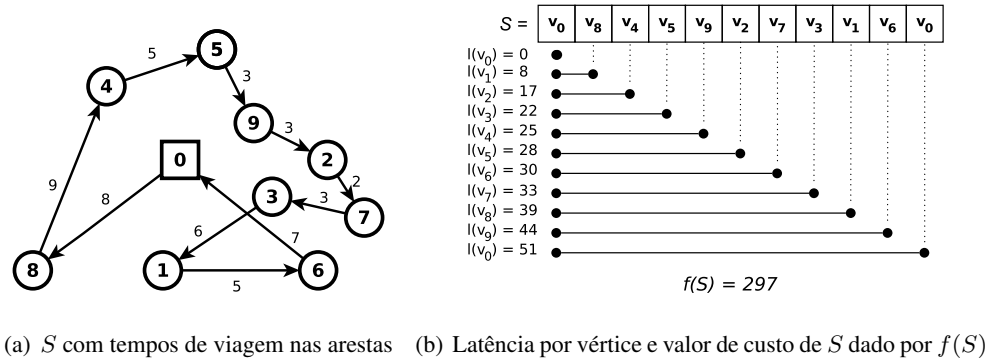


Figura 1: Exemplo de uma solução viável do PML e seu valor de custo. Figura adaptada de [Rios, 2016]

Devido às suas características, o PML pode ser facilmente encontrado em importantes aplicações práticas, onde o tempo de espera dos clientes é o fator crucial de análise, incluindo-se resgates em situações de desastres, serviços de entrega a clientes e recuperação de dados em redes de computadores [Moshref-Javadi e Lee, 2013]. Segundo [Abeledo et al., 2013], métodos exatos foram capazes de resolver instâncias de até 107 clientes, o que os tornam inviáveis em proporções realistas. Estratégias paralelas baseadas no GILS-RVND foram desenvolvidas para o mesmo problema em [Rios, 2016].

3. A heurística GILS-RVND

GILS-RVND é uma heurística baseada em GRASP [Feo e Resende, 1995], que é uma metaheurística multipartida composta por duas etapas: uma fase construtiva e uma estratégia de busca local. Na primeira etapa, uma solução inicial é gerada por meio de um método construtivo semiguloso controlado por um valor $\alpha \in [0, 1]$, que define o nível de aleatoriedade da construção, onde valores mais próximos de zero indicam uma construção mais gulosa. Em seguida, essa solução é submetida à etapa de busca local, onde é feita uma exploração sistemática do espaço de busca do problema até que o critério de parada seja satisfeito. É importante ressaltar que essas iterações são independentes, *i.e.*, a informação gerada por uma iteração não é propagada para as demais.

O pseudocódigo do GILS-RVND é apresentado no Algoritmo 1. Esse algoritmo realiza I_{Max} iterações (linhas 2-6), onde, para cada iteração, uma solução s é gerada pelo método construtivo, considerando um valor aleatório α de um conjunto R , parâmetro de entrada, que contém os possíveis valores para α (linha 3). Em seguida, s é submetida à etapa de busca local (linha 4). Logo depois, se s é melhor que a melhor solução global s^* , então s^* é atualizada (linhas 5-6). Após todas as iterações realizadas, s^* é retornada pelo algoritmo (linha 7).

O Algoritmo 2 descreve o método construtivo do GILS-RVND. No início, uma solução s é inicializada com o cliente 0 (linha 1), chamado de depósito, e a lista de clientes (LC) é gerada com os clientes remanescentes (linha 2). No laço, os clientes de LC são ordenados de forma crescente de acordo com o tempo de viagem entre cada cliente e o último cliente inserido em s (linha 5), que está sempre associado à variável r . Depois, a lista restrita de clientes (LRC) é preenchida com os $\alpha\%$



melhores clientes de LC (linha 6). Posteriormente, um cliente $c \in LRC$ é selecionado de forma aleatória, sendo inserido em s e removido de LC (linhas 7-10). Esse algoritmo termina quando LC se tornar vazia, e então, s é retornada pelo algoritmo (linha 11).

Algoritmo 1: GILS-RVND(I_{Max} , I_{ILS} , R)

```

1  $f(s^*) \leftarrow \infty$ ;
2 para  $i \leftarrow 1, \dots, I_{Max}$  faça
3    $s \leftarrow MetodoConstrutivo(\alpha \in R)$ ;
4    $s \leftarrow BuscaLocal(s, I_{ILS})$ ;
5   se  $f(s) < f(s^*)$  então
6      $s^* \leftarrow s$ ;
7 retorna  $s^*$ 
```

Algoritmo 2: MetodoConstrutivo(α)

```

1  $s \leftarrow \{0\}$ ;
2  $LC \leftarrow GerarLC()$ ;
3  $r \leftarrow 0$ ;
4 enquanto  $LC \neq \emptyset$  faça
5    $LC \leftarrow OrdenarLC(LC, r)$ ;
6    $LRC \leftarrow PreencherLRC(LC, \alpha)$ ;
7    $c \leftarrow SelecionarCliente(LRC)$ ;
8    $s \leftarrow s \cup \{c\}$ ;
9    $r \leftarrow c$ ;
10   $LC \leftarrow LC - \{c\}$ ;
11 retorna  $s$ 
```

A busca local reportada em [Silva et al., 2012], baseia-se na metodologia RVND, que por sua vez é uma extensão do método VNS (*Variable Neighborhood Search*) [Mladenović e Hansen, 1997]. O VNS é um método iterativo que aplica de forma sistemática um conjunto de vizinhanças, com a finalidade de melhorar a solução corrente de um determinado problema. Diferente do VNS onde a ordem de aplicação das vizinhanças é estipulada de maneira determinística, para o RVND a ordem é definida de forma aleatória. Em [Silva et al., 2012], cinco estruturas de vizinhanças clássicas do PCV são utilizadas no RVND: *Swap* - dois clientes da solução são trocados entre si; *2-opt* - dois arcos não adjacentes são removidos e dois outros são inseridos, mantendo a viabilidade da solução; *Reinsertion* - um cliente é realocado para outra posição da solução; *Or-opt-2* - um arco é realocado para outra posição da solução; *Or-opt-3* - dois arcos adjacentes são realocados para outra posição da solução.

Além do RVND, a etapa de busca local do GILS-RVND possui um mecanismo de perturbação originalmente proposto para o PCV, denominado de *double-bridge* [Martin et al., 1991]. Esse mecanismo consiste em remover quatro arcos e inserir quatro outros arcos na solução, mantendo a viabilidade da solução.

A estrutura básica da busca local é descrita no Algoritmo 3. Antes do laço, s' recebe uma cópia de s e é definida como a melhor solução da busca local (linha 1). No laço, definido entre as linhas 3 e 9, s é submetida ao RVND (linha 4). Caso s melhore s' , então s' é atualizada e o contador $iter_{ILS}$ é reiniciado (linhas 5-7). Em seguida, o mecanismo de perturbação é aplicado em s' (linha 8) e $iter_{ILS}$ é incrementado (linha 9). Esse laço é executado enquanto $iter_{ILS}$ for menor que I_{ILS} iterações sem melhora em s' (linhas 3-9), caso contrário, s' é retornada (linha 10).

4. Heurística Híbrida com Mineração de Dados: DM-GILS-RVND

Em mineração de dados, diversas técnicas podem ser aplicadas em bases de dados a fim de identificar regras e padrões. Uma dessas técnicas, conhecida como mineração de conjuntos frequentes (MCF), tem sido utilizada na criação de heurísticas híbridas com mineração de dados [Santos et al., 2008]. Basicamente, a MCF extrai conjuntos de elementos que aparecem com frequência na base de dados, de acordo com uma frequência mínima estabelecida, ou suporte mínimo (*SupMin*).



Em outras palavras, um conjunto frequente é aquele que ocorre em, pelo menos, $SupMin\%$ dos registros de uma base de dados.

Algoritmo 3: BuscaLocal(s, I_{ILS})

```
1  $s' \leftarrow s$ ;  
2  $iterILS \leftarrow 0$ ;  
3 enquanto  $iterILS < I_{ILS}$  faça  
4    $s \leftarrow RVND(s)$ ;  
5   se  $f(s) < f(s')$  então  
6      $s' \leftarrow s$ ;  
7      $iterILS \leftarrow 0$ ;  
8    $s \leftarrow Perturbacao(s')$ ;  
9    $iterILS \leftarrow iterILS + 1$ ;  
10 retorna  $s'$ 
```

Em geral, a incorporação dessa técnica em heurísticas é dividida em duas fases: a primeira é responsável por armazenar soluções de alta qualidade em um conjunto, denominado de conjunto elite, e a outra utiliza padrões encontrados nesse conjunto, que servirão como guia na exploração eficiente do espaço de busca do problema. Nesse contexto, esses padrões são obtidos por meio da aplicação da técnica MCF nas soluções (registros) do conjunto elite (base de dados). Para esse trabalho, o algoritmo FPMax*, que é uma implementação estado-da-arte na extração de conjuntos frequentes maximais, foi adotado na mineração desses padrões [Grahne e Zhu, 2003]. Um conjunto frequente é considerado maximal quando não existe nenhum superconjunto que o contenha.

No entanto, a identificação desses conjuntos não leva em consideração a ordem dos seus elementos e, portanto, a aplicação direta dessa técnica nas soluções do conjunto elite não pode ser realizada, pois as sequências de visitação de clientes, presente nas soluções, serão perdidas. Para solucionar esse novo problema, uma abordagem proposta por [Guerine et al., 2016], que realiza a mineração de arcos frequentes, foi adaptada nesse trabalho. Essa abordagem consiste em transformar cada par de clientes consecutivos (i e j) de uma solução para o seu respectivo arco ($a_{i \rightarrow j}$) e, assim, por meio de um mapeamento de arcos, a extração de padrões baseada em arcos pela técnica MCF mantém as ordens de visitação de clientes das soluções.

A Figura 2 ilustra o uso de arcos frequentes minerados. Considere um conjunto elite com duas soluções, representadas pelas Figuras 2(a) e 2(b), as quais são submetidas ao procedimento de mineração de dados com suporte igual a dois. Logo, somente os arcos que estão presentes nas duas soluções serão considerados frequentes (ver Figura 2(c)). Em seguida, esses arcos são usados na construção de soluções iniciais em um método construtivo adaptado, que será explicado posteriormente. Portanto, por meio desse método construtivo adaptado, soluções iniciais são geradas a partir de componentes frequentes de soluções de alta qualidade, como exemplificado na Figura 2(d).

A versão híbrida do GILS-RVND com mineração de dados, denominada DM-GILS-RVND, foi desenvolvida com base na hibridização que deu origem ao DM-GRASP [Santos et al., 2008]. Na versão híbrida desenvolvida (DM-GILS-RVND), o conjunto total de iterações do algoritmo original (GILS-RVND) é executado em duas partes com metade do número total de iterações em cada e possui, entre elas, o procedimento de mineração de dados. A primeira parte da versão híbrida é idêntica ao algoritmo original, exceto pela introdução do armazenamento de soluções no conjunto elite, onde uma solução só é inserida se essa é melhor que a pior solução do conjunto e diferente de todas as soluções do conjunto ou se o conjunto elite não estiver completo. Assim como a primeira, a segunda parte é idêntica ao algoritmo original, exceto pelo fato de o método construtivo ser substituído por uma abordagem de construção híbrida.

O Algoritmo 4 descreve em detalhes a proposta deste trabalho. A primeira parte (linhas 3-7) permanece igual ao algoritmo original, salvo pela etapa de busca local, a qual é diretamente derivada da busca local original. Essa etapa é idêntica ao Algoritmo 3, exceto pela introdução da função de atualização do conjunto elite Ω de tamanho d após o RVND, que se encarrega de tentar



inserir a solução s retornada pelo RVND em Ω . Logo após a primeira parte, o conjunto elite é submetido ao FPmax* (linha 8). Neste procedimento, são requeridos Ω , d , $SupMin$ e $MaxP$. O último parâmetro se refere a um valor inteiro que limita o número de padrões a serem minerados. Após esse procedimento, o minerador retorna uma lista de padrões P de tamanho $MaxP$ ordenados de forma crescente pela quantidade de arcos minerados por padrão. Na segunda parte (linhas 9-13), toda a estrutura do algoritmo original é mantida, exceto pela substituição do método construtivo original pelo método construtivo híbrido, onde, para cada iteração, um padrão $p \in P$ e um valor aleatório α do conjunto R são utilizados (linha 10).

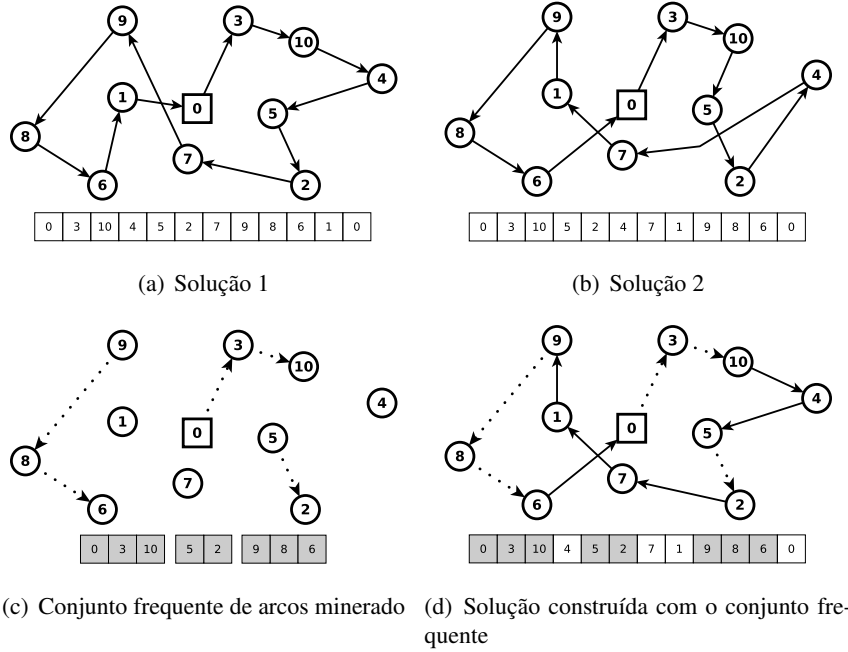


Figura 2: Processo de mineração de conjuntos frequentes de arcos

Algoritmo 4: DM-GILS-RVND(I_{Max} , I_{ILS} , R , $SupMin$, $MaxP$, d)

```

1  $f(s^*) \leftarrow \infty$ ;
2  $\Omega \leftarrow \emptyset$ ;
3 para  $i \leftarrow 1, \dots, I_{Max}/2$  faça
4    $s \leftarrow MetodoConstrutivo(\alpha \in R)$ ;
5    $s \leftarrow BuscaLocal(s, I_{ILS}, \Omega, d)$ ;
6   se  $f(s) < f(s^*)$  então
7      $s^* \leftarrow s$ ;
8  $P \leftarrow MineraPadroes(\Omega, d, SupMin, MaxP)$ ;
9 para  $i \leftarrow 1, \dots, I_{Max}/2$  faça
10   $s \leftarrow MetodoConstrutivoHibrido(\alpha \in R, p \in P)$ ;
11   $s \leftarrow BuscaLocal(s, I_{ILS})$ ;
12  se  $f(s) < f(s^*)$  então
13     $s^* \leftarrow s$ ;
14 retorna  $s^*$ 

```

No método construtivo híbrido descrito no Algoritmo 5, os elementos dos padrões minerados são aproveitados na construção de soluções iniciais. Inicialmente, após o depósito ser inserido na solução s (linha 1), as listas de arcos consecutivos (LAC) encontradas em p são geradas (linha 2), onde cada lista será utilizada de forma integral quando o primeiro cliente da lista for selecionado. A lista de clientes (LC), baseada em LAC, é gerada com os clientes que não pertencem a nenhuma lista de LAC e com aqueles que estão na primeira posição de cada lista de LAC (linha 3).



Caso o depósito seja o primeiro elemento de alguma lista de LAC, então essa lista é inserida em s e removida de LAC (linhas 4-6). No laço, os clientes de LC são ordenados de forma crescente de acordo com o tempo de viagem de cada cliente e r (último cliente inserido em s) (linha 9). A seguir, LRC é preenchida com os $\alpha\%$ melhores clientes de LC, para que um cliente $c \in LRC$ seja selecionado de forma aleatória (linhas 10-11). Se c é o primeiro cliente de alguma lista de LAC, então essa lista é inserida em s e removida de LAC (linhas 12-14). Caso contrário, somente c será inserido em s (linhas 15-16). Em seguida, c é removido de LC e o último cliente inserido em s é associado a r como referência na ordenação da próxima iteração (linhas 17-18). Este laço termina quando LC estiver vazia e, então, s é retornada (linha 19).

Algoritmo 5: MetodoConstrutivoHibrido(α, p)

```

1  $s \leftarrow \{0\}$ ;
2  $LAC \leftarrow GerarLAC(p)$ ;
3  $LC \leftarrow GerarLC(LAC)$ ;
4 se  $\{0 \text{ é o primeiro cliente de } lac \mid lac \subset LAC\}$  então
5    $s \leftarrow s \cup lac$ ;
6    $LAC \leftarrow LAC - lac$ ;
7  $r \leftarrow SelecionarUltimoCliente(s)$ ;
8 enquanto  $LC \neq \emptyset$  faça
9    $LC \leftarrow OrdenarLC(LC, r)$ ;
10   $LRC \leftarrow PreencherLRC(LC, \alpha)$ ;
11   $c \leftarrow SelecionarCliente(LRC)$ ;
12  se  $\{c \text{ é o primeiro cliente de } lac \mid lac \subset LAC\}$  então
13     $s \leftarrow s \cup lac$ ;
14     $LAC \leftarrow LAC - lac$ ;
15  senão
16     $s \leftarrow s \cup \{c\}$ ;
17     $LC \leftarrow LC - \{c\}$ ;
18     $r \leftarrow SelecionarUltimoCliente(s)$ ;
19 retorna  $s$ 

```

5. Experimentos Computacionais

O código-fonte do GILS-RVND desenvolvido por [Silva et al., 2012] foi cedido pelos autores para a incorporação da técnica de mineração de dados. Ambas heurísticas foram codificadas em C++ (g++ 4.4.3) e executadas em um Intel®Core™ i7 3,40GHz, com 16GB de memória RAM e GNU/Linux Ubuntu 14.04 de 64 bits.

Em ambas abordagens, todos os experimentos foram realizados sob o mesmo conjunto de parâmetros definidos por [Silva et al., 2012], os quais são $I_{Max} = 10$, $I_{ILS} = \min\{100; n\}$ e $R = \{0, 00; 0, 01; \dots; 0, 25\}$, onde n representa o número de clientes da instância. Para a versão hibridizada com mineração de dados, os parâmetros $d = 10$, $SupMin = 7$ e $MaxP = 5$ foram utilizados com base em testes preliminares. Estes experimentos foram realizados em todas as 150 instâncias da versão de caminho hamiltoniano do PML em [Silva et al., 2012], as quais são divididas em: sete grupos de 20 instâncias de mesma dimensão com 10, 20, 50, 100, 200, 500 e 1000 clientes (Grupos 1 a 7) e um grupo de dez instâncias variando entre 70 e 532 clientes¹ (Grupo 8).

Para realizar uma avaliação justa e rigorosa, os experimentos envolvendo o GILS-RVND foram completamente reproduzidos, utilizando a mesma versão de compilador, as mesmas sementes de números aleatórios e dez execuções por instância, características definidas por [Silva et al., 2012]. Posteriormente, o DM-GILS-RVND também foi submetido a essas mesmas condições a fim de apresentar uma análise comparativa justa entre as duas heurísticas testadas.

Nas Tabelas 1 e 2, os resultados dos grupos das instâncias de 500 e 1000 clientes são detalhados, visto que esses grupos são os maiores e mais difíceis da literatura. Nessas tabelas

¹Instâncias do TSPLib: st70, rat99, kroD100, rat195, lin105, pr107, pr226, lin318, pr439, att532. Disponíveis em: <http://comopt.ifl.uni-heidelberg.de/software/TSPLIB95/tsp/>



são reportados os resultados de melhor solução obtida, solução média e tempo médio em segundos de ambas heurísticas sobre os Grupos 6 e 7, com os valores em negrito indicando o melhor resultado obtido entre as duas heurísticas e custos sublinhados mostrando os melhores valores de custo de solução para a literatura. Para os resultados de melhor solução e solução média, também são apresentadas as diferenças percentuais desses valores em relação à melhor solução conhecida (*BKS*), que são dadas por $\Delta\% = \frac{100 \times (Valor - BKS)}{BKS}$, onde *Valor* denota o valor de custo (melhor ou médio) de solução reportado. Os valores de *BKS* reportados estão associados aos símbolos \star e \dagger que indicam a origem, [Silva et al., 2012] e [Rios, 2016], respectivamente. Na última coluna são apresentadas as diferenças percentuais dos tempos médios entre as duas heurísticas, que são calculadas por $\Delta\% = \frac{100 \times (Tempo_{DMGILS} - Tempo_{GILS})}{Tempo_{GILS}}$. Além disso, o nome de cada instância foi reduzido para o seu respectivo sufixo, as quais estão no formato TRP-S500-Ri e TRP-S1000-Ri para os grupos de 500 e de 1000 clientes, respectivamente. Por último, todas as médias gerais das diferenças percentuais são apresentadas na última linha.

Instância	BKS	GILS-RVND					DM-GILS-RVND					
		Melhor Solução	$\Delta\%$ Melhor	Solução Média	$\Delta\%$ Média	Tempo Médio	Melhor Solução	$\Delta\%$ Melhor	Solução Média	$\Delta\%$ Média	Tempo Médio	$\Delta\%$ Tempo
R1	1841386 \star	1841386	0,000	1856018,7	0,795	830,85	1841386	0,000	1852238,0	0,589	620,18	-25,36
R2	1815664 \dagger	1816568	0,050	1823196,9	0,415	724,17	1817288	0,089	1821667,9	0,331	502,99	-30,54
R3	1826855 \dagger	1833044	0,339	1839254,2	0,679	761,86	1831357	0,246	1838557,2	0,641	580,34	-23,83
R4	1804894 \dagger	1809266	0,242	1815876,4	0,608	810,63	1808563	0,203	1816765,0	0,658	589,33	-27,30
R5	1821250 \dagger	1823975	0,150	1834031,7	0,702	734,32	1823135	0,104	1831575,2	0,567	565,36	-23,01
R6	1782731 \dagger	1786620	0,218	1790912,4	0,459	796,28	1785217	0,139	1789675,6	0,390	596,65	-25,07
R7	1847999 \star	1847999	0,000	1857926,6	0,537	781,01	1847089	-0,049	1854324,8	0,342	609,98	-21,90
R8	1819636 \dagger	1820846	0,066	1829257,3	0,529	769,33	1820639	0,055	1829831,8	0,560	580,50	-24,55
R9	1733819 \star	1733819	0,000	1737024,9	0,185	693,82	1730561	-0,188	1736278,3	0,142	545,16	-21,43
R10	1761174 \dagger	1762741	0,089	1767366,3	0,352	784,64	1762197	0,058	1767912,7	0,383	587,06	-25,18
R11	1797881 \star	1797881	0,000	1801467,9	0,200	741,50	1797881	0,000	1802957,9	0,282	536,77	-27,61
R12	1774452 \star	1774452	0,000	1783847,1	0,529	766,23	1774452	0,000	1781588,2	0,402	564,99	-26,26
R13	1863905 \dagger	1873699	0,525	1878049,4	0,759	797,54	1867803	0,209	1875973,1	0,647	556,95	-30,17
R14	1799171 \star	1799171	0,000	1805732,9	0,365	835,86	1798130	-0,058	1803379,2	0,234	585,61	-29,94
R15	1785263 \dagger	1791145	0,329	1797532,9	0,687	800,69	1790524	0,295	1795858,3	0,593	587,39	-26,64
R16	1804392 \dagger	1810188	0,321	1816484,0	0,670	761,93	1808183	0,210	1814941,6	0,585	590,35	-22,52
R17	1825748 \star	1825748	0,000	1834443,2	0,476	738,57	1824674	-0,059	1832635,8	0,377	533,61	-27,75
R18	1825615 \dagger	1826263	0,035	1833323,7	0,422	780,37	1826263	0,035	1831998,5	0,350	636,26	-18,46
R19	1776855 \dagger	1779248	0,135	1782763,9	0,333	773,39	1774846	-0,113	1785575,3	0,491	553,04	-28,49
R20	1820813 \star	1820813	0,000	1830483,3	0,531	726,50	1820713	-0,005	1829740,0	0,490	544,20	-25,09
Média			0,125		0,512			0,059		0,453		-25,59

Tabela 1: Resultados sobre as instâncias do Grupo 6

Os experimentos computacionais do Grupo 6, reportados na Tabela 1, apontam que o comportamento médio do DM-GILS-RVND foi superior ao GILS-RVND com base em 15 melhores resultados de solução média. A diferença percentual média da versão híbrida com mineração de dados em relação aos valores de BKS foi 0,453%. Já a da heurística-base foi 0,512%. Assim, os resultados da versão híbrida com mineração de dados se aproximam mais do BKS do que a heurística original. Em termos de melhor solução, o DM-GILS-RVND também superou o GILS-RVND em 15 instâncias, sendo seis dessas novos valores de custos para a literatura, e com diferença percentual de 0,059% contra 0,125%. Ainda sobre este resultado, houve quatro empates e somente uma vitória do GILS-RVND. O DM-GILS-RVND também se mostrou superior em relação ao tempo computacional médio, onde obteve médias de tempo melhores que a heurística original em todas as instâncias, sendo a abordagem proposta, em geral, 25,59% mais rápida que a versão original.

A Tabela 2, que reporta os experimentos computacionais do Grupo 7, apresenta novamente o DM-GILS-RVND com comportamento médio superior ao GILS-RVND com base em 15 melhores resultados de solução média. Neste caso, a versão híbrida com mineração de dados apresentou 0,461% de percentual médio em relação ao BKS, ao passo que a versão original apresentou 0,566%, indicando, mais uma vez, que a nova proposta apresentou resultados mais próximos dos valores de BKS que a heurística original. A heurística híbrida com mineração de dados melhorou 17 melhores soluções encontradas pela heurística original, sendo 13 desses novos valores de custos



para a literatura, e com diferença percentual em relação ao BKS de -0,042% contra 0,141%. Ainda, houve um empate e duas vitórias do GILS-RVND. Em termos de tempo computacional médio, a versão proposta foi mais eficiente que a heurística original em todos os casos, reduzindo o esforço computacional, em média, 31,01% neste grupo.

Instância	BKS	GILS-RVND					DM-GILS-RVND					$\Delta\%$ Tempo
		Melhor Solução	$\Delta\%$ Melhor	Solução Média	$\Delta\%$ Média	Tempo Médio	Melhor Solução	$\Delta\%$ Melhor	Solução Média	$\Delta\%$ Média	Tempo Médio	
R1	5107395*	5107395	0,000	5133698,3	0,515	19889,15	5099593	-0,153	5122436,1	0,294	14123,42	-28,99
R2	5106161*	5106161	0,000	5127449,4	0,417	19218,17	5084762	-0,419	5112406,6	0,122	13532,12	-29,59
R3	5096977*	5096977	0,000	5113302,9	0,320	18798,18	5089701	-0,143	5111390,5	0,283	12553,89	-33,22
R4	5112465‡	5118006	0,108	5141392,6	0,566	18493,11	5110184	-0,045	5142219,9	0,582	12404,32	-32,92
R5	5097991‡	5103894	0,116	5122660,7	0,484	18906,29	5085136	-0,252	5116167,7	0,357	13124,07	-30,58
R6	5109946‡	5115816	0,115	5143087,1	0,649	19143,87	5102671	-0,142	5132201,1	0,436	13862,07	-27,59
R7	4995703‡	5021383	0,514	5032722,0	0,741	17681,02	4984950	-0,215	5014796,3	0,382	13154,44	-25,60
R8	5109325*	5109325	0,000	5132722,6	0,458	18065,23	5109325	0,000	5129942,9	0,404	12781,96	-29,25
R9	5046566‡	5052599	0,120	5073245,3	0,529	17979,62	5045262	-0,026	5070469,7	0,474	11758,44	-34,60
R10	5060019‡	5078191	0,359	5093592,6	0,664	17596,33	5070109	0,199	5089275,6	0,578	12274,40	-30,24
R11	5031455‡	5041913	0,208	5066161,5	0,690	18307,69	5052459	0,417	5066612,8	0,699	11826,31	-35,40
R12	5029792*	5029792	0,000	5051235,2	0,426	19149,54	5030837	0,021	5043866,1	0,280	12940,67	-32,42
R13	5102520*	5102520	0,000	5131437,5	0,567	19604,18	5098034	-0,088	5123032,4	0,402	13895,03	-29,12
R14	5092861‡	5099433	0,129	5118980,6	0,513	18974,58	5089565	-0,065	5116989,7	0,474	13171,29	-30,58
R15	5131013‡	5142470	0,223	5174493,2	0,847	18889,53	5123240	-0,151	5166046,9	0,683	13687,70	-27,54
R16	5064094‡	5073972	0,195	5090280,5	0,517	18206,27	5070422	0,125	5091420,9	0,540	11962,50	-34,29
R17	5052283‡	5071485	0,380	5084450,4	0,637	18571,62	5065952	0,271	5082717,2	0,602	11847,47	-36,21
R18	5005789‡	5017589	0,236	5037094,0	0,625	19745,37	5003047	-0,055	5038269,6	0,649	12966,44	-34,33
R19	5064873‡	5076800	0,235	5097167,6	0,638	19790,68	5065743	0,017	5087261,2	0,442	13365,27	-32,47
R20	4977262*	4977262	0,000	5002920,6	0,516	18715,65	4970735	-0,131	5003599,3	0,529	13150,60	-29,73
Média			0,147		0,566			-0,042		0,461		-31,23

Tabela 2: Resultados sobre as instâncias do Grupo 7

Por último, em função dos valores de custo de solução de ambas heurísticas nos Grupos 6 e 7 se aproximarem de uma amostra com uma distribuição normal, verificados por meio do teste de normalidade de Shapiro-Wilk, o teste estatístico t Student unicaudal pareado com nível de significância igual a 5% foi aplicado. Em quatro instâncias do Grupo 6, a diferença entre as médias apresentou significância estatística a favor do DM-GILS-RVND e nenhuma para o GILS-RVND. Para o Grupo 7, em sete instâncias, a diferença entre as médias apresentou significância estatística para o DM-GILS-RVND e nenhuma para o GILS-RVND.

A Tabela 3 reúne os resultados dos experimentos de todos os grupos de instâncias entre as duas heurísticas tratadas neste trabalho. Nesta tabela, as duas primeiras colunas identificam os grupos já definidos e suas respectivas dimensões. Nas duas colunas seguintes, os tempos médios de todos os grupos para cada heurística e, nas três colunas seguintes, as diferenças percentuais de melhor solução, solução média e tempo computacional médio em segundos entre as duas heurísticas, que foram calculados usando a fórmula $\Delta\% = \frac{100 \times (\text{ResultadoDMGILS} - \text{ResultadoGILS})}{\text{ResultadoGILS}}$.

Origem	Dimensão	GILS-RVND	DM-GILS-RVND	Melhor Solução ($\Delta\%$)	Solução Média ($\Delta\%$)	Tempo Médio ($\Delta\%$)
		Tempo Médio	Tempo Médio			
Grupo 1	10	0,010	0,010	0,000	0,000	0,00
Grupo 2	20	0,020	0,020	0,000	0,000	0,00
Grupo 3	50	0,350	0,340	0,000	0,000	-2,86
Grupo 4	100	4,675	4,180	0,000	0,000	-10,58
Grupo 5	200	43,372	35,475	0,000	0,000	-18,21
Grupo 6	500	770,465	573,335	-0,066	-0,059	-25,59
Grupo 7	1000	18786,305	12919,120	-0,188	-0,105	-31,23
Grupo 8	70-532	142,135	111,890	0,008	0,002	-21,28

Tabela 3: Oito grupos de instâncias executados utilizando as duas heurísticas testadas

Analisando a Tabela 3, o comportamento médio do DM-GILS-RVND, medido pela diferença percentual de solução média, foi superior ao GILS-RVND nos Grupos 6 e 7, empatando nos



Grupos 1, 2, 3, 4 e 5 e perdendo no Grupo 8. É importante salientar que, a vitória do GILS-RVND apresentou uma ligeira vantagem, sendo essa heurística muito mais lenta que a versão proposta. Outro aspecto importante é a tendência de melhoria do DM-GILS-RVND, tanto em qualidade de solução, quanto em tempo computacional médio, à medida que a dimensão da instância aumenta e o problema se torna mais difícil.

5.1. Análises de Comportamento das Heurísticas

A fim de analisar e entender o impacto do uso da mineração de dados, os dois algoritmos foram executados usando-se a instância TRP-S1000-R1 com a mesma semente inicial e os valores de custo gerados pela etapa construtiva e de busca local de cada iteração são mostrados na Figura 3.

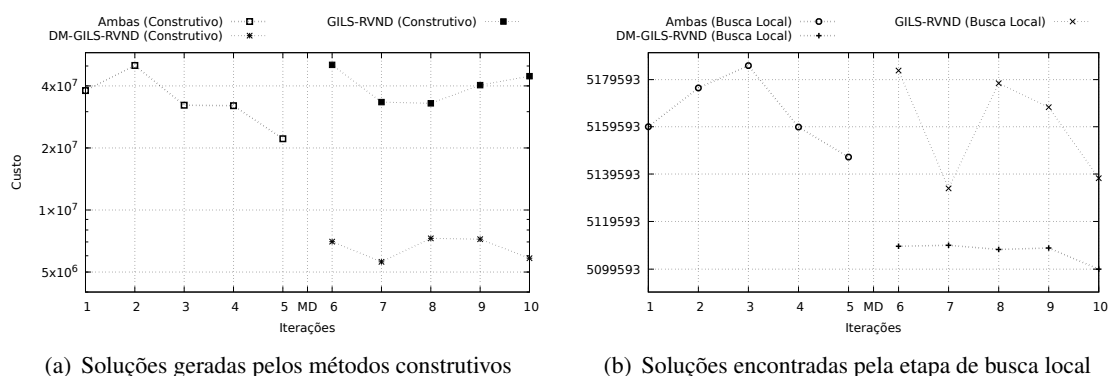


Figura 3: Gráficos Custo versus Iteração - TRP-S1000-R1

As duas heurísticas possuem comportamento idêntico nas cinco primeiras iterações, antes da execução do processo de mineração. Na Figura 3(a), logo após a etapa de mineração de dados (MD no eixo horizontal), o emprego dos arcos minerados reduz drasticamente os valores de custo das soluções iniciais geradas pelo método construtivo híbrido em comparação com os valores de custo das soluções iniciais geradas pelo método construtivo original. Neste caso específico, o número médio de arcos frequentes minerados é de 931 por padrão minerado. A Figura 3(b) considera apenas os custos das soluções encontradas pela etapa de busca local. Nessa figura, é possível notar que o DM-GILS-RVND, após a execução do processo de mineração de dados, possui valores de custo de solução menores e, portanto, mais próximos dos valores de custo das melhores soluções dessa instância em relação aos valores de custos das soluções apresentadas pelo GILS-RVND.

Uma maneira de avaliar comportamentos de algoritmos com componentes aleatórias é por meio de gráficos *Time-to-Target Plots (TTTPlots)* [Aiex et al., 2007]. Esses gráficos apresentam, no eixo das ordenadas, a probabilidade acumulada do algoritmo encontrar uma solução pelo menos tão boa quanto um dado alvo (valor de custo) dentro de um determinado tempo, representado no eixo das abscissas. A Figura 4 apresenta os gráficos *TTTPlots* para as duas heurísticas, utilizando a instância TRP-S500-R5, e dois alvos distintos: 1834031 (mais fácil) e 1831575 (mais difícil). Nesta avaliação, as heurísticas foram executadas 100 vezes com o mesmo conjunto de 100 sementes iniciais distintas.

Na Figura 4(a), o comportamento do DM-GILS-RVND, em 95% das suas execuções, atinge o alvo definido em aproximadamente 1750 segundos, enquanto, para essa mesma probabilidade, o GILS-RVND necessita de um pouco mais de 3100 segundos. Verifica-se também que o DM-GILS-RVND atinge o alvo em 60% das execuções em torno de 500 segundos, enquanto, para este mesmo tempo, a probabilidade para o GILS-RVND alcançar este alvo cai para 45%. No segundo gráfico, que apresenta um alvo mais difícil que o anterior, a Figura 4(b) mostra que a eficiência do DM-GILS-RVND em atingir o alvo é de 100% em cerca de 3600 segundos, ao passo que, para a heurística original, o tempo necessário para alcançar esse mesmo alvo em todas as execuções foi um pouco mais de 5500 segundos. Além disso, a versão híbrida com mineração de



dados atinge o alvo em 90% das execuções em menos de 2500 segundos, enquanto, para a versão original, o tempo necessário aumenta para um pouco mais de 3200 segundos.

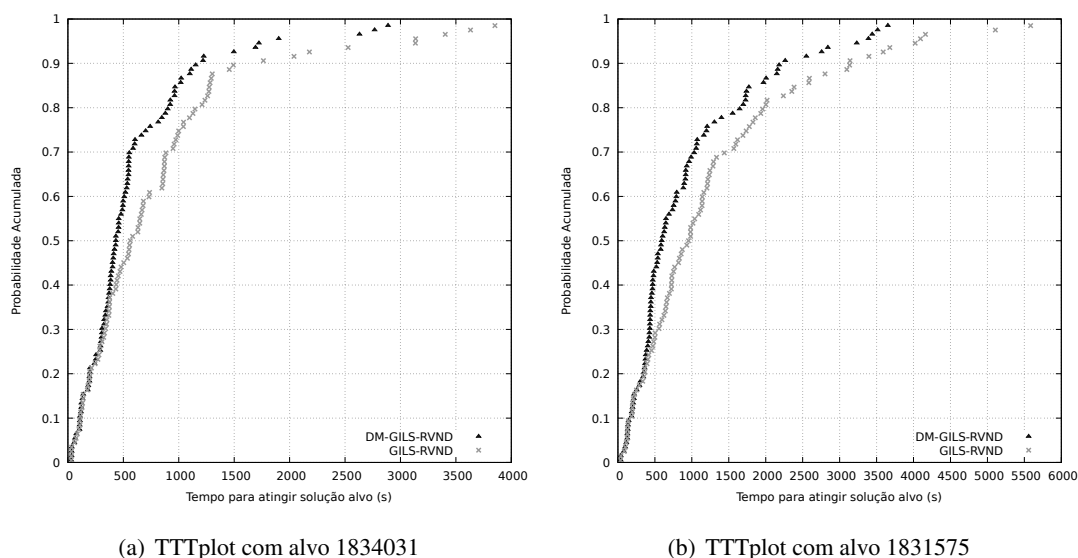


Figura 4: Gráficos TTTplots para alvos da instância TRP-S500-R5

6. Conclusões e Trabalhos Futuros

A heurística híbrida com mineração de dados introduzida neste trabalho apresentou-se como uma abordagem bem-sucedida para o PML. Esse comportamento foi evidenciado por meio dos experimentos computacionais realizados em 150 instâncias. Não somente a eficiência da nova abordagem proposta em encontrar melhores soluções foi superior à da heurística original, mas também houve uma redução perceptível do tempo computacional, especialmente para instâncias de 500 e 1000 clientes. Além disso, a heurística híbrida com mineração de dados foi capaz de superar 19 melhores valores de custo de solução existentes na literatura. Portanto, a hibridização proposta neste trabalho demonstrou ganhos significativos de qualidade de solução e de tempo de execução, principalmente nas maiores e mais difíceis instâncias do PML da literatura.

Em trabalhos futuros, pretende-se aperfeiçoar a hibridização de mineração de dados no GILS-RVND. Para tanto, pretende-se executar a mineração de dados como no modelo *Multi* DM-GRASP (MDM-GRASP) [Santos et al., 2008], realizando minerações sucessivas de padrões de modo adaptativo ao longo da execução do algoritmo.

Referências

- Abeledo, H., Fukasawa, R., Pessoa, A., e Uchoa, E. (2013). The Time Dependent Traveling Salesman Problem: Polyhedra and Algorithm. *Mathematical Programming Computation*, 5:27–55.
- Aiex, R. M., Resende, M. G. C., e Ribeiro, C. C. (2007). TTT plots: A Perl Program to Create Time-to-target Plots. *Optimization Letters*, 1:355–366.
- Barbalho, H., Rosseti, I., Martins, S. L., e Plastino, A. (2013). A Hybrid Data Mining GRASP with Path-Relinking. *Computers and Operations Research*, 40:3159–3173.
- Bianco, L., Mingozzi, A., e Ricciardelli, S. (1993). The Traveling Salesman Problem with Cumulative Costs. *Networks*, 23:81–91.
- Blum, A., Chalasani, P., Coppersmith, D., Pulleyblank, B., Raghavan, P., e Sudan, M. (1994). The Minimum Latency Problem. In *Proceedings of the 26th annual ACM symposium on Theory of computing*, p. 163–171, Montreal, Canadá.



- Chaudhuri, K., Godfrey, B., Rao, S., e Talwar, K. (2003). Paths, Trees, and Minimum Latency Tours. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2003*, p. 36–45, Cambridge, Estados Unidos.
- Feo, T. A. e Resende, M. G. C. (1995). Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization*, 6:109–133.
- Fischetti, M., Laporte, G., e Martello, S. (1993). The Delivery Man Problem and Cumulative Matroids. *Operations Research*, 41:1055–1064.
- Grahne, G. e Zhu, J. (2003). Efficiently Using Prefix-trees in Mining Frequent Itemsets. In *Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, p. 236–245, Melbourne, Estados Unidos.
- Guerine, M., Rosseti, I., e Plastino, A. (2016). Extending the Hybridization of Metaheuristics with Data Mining: Dealing with Sequences. *Intelligent Data Analysis*, 20:1133–1156.
- Han, J., Kamber, M., e Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, Estados Unidos, 3rd edition.
- Martin, O., Otto, S. W., e Felten, E. W. (1991). Large-step Markov Chains for the Traveling Salesman Problem. *Complex Systems*, 5:299–326.
- Martins, D., Vianna, G., Rosseti, I., Martins, S., e Plastino, A. (2014). Making a State-of-the-art Heuristic Faster with Data Mining. *Aceito para publicação em: Annals of Operations Research*.
- Mladenović, N. e Hansen, P. (1997). Variable Neighborhood Search. *Computers and Operations Research*, 24:1097–1100.
- Moshref-Javadi, M. e Lee, S. (2013). A Taxonomy to the Class of Minimum Latency Problems. In *Proceedings of the 2013 Industrial and Systems Engineering Research Conference*, p. 3896–3905, San Juan, Porto Rico.
- Plastino, A., Fonseca, E. R., Fuchshuber, R., Freitas, A. A., Luis, M., e Martins, S. L. (2011). A Hybrid Data Mining Metaheuristic for the p-Median Problem. *Stat. Anal. Data Min.*, 4:313–335.
- Ribeiro, M. H., Plastino, A., e Martins, S. L. (2006). Hybridization of GRASP Metaheuristic with Data Mining Techniques. *Journal of Mathematical Modelling and Algorithms*, 5:23–41.
- Rios, E. (2016). *Exploração de Estratégias de Busca Local em Ambientes CPU / GPU*. Tese de Doutorado, Programa de Pós-graduação em Computação, Instituto de Computação, Universidade Federal Fluminense.
- Santos, L. F. M., Martins, S. L., e Plastino, A. (2008). Applications of the DM-GRASP Heuristic: A Survey. *International Transactions in Operational Research*, 15:387–416.
- Silva, M. M., Subramanian, A., Vidal, T., e Ochi, L. S. (2012). A Simple and Effective Metaheuristic for the Minimum Latency Problem. *European Journal of Operational Research*, 221:513–520.
- Talbi, E.-G. (2002). A Taxonomy of Hybrid Metaheuristics. *Journal of Heuristics*, 8:541–564.
- Tsitsiklis, J. N. (1992). Special Cases of Traveling Salesman and Repairman Problems with Time Windows. *Networks*, 22:263–282.