Open University of Catalonia

Doctoral Thesis

From Metaheuristics to Learnheuristics: Applications to Logistics, Finance, and Computing

Author:
Laura Calvet Liñán

Thesis committee:
Dr. Ángel A. Juan Pérez
Dr. Carles Serrat i Piè
Dr. David Masip Rodó

A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Network and Information Technologies

in the

Internet Computing & Systems Optimization (ICSO)
Internet Interdisciplinary Institute (IN3)

May 25, 2017

OPEN UNIVERSITY OF CATALONIA

Abstract

Internet Interdisciplinary Institute (IN3)

Doctor of Network and Information Technologies

From Metaheuristics to Learnheuristics: Applications to Logistics, Finance, and Computing

by Laura Calvet Liñán

A large number of decision-making processes in strategic sectors such as transport, production and finance involve $\mathcal{NP}$-hard problems. Trends such as globalization make systems larger and more complex. Frequently, these problems are characterized by high levels of uncertainty and dynamism. Metaheuristics have become predominant methods for solving challenging optimization problems in reasonable computing times. However, they frequently assume that the inputs, the objective functions, and the set of optimization constraints are deterministic and known in advance. These constitute strong assumptions that lead to work on oversimplified versions of real-world problems. As a consequence, the solutions obtained may have a poor performance when implemented. Simheuristics integrate simulation into metaheuristics to solve stochastic optimization problems in a natural way. Similarly, learnheuristics combine statistical learning and metaheuristics to tackle optimization problems in dynamic environments, where inputs may depend on the structure of the solution.

From a methodological perspective, the main contributions of this thesis are the design of learnheuristics and a classification of works hybridizing statistical / machine learning and metaheuristics. It discusses the potential of learnheuristics in a number of fields and studies two specific cases. The first is a routing problem in which the depots are heterogeneous, in terms of their commercial offer, and customers show different willingness to consume depending on how well the assigned depot fits their preferences. Thus, different customer-depot assignment maps lead to different customer-expenditure levels. Regression models are employed to capture the relationship between each customer's willingness to spend as a function of several variables, including the assigned depot as well as other customer's features (age, gender, etc.). The second case describes a vehicle routing problem where each customer's demand depends on the order in which the customers are visited. Moreover, several applications are presented in transport, production, finance, and computing. In the first field, the smart design of routes, including capacitated depots and vehicles, are addressed analyzing stochastic demands, and sustainability indicators. Moreover, the waste collection problem and a routing problem with a heterogeneous fleet, asymmetric costs and site-dependency are studied. In the production arena, the optimization of jobs' sequences under stochasticity, considering multiple production lines and a common deadline, is discussed. Strategies to invest on risky assets are proposed and assessed. Finally, the parameter fine-tuning of metaheuristics and the effect of the number of agents and the computing time on metaheuristics' performance are investigated.

OPEN UNIVERSITY OF CATALONIA

Resum

Internet Interdisciplinary Institute (IN3)

Doctor of Network and Information Technologies

From Metaheuristics to Learnheuristics: Applications to Logistics, Finance, and Computing

by Laura CALVET LIÑÁN

Un gran nombre de processos de presa de decisions en sectors estratègics com el transport, la producció i les finances impliquen problemes $\mathcal{NP}$-difícils. Tendències com la globalització fan que els sistemes siguin cada cop més grans i complexos. Sovint, aquests problemes es caracteritzen per alts nivells d'incertesa i dinamisme. Les metaheurístiques s'han convertit en mètodes molt populars per resoldre problemes d'optimització difícils en temps de càlcul raonables. No obstant això, sovint assumeixen que els inputs, les funcions objectiu, i el conjunt de restriccions d'optimització són deterministes i coneguts. Aquests constitueixen supòsits forts que obliguen a treballar en versions simplistes de problemes del món real. Com a conseqüència, les solucions poden conduir a resultats pobres quan s'apliquen. Les simheurístiques integren la simulació a les metaheurístiques per resoldre problemes d'optimització estocàstica d'una manera natural. Anàlogament, les learnheurístiques combinen l'estadística amb les metaheurístiques per fer front a problemes d'optimització en entorns dinàmics, on els inputs poden dependre de l'estructura de la solució.

Des d'un punt de vista metodològic, les principals contribucions d'aquesta tesi són el disseny de les learnheurístiques i una classificació dels treballs que combinen l'estadística / l'aprenentatge automàtic i les metaheurístiques. La tesi discuteix el potencial de les learnheurístiques en un conjunt de camps i estudia dos casos específics. El primer és un problema de rutes en el qual els magatzems són heterogenis, en termes de la seva oferta comercial, i els clients mostren diferents disposicions a consumir en funció de com el magatzem assignat s'ajusti a les seves preferències. Per tant, diferents mapes d'assignació de clients a magatzems condueixen a diferents nivells de despesa. Es fan servir models de regressió per representar la relació entre la disposició de cada client per consumir com una funció de diverses variables, incloent-hi el magatzem assignat, així com característiques d'altres clients (edat, gènere, etc.). El segon cas descriu un problema de rutes on la demanda de cada client depèn de l'ordre en què es visiten aquests. D'altra banda, es presenten diverses aplicacions en el transport, la producció, les finances, i la informàtica. En el primer camp, el disseny intel·ligent de rutes, incloent-hi magatzems i vehicles amb capacitat limitada, s'aborda analitzant demandes estocàstiques i indicadors de sostenibilitat. A més a més, el problema de la recol·lecció de residus i un problema de rutes amb una flota heterogènia, i costos asimètrics i dependents del lloc s'estudien. En l'àmbit de la producció, es discuteix l'optimització de seqüències de tasques considerant estocasticitat, múltiples línies de producció i una data límit. Es proposen i avaluen estratègies per invertir en actius de risc. Finalment, s'investiguen la selecció de valors dels paràmetres de les metaheurístiques i l'efecte de la quantitat d'agents i del temps de càlcul en el rendiment de les metaheurístiques.

OPEN UNIVERSITY OF CATALONIA

Resumen

Internet Interdisciplinary Institute (IN3)

Doctor of Network and Information Technologies

From Metaheuristics to Learnheuristics: Applications to Logistics, Finance, and Computing

by Laura CALVET LIÑÁN

Un gran número de procesos de toma de decisiones en sectores estratégicos como el transporte, la producción y las finanzas implican problemas $\mathcal{NP}$-difíciles. Tendencias como la globalización hacen que los sistemas sean cada vez más grandes y complejos. Con frecuencia, estos problemas se caracterizan por altos niveles de incertidumbre y dinamismo. Las metaheurísticas se han convertido en métodos muy usados para resolver problemas difíciles de optimización en tiempos de computación razonables. Sin embargo, suelen asumir que los inputs, las funciones objetivo y el conjunto de restricciones de optimización son deterministas y se conocen de antemano. Estas fuertes suposiciones conducen a trabajar en versiones simplificadas de problemas del mundo real. Como consecuencia, las soluciones obtenidas pueden tener un pobre rendimiento cuando se implementan. Las simheurísticas integran simulación en metaheurísticas para resolver problemas de optimización estocástica de una manera natural. De manera similar, las learnheurísticas combinan aprendizaje estadístico y metaheurísticas para abordar problemas de optimización en entornos dinámicos, donde los inputs pueden depender de la estructura de la solución.

Desde un punto de vista metodológico, las principales aportaciones de esta tesis son el diseño de las learnheurísticas y la clasificación de los trabajos que combinan estadística / aprendizaje automático y metaheurísticas. La tesis discute el potencial de las learnheurísticas en una serie de campos y estudia dos casos específicos. El primero es un problema de enrutamiento en el que los almacenes son heterogéneos, en términos de su oferta comercial, y los clientes muestran una disposición diferente de consumir dependiendo de lo bien que el almacén asignado se ajuste a sus preferencias. Por lo tanto, diferentes mapas de asignación de clientes a almacenes conducen a diferentes niveles de consumo. Se utilizan modelos de regresión para representar la relación entre la disposición de cada cliente a gastar en función de varias variables, incluyendo el almacén asignado, así como las características de otros clientes (edad, género, etc.). El segundo caso describe un problema de enrutamiento de vehículos en el que la demanda de cada cliente depende del orden en que se visitan los clientes. Además, se presentan varias aplicaciones en transporte, producción, finanzas e informática. En el primer campo, el diseño inteligente de rutas, incluyendo almacenes y vehículos con capacidad limitada, se aborda analizando demandas estocásticas e indicadores de sostenibilidad. También se estudia el problema de la recolección de residuos y un problema de enrutamiento con una flota heterogénea, y costes asimétricos y en función del sitio. En el ámbito de la producción, se analiza la optimización de secuencias de tareas considerando estocasticidad, múltiples líneas de producción y un plazo común. Se proponen y evalúan estrategias para invertir en activos de riesgo. Finalmente, se investigan el ajuste de parámetros de metaheurísticas y el efecto del número de agentes y el tiempo de computación en el rendimiento de las metaheurísticas.

Acknowledgements

First of all I would like to thank my supervisors Dr. Angel A. Juan and Dr. Carles Serrat for their guidance, support, innovative ideas and timely feedbacks. I owe a great debt of gratitude for their patience and inspiration. I am also grateful to my advisor Dr. David Masip for his supervision, help and motivation. I feel highly privileged to have worked with him.

I would also like to thank my colleagues in the Internet Computing & Systems Optimization (ICSO) research group and collaborators for their support both in research and personal life. My warmest greetings to Enoc Martínez, Jana Doering, Dr. Javier Panadero, Lorena Reyes, Dr. Sara Hatami, Dr. Carlos Quintero, Aljoscha Gruler, Carla Talens, Stephanie M. Alvarez, Dr. Adela Pages and Dr. Jésica de Armas.

I would like to extend my sincere acknowledgment to Dr. Madeleine Lopeman and Dr. Guillermo Franco, and Dr. Renatas Kizys for their hospitality during the research stays in Dublin and Portsmouth, respectively.

I am also grateful to those who offer me their help in the most crucial stage, the admission to the doctoral program. I would like to express my very warm gratitude to Dr. Joan Manuel Marquès for his encouragement, and Dr. Daniel Riera and Dr. Àlex Sànchez for their support.

Likewise, I want to convey my gratitude to the Doctoral School of the Open University of Catalonia (UOC) for all the help and guidance provided. This thesis would have never been possible without the support of the Internet Interdisciplinary Institute (IN3) that funded me.

Finally, I would like to thank my family and friends (present and gone, humans and dogs) for their tireless support and endless love. They have brought me here.

To all of them I dedicate this thesis.

Contents

List of Figures

List of Tables

List of Abbreviations

ABC	**A**rtificial **B**ee **C**olony
ACO	**A**nt **C**olony **O**ptimization
AIS	**A**rtificial **I**mmune **S**ystems
ALNS	**A**daptive **LNS**
ARP	**A**rc **R**outing **P**roblem
ARPO	**A**lgorithm for **R**ich **P**ortfolio **O**ptimization
ASP	**A**lgorithm **S**election **P**roblem
BA	**B**at **A**lgorithm
BDS	**B**est **D**eterministic **S**olution
BKS	**B**est **K**nown **S**olution
BSS	**B**est **S**tochastic **S**olution
CBR	**C**ase-**B**ased **R**easoning
CEF	**C**onstrained **E**fficient **F**rontier
COP	**C**ombinatorial **O**ptimization **P**roblem
COPDI	**COP** with **D**ynamic **I**nputs
CVRP	**C**apacitated **VRP**
CWS	**C**larke and **W**right **S**avings
DE	**D**ifferential **E**volution
DOE	**D**esign **O**f **E**xperiments
DPCS	**D**istributed and **P**arallel **C**omputing **S**ystems
DFSP	**D**istributed **F**lowshop **S**cheduling **P**roblem
DPFSP	**D**istributed **PFSP**
DPFSP-ST	**DPFSP** with **S**tochastic **T**imes
DSS	**D**ecision **S**upport **S**ystem
EA	**E**volutionary **A**lgorithm
EDA	**E**stimation of **D**istribution **A**lgorithm
EITP	**E**nhanced **I**ndex **T**racking **P**roblem
FA	**F**irefly **A**lgorithm
FSP	**F**lowshop **S**cheduling **P**roblem
GA	**G**enetic **A**lgorithm
GP	**G**enetic **P**rogramming
GRASP	**G**reedy **R**andomized **A**daptive **S**earch **P**rocedure
GVRP	**G**reen **VRP**
HBMO	**H**oney **B**ees **M**ating **O**ptimization
HS	**H**armony **S**earch
HSAVRP	**H**eterogeneous **S**ite-dependent **A**symmetric **VRP**
HoSAVRP	**H**omogeneous **S**ite-dependent **A**symmetric **VRP**
HSAVRP-SD	**HSAVRP** with **S**tochastic **D**emands
ILS	**I**terated **L**ocal **S**earch
IG	**I**terated **G**reedy
IPTS	**I**nstance-specific **P**arameter **T**uning **S**trategies
IRP	**I**nventory **R**outing **P**roblem
ITP	**I**ndex **T**racking **P**roblem
IWO	**I**nvasive **W**eed **O**ptimization
LNS	**L**arge **N**eighborhood **S**earch
LR	**L**ogistic **R**egression
LRP	**L**ocation **R**outing **P**roblem
MDVRP	**M**ulti-**D**epot **VRP**
MDVRP-HD	**MDVRP** with **H**eterogeneous **D**epots
MDVRP-SD	**MDVRP** with **S**tochastic **D**emands

MCS	**M**onte **C**arlo **S**imulation
MFN	**M**ulti-layer **F**eedforward **N**etwork
MLR	**M**ultiple **L**inear **R**egression
MOEA	**M**ulti-**O**bjective **EA**
MS	**M**ulti-**S**tart
NN	**N**eural **N**etwork
OR	**O**perations **R**esearch
PCA	**P**rincipal **C**omponents **A**nalysis
PCS	**P**arameter **C**ontrol **S**trategies
PFSP	**P**ermutation **F**lowshop **S**cheduling **P**roblem
PFSP-ST	**PFSP** with **S**tochastic processing **T**imes
PM	**P**opulation-based **M**etaheuristic
POP	**P**ortfolio **O**ptimization **P**roblem
PSO	**P**article **S**warm **O**ptimization
PSP	**P**arameter **S**etting **P**roblem
PTS	**P**arameter **T**uning **S**trategies
PVRP	**P**ollution **VRP**
RAT	**R**adio **A**ccess **T**echnologies
RMP	**R**isk **M**anagement **P**roblem
RVRP	**R**ich **VRP**
SA	**S**imulated **A**nnealing
SAM	**S**uccessive **A**pproximations **M**ethod
SOM	**S**elf-**O**rganizing **M**aps
SCOP	**S**tochastic **COP**
SM	**S**ingle-solution based **M**etaheuristic
SMEs	**S**mall and **M**edium **E**nterprises
SPOP	**S**tochastic **POP**
SR-GCWS-CS	**S**imulation in **R**outing via the **G**eneralized **CWS** heuristic with **C**ache and **S**plitting
SS	**S**catter **S**earch
SVM	**S**upport **V**ector **M**achine
TS	**T**abu **S**earch
TSP	**T**ravelling **S**alesman **P**roblem
UEF	**U**nconstrained **E**fficient **F**rontier
VNS	**V**ariable **N**eigborhood **S**earch
VRP	**V**ehicle **R**outing **P**roblem
VRP-SD	**VRP** with **S**tochastic **D**emands
WCP	**W**aste **C**ollection **P**roblem
WCP-SW	**WCP** with **S**tochastic **W**aste levels

Chapter 1

Introduction

This chapter introduces the thesis. Its sections are: motivation, main goal and original contributions, and dissertation outline.

1.1 Motivation

Metaheuristics constitute a heterogeneous family of algorithms designed to solve a high number of complex combinatorial optimization problems (COPs) without having to deeply adapt them to each problem (Boussaïd et al., 2013). They represent the first recourse for $\mathcal{NP}$-hard problems that are required to be solved in real time. Their quickness and flexibility to address realistic and rich (i.e., complex) problems are two significant advantages in comparison to exact methods. However, they can not guarantee optimal solutions, but tend to provide near-optimal ones. As synthesized in Talbi (2009), "optimization is everywhere; optimization problems are often complex; then metaheuristics are everywhere". Certainly, metaheuristics are highly popular among researchers and companies, and may be found in a large number of fields. For instance, their use is frequent in: routing, scheduling, telecommunications, machine learning, cryptology, etc. The first works on heuristics (more experience-based procedures) were written in the 1940s (in particular, Polya, 1945, is considered to be the first), but metaheuristics started to be widely used in the 1970s and 1980s. Since then, their importance in operation research (OR) has rapidly increased.

Fields such as logistics (including both transport and production logistics), finance and computing are strategic sectors for all developed economies. A number of complex decision-making processes in real-life related applications can be modeled as COPs (Faulin et al., 2012). All theses problems are $\mathcal{NP}$-hard in nature, which leads to the use of metaheuristics. In logistics, road transport is key for the efficient flow of goods in supply chains. The multi-depot vehicle routing problem (MDVRP) represents a non-trivial extension of the classical vehicle routing problem (VRP) combining assignation and routing issues. In the MDVRP with heterogeneous depots (MDVRP-HD), the customers show different willingness to consume depending on how well the assigned depot fits their preferences. The MDVRP with stochastic demands (MDVRP-SD) allows demands to follow probability distributions, either theoretical o empirical ones. Also in the routing arena, the interest of the waste collection problem (WCP) and the WCP with stochastic waste levels (WCP-SW) is growing due to the expansion of cities and the relevance of the negative externalities of this service. In fact, there is a growing concern for environmental and social impacts of routing activities in general, which calls for the design of routes based on sustainability indicators. Regarding production, task scheduling is present in the elaboration of products, the design of timetables, etc. The permutation flowshop scheduling problem (PFSP) is a classical problem, which usually aims to find the permutation of jobs that minimizes the total makespan, considering different machines and related restrictions. For instance, the PFSP with stochastic processing times (PFSP-ST) has been extensively studied during the last decade. Frequently, there is a product composed of several components that need to be independently processed before a deadline, when they have to be assembled and the product delivered. This problem is called distributed permutation flowshop scheduling problem with stochastic times (DPFSP-ST). In finance, investments drive the economic growth and social welfare of countries. Metaheuristics are becoming key methodologies for addressing a wide range of problems in this field. For instance, the portfolio optimization problem (POP) consists in selecting a subset of risky assets from a portfolio and setting the weight of the investment of each asset in order to minimize the portfolio's variance for a given required rate of return. Most works fail to account for stochastic returns and covariances, rendering them unrealistic in the presence of heightened uncertainty in financial markets. On the contrary, the

stochastic POP (SPOP) deal with returns and covariances modelled as random variables. Finally, computing includes a large number of procedures that may be optimized. Some examples are the parameter-fine tuning on metaheuristics, and the analysis of the effect of the number of agents and the maximum computing time on metaheuristics' performance.

Nowadays, there are two important trends in the literature on metaheuristics. The first sustains the original ideas of metaheuristics: the practical usefulness and logic of simple methodologies relying on local searches (see e.g., Gardi et al., 2014). In contrast, the second encompasses hybrid methodologies, which benefit from the advantages of each component. Indeed, simplicity is an important criteria to assess an algorithm. It facilitates the correct implementation of the algorithm by researchers and companies. However, there are many reasons why an hybrid algorithm may be required: *(i)* to obtain better results in terms of objective function values and/or computational times; and *(ii)* to deal with more realistic and richer problems. For instance, most matheuristics (Maniezzo et al., 2009) fall into the first case, solving a subproblem with an exact method. Talbi (2013) presents the combinations of metaheuristics and: *(i)* complementary metaheuristics; *(ii)* exact methods; *(iii)* constraint programming; and *(iv)* machine learning. While the author discusses interesting ideas, the number of works cited is low and the proposed classification is neither based on works nor on applications, but is a very general framework usable for all the combinations mentioned. In the context of hybrid algorithms, there is another powerful type which combines metaheuristics and simulation, so called simheuristics (Juan et al., 2015a). The framework has been developed during the last years and aims to reduce the lack of works addressing stochastic COPs (SCOPs) (Bianchi et al., 2009). Indeed, the literature has studied some of these problems but usually relying on analytical approaches making hard assumptions or complex approaches. It is crucial to address the stochasticity of these problems, since it is present in most real-life applications. For instance, traveling times greatly depends on factors which are difficult to predict (and so their effects) such as the weather, road works, accidents, etc. Similarly, processing times in scheduling can be affected by machine failures, delays in inputs delivers, etc.

This thesis studies and integrates powerful and well-known methodologies such as simulation, metaheuristics and statistical learning. It presents several works aiming to further explore, test and disseminate simheuristics. An original contribution is the development of learnheuristics combining statistical learning and metaheuristics to deal with COP with dynamic inputs (COPDIs), i.e., problems in which the inputs depend on the solution. A number of applications are analyzed, focusing on the fields previously mentioned: routing, production, finance, and computing. A graphical representation of the main methodologies and applications is shown in Figure **??**.

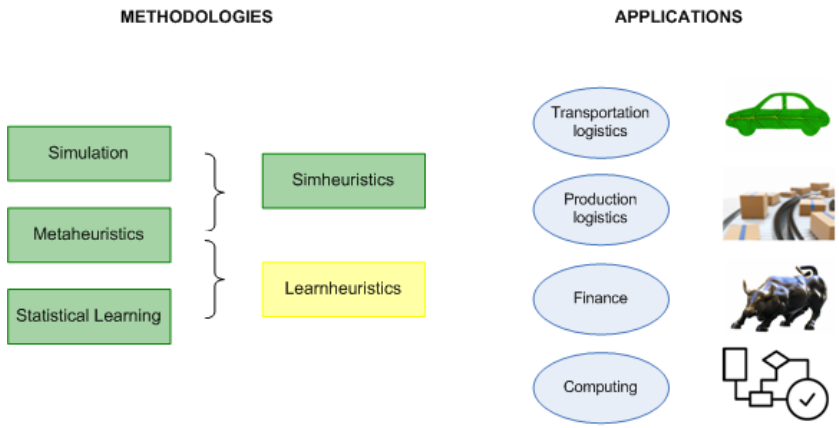


FIGURE 1.1: Scheme of key methodologies and applications of this thesis.

1.2 Main goal and original contributions

The main goal of this research is to explore the advantages of hybrid algorithms combining statistical learning and/or simulation techniques with metaheuristics. It facilitates the design of methodologies able to solve realistic and rich problems, avoiding hard assumptions usually present in the related literature. This general goal has been concreted in the following results and original contributions:

C1. An original approach combining metaheuristics and statistical learning for solving COPDIs, and a general classification and review of works combining statistical learning and metaheuristics.

C2. In the routing arena, efficient metaheuristics for solving the MDVRP-HD, the WCP, powerful simheuristics for addressing the MDVRP-SD, the WCP-SW, and the HSAVRP-SD, and a novel approach considering sustainability indicators.

C3. Regarding production, a simple simheuristic for tackling the DPFSP-ST.

C4. Related to finance, a comprehensive review of works on portfolio optimization and risk management relying on metaheuristics, an efficient metaheuristic and simheuristic for addressing the POP and the SPOP, respectively, and an analysis of the benefits due to diversification of introducing commodity futures in stock portfolios.

C5. Considering computing, a classification and an extensive review of works on the parameter fine-tuning of metaheuristics, a methodology based on clustering techniques and design of experiments (DOE) for the parameter fine-tuning of metaheuristics, and an analysis of the effects of increasing the number of agents and the computing time on the performance of well-known heuristics.

1.3 Dissertation outline

The rest of this thesis is structured in the following three blocks: methodology (chapters **??** to **??**), applications (**??** to **??**), and conclusions, future research, and contributions (**??** and **??**).

The first block focuses on the existing methodology employed and the pure methodological contributions. In particular, chapter **??** introduces metaheuristics, describing their context, reviewing the main definitions and classifications, and presenting a few popular ones. Chapter **??** is devoted to simheuristics, i.e., the integration of simulation techniques into metaheuristics-based frameworks to deal with SCOPs. Chapter **??** provides a brief definition of statistical learning, highlighting the main branches and methods. Afterwards, chapter **??** puts forward learnheuristics, which combine statistical learning and metaheuristics to address COPDIs.

The block of applications covers problems in transportation, production, finance, and computing. Chapter **??** analyzes five challenging transportation problems: the MDVRP-SD, the MDVRP-HD, the MDVRP with sustainability indicators, the WCP, and the HSAVRP-SD. Different metaheuristics/simheuristics are designed, implemented and validated for them. In the context of production, chapter **??** deals with the DPFSP-ST. Chapter **??** studies optimization problems in finance, presenting a review and focusing on the POP and the SPOP. Next, chapter **??** addresses the parameter fine-tuning of metaheuristics, and discusses issues of parallel computing.

Finally, the last block draws some conclusions, and identifies potential lines of future work in chapter **??**, while lists the publications, and presentations in chapter **??**.

Part I

METHODOLOGY

Chapter 2

Metaheuristics optimization

This chapter presents metaheuristics. After introducing them, the main classification criteria are discussed, and biased randomization techniques are presented. Later, the following ones are described: the multi-start, the iterated local search, the simulated annealing and the variable neighborhood search.
It is based on the following journal articles: Calvet et al. (2017), Calvet et al. (submitted[a]), and Calvet et al. (submitted[b]).

2.1 Introduction

OR is a well-established field with a huge and active research community. One of its main goals is to support decision-making processes in complex scenarios, i.e., providing optimal (or near-optimal) solutions to COPs defined by a given objective function and a set of realistic constraints. The number of applications is immense, e.g.: transportation and logistics, finance, production, and telecommunication systems. A noticeable part of the efforts developed by the OR community has focused on developing exact methods to find optimal solutions to a wide range of COPs. When dealing with $\mathcal{NP}$-hard COPs, this usually requires simplifying somewhat the model and/or addressing only small- and medium-sized instances to avoid incurring in prohibitive computing times. Another noticeable part of the efforts has been invested in developing heuristic and metaheuristic approaches that cannot guarantee optimality of the provided solutions but are usually more powerful in terms of the size of the instances they can solve in reasonable computing times (Talbi, 2009). Additionally, these approximated methods are quite flexible, which makes them suitable for tackling more realistic and richer models. While heuristics are simple and fast procedures based on the specific COP being addressed, metaheuristics represent a heterogeneous family of algorithms designed to solve a high number of COPs without having to deeply adapt them to each problem.

Metaheuristics have an enormous number of applications in many fields such as: engineering design, telecommunications, robotics, bioinformatics, system modeling, chemistry, and physics, among many others. A number of them are nature-inspired, include stochastic components, and have several parameters that must be fine-tuned and may interact (Boussaïd et al., 2013). As Feo and Resende (1995) state, the effectiveness of metaheuristics depends upon their ability to adapt to a particular instance problem, avoid entrapment at local optima, and exploit the structure of the problem. The authors also highlight the potential benefit of restart procedures, controlled randomization, efficient data structures, and preprocessing.

2.2 Classification

Many classification criteria have been proposed to differentiate metaheuristics (Talbi, 2009). The most important are highlighted next.

- Memory usage versus memoryless methods.
- Iterative versus greedy. An iterative metaheuristic is built from one (or more) complete solution, which is transformed at each iteration. On the other hand, a greedy algorithm starts from an empty solution and, as the execution proceeds, it is built progressively.

- Deterministic versus stochastic. A deterministic metaheuristic makes deterministic decisions; consequently, using the same initial solution will lead to the same final solution. Whereas with stochastic metaheuristics, one could obtain different solutions.
- Single-solution based search versus population-based search. Single-solution based metaheuristics transform a single solution during their execution. While in population-based metaheuristics, a set of solutions is considered. The first group is exploitation oriented, they intensify the search in local regions. In contrast, population-based metaheuristics are exploration oriented, they allow a better diversification.

Figure **??** includes some of the most popular metaheuristics (first works are cited): ant colony optimization (ACO) (Dorigo, 1992), artificial immune systems (AIS) (Farmer et al., 1986), genetic algorithms (GA) (Holland, 1962), greedy randomized adaptive search procedure (GRASP) (Feo and Resende, 1989), iterated local search (ILS) (Martin et al., 1992), particle swarm optimization (PSO) (Kennedy and Eberhart, 1995), scatter search (SS) (Glover, 1977), simulated annealing (SA) (Kirkpatrick, 1984), tabu search (TS) (Glover, 1986), and variable neighborhood search (VNS) (Mladenovic, 1995). They are grouped according to the following criteria: *(i)* single-solution versus population-based metaheuristics (SMs and PMs, respectively); *(ii)* whether they use memory; and *(iii)* whether they are nature-inspired. Circles' size is proportional to the number of Google Scholar indexed articles, from 2006 to 2015, that include the complete name of the specific metaheuristic and "metaheuristics" or "heuristics" in the article (March 15, 2016). The success of the first implementations of metaheuristics aroused the interest of journals in new versions of these methods, which increased the number of authors exploring this topic. Unfortunately, some publications add only marginal contributions to the already existing frameworks (Sörensen, 2015). In this chapter four metaheuristics will be introduced: the multi-start (MS), the ILS, the SA and the VNS. Despite being relatively simple, many state-of-the-art optimization methods are based on them.

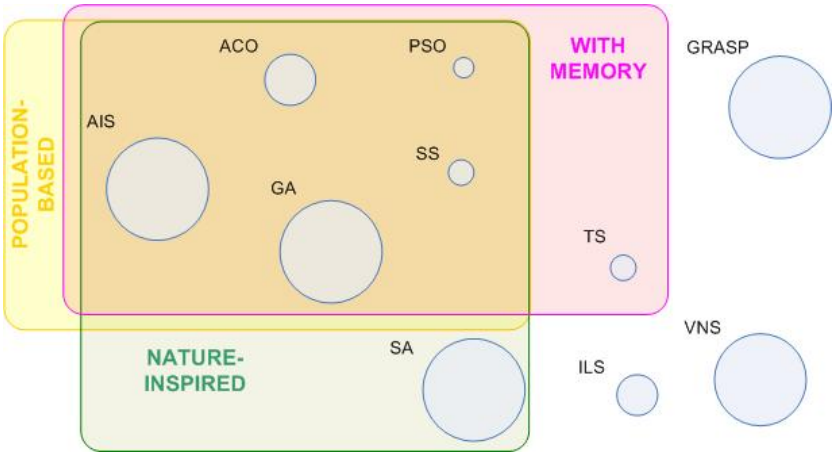


FIGURE 2.1: Main metaheuristics grouped by different criteria.

2.3 Biased randomization

In optimization, a heuristic is defined as a method for building a feasible solution based on an iterative process. At each iteration, the next movement is chosen from a list of potential candidates that has been previously sorted according to a problem-specific criterion. Pure greedy heuristics select the best next element on the short run, aiming to get a good solution at the end. This simple procedure has two drawbacks: *(i)* there is no guarantee of finding an optimal solution; and *(ii)* it is deterministic, so it always returns the same solution. Some pure greedy heuristics are the classical Clarke and Wright savings (CWS) heuristic (Clarke and Wright, 1964) in routing, and the NEH heuristic (Nawaz et al., 1983) in scheduling.

Biased randomization (Grasas et al., submitted) refers to the introduction of randomization in the construction phase and/or the neighborhood search of optimization algorithms. On the one

hand, randomization allows increasing the search area covered by selecting a candidate other than the best next option. On the other hand, biased means that the procedure is not totally random in the sense that not all the elements have the same probability of being selected. In particular, each element is assigned a given probability depending on the criterion of the procedure, the higher the element is in the list, the higher the probability.

Thus, using biased randomization leads to potentially different outputs each time the procedure is executed. Since running a simple heuristic may take a few seconds or less, implementing biased randomization may improve the solution found taking small amounts of time. The assignation of probabilities can be done by using empirical or theoretical probability distributions. There are analytical expressions that allow the quickly generation of random observations from most theoretical distributions. For this reason, these distributions constitute an efficient option. Examples of distributions widely used are the geometric, the triangular, and the log-normal. Algorithm **??** describes the steps required to implement biased randomization.

Algorithm 1 Biased randomization techniques

1: **procedure** BIASED RANDOMIZATION
2: $\mu \leftarrow$ get random number uniformly distributed in $[0, 1)$ given a specific seed
3: $\rho \leftarrow$ get random number from a distribution $PD(parameters, \mu)$
4: $l \leftarrow$ get the ρ element of the sorted list
5: **return** l
6: **end procedure**

Some recent works applying biased randomization are introduced here. The reader interested in a comprehensive review is refered to Grasas et al. (submitted). In the context of smart cities, Mazza et al. (2016) study the use of computation offloading for delegating computing-intensive tasks of smart mobile devices to the cloud. The authors develop a biased-randomized algorithm for solving this assignation problem. Related to real-life transportation activities, Dominguez et al. (2016a) focus on the two-dimensional loading VRP with clustered backhauls, where both delivery and pickup demands are composed of non-stackable items. This work presents a hybrid algorithm integrating biased-randomised versions of vehicle routing and packing heuristics within a large neighbourhood search (LNS) metaheuristic framework. Dominguez et al. (2016b) discuss the two-dimensional loading capacitated VRP with heterogeneous fleet. A MS algorithm based on biased randomization of routing and packing heuristics is proposed. Quintero-Araujo et al. (2016) solve the location routing problem (LRP), which deals with the simultaneous decisions of: *(i)* locating facilities; *(ii)* assigning customers to facilities; and *(iii)* defining routes of vehicles departing from and finishing at each facility to serve the associated customers' demands. A biased-randomized metaheuristic relying on classical heuristics is proposed.

2.4 Multi-start

The MS is a simple metaheuristic consisting of two steps that are alternated for a certain number of global iterations (Algorithm **??**). They are: *(i)* generating a solution; and *(ii)* applying a local search (i.e., a procedure to move from one solution to a better one by applying local changes). Each iteration produces a solution, usually a local optimum, and the best one is returned. Muth and Thompson (1963) and Crowston et al. (1963), both focused on scheduling, are considered the first works proposing a MS framework. However, Glover (1977) is the one introducing a local search to improve starting solutions. This author compared procedures for generating starting values for variables and for generating values perturbed from other starting points (known as restarts), and adressed controlled randomization, learning strategies, induced decomposition, and adaptive memory processes (Glover, 1986; Glover, 1989; Glover, 2000).

A comprehensive review on this metaheuristic is provided by Martí et al. (2013). This work describes the origins of the methodology, includes a classification of versions in terms of their use of memory, and introduces adaptive memory programming and the GRASP metaheuristic.

Algorithm 2 Multi-start structure

```
1: procedure MS METAHEURISTIC
2:     repeat
3:         Generate a solution s
4:         s ← LocalSearch(s)
5:     until stopping criterion is met
6: end procedure
```

2.5 Iterated local search

The ILS metaheuristic is a flexible metaheuristic that became very popular at the beginning of this century with the publication of Lourenço et al. (2010), an article that describes and analyzes its framework, reviews the literature, explains the importance of each of the elements involved and the interactions between them, and discusses its relationship with other metaheuristics. Burke et al. (2010) show that the ILS obtains the best average performance among a set of selected metaheuristic approaches in three classical COPs: bin packing, PFSP, and personnel scheduling. The authors also emphasize two main factors for its success: *(i)* an excellent balance between exploration and exploitation by "systematically combining a perturbation followed by local search"; and *(ii)* its simplicity and the reduced number of parameters required, factors that facilitate its quick implementation in practical applications.

The high level architecture of the ILS is shown in Algorithm **??**. First, an initial solution is generated, usually employing a random solution or the return of a fast heuristic. Afterwards, a local search is applied to the initial solution. It starts then an iterative process that stops when a termination condition is met; this condition can be based on time, number of iterations or solution converge, among others. Initially, the current solution is perturbed; this process may have memory, i.e., depend on the previous walk (history). It is recommended to implement a random move in a neighborhood of higher order than the one used by the local search algorithm. The following step consists in applying a local search to the perturbed solution. This solution will become the next element of the walk if it passes an acceptance test. Otherwise, one returns to the previous accepted solution. The criteria designed can be adaptive.

An important advantage of this metaheuristic is its modularity, which enables its development without problem-dependent knowledge. However, usually the most knowledge about the problem one introduces, the best performance it gets. The metaheuristic relies on the assumption that local minima are distributed in clusters.

Algorithm 3 Iterated local search structure

```
1: procedure ILS METAHEURISTIC
2:     Generate an initial solution s_0
3:     s* ← LocalSearch(s_0)
4:     repeat
5:         s' ← Perturbation(s*, history)
6:         s* ← AcceptanceCriterion(s*, s', history)
7:     until stopping criterion is met
8: end procedure
```

2.6 Simulated annealing

The SA metaheuristic is a well-established metaheuristic used in both discrete and continuous optimization. It is inspired by the process of physical annealing with solids in which a crystalline solid is heated, and then allowed to cool slowly until it achieves its most regular possible crystal lattice configuration, without crystal defects (Nikolaev and Jacobson, 2010). Similarly, the metaheuristic searches a global solution following this thermodynamic behavior. Algorithm **??** shows the steps in detail. First, a temperature change counter k is initialized to 0. Additionally, a starting temperature t_0 is set, a temperature cooling schedule is designed, and a repetition schedule M_k is stablished, which represents the number of iterations executed at each temperature t_k. An initial

solution s is created, and an outer loop is started. In this loop, a repetition counter m is set to 0. Afterwards, an inner loop starts which repeats the following steps M_k times: *(1)* a new solution s' in the neighborhood of s, $N(s)$, is built; *(2)* the difference between the objective function value of s' and s, $\Delta_{s,s'}$, is calculated; *(3)* if $\Delta_{s,s'}$ is negative (assuming a minimization problem), then s' replaces s, otherwise this replacement is performed with a probability of $exp(-\Delta_{s,s'}/t_k)$; and *(4)* m is incremented by one. After the inner loop stops, k is increased by one, which represents a decrease in the temperature. The criterion applied to decide whether s must be replaced by s' is called Metropolis acceptance criterion (Metropolis et al., 1953). It allows the algorithm to escape from local optima. An interesting overview of this metaheuristic can be found in Suman and Kumar (2006).

Algorithm 4 Simulated annealing structure

```
1:  procedure SA METAHEURISTIC
2:      Build a solution s
3:      Set temperature change counter k = 0
4:      Design a temperature cooling schedule t_k
5:      Design a repetition schedule M_k
6:      repeat
7:          Set repetition counter m = 0
8:          repeat
9:              Build a solution s' ∈ N(s)
10:             Calculate Δ_{s,s'} = f(s') − f(s)
11:             if Δ_{s,s'} ≤ 0 then s ← s'
12:             elses ← s' with probability exp(−Δ_{s,s'}/t_k)
13:             end if
14:             m ← m + 1
15:         until m = M_k
16:         k ← k + 1
17:     until stopping criterion is met
18: end procedure
```

2.7 Variable neighborhood search

The VNS was first proposed by Mladenović and Hansen (1997). Besides being a popular metaheuristic in combinatorial as well as global optimization, it has been used in a wide range of research fields such as scheduling, routing, telecommunications, biology, and artificial intelligence. For extensive reviews on applications the reader is referred to Moreno-Vega and Melián (2008) and Hansen et al. (2010). In essence, the VNS proposes systematic changes of neighborhood to find a local minimum by intensifying the search, and to escape from the associated valley by diversifying. It relies on three facts: *(i)* a local minimum with respect to one neighborhood structure is not necessarily so for another; *(ii)* a global minimum is a local minimum with respect to all possible neighborhood structures; and *(iii)* for many problems, local minima with respect to one or several neighborhoods are relatively close to each other.

Algorithm **??** shows a simple version of the VNS. Its inputs are the problem instance to solve, the number of neighborhoods considered (K), and the maximum computational time (T). Frequently, K is set to two or three, and the neighborhoods are nested. First, the variable t for measuring the time is initialized at zero. Afterwards, an initial solution is obtained and stored in *currentSol*. An outer loop sets the current neighborhood to the first one, and controls that the time-based constraint is satisfied. Inside, another loop builds and tests new solutions. Within this loop, the current solution is initially shaken (or perturbed), generating a solution from the k-th neighborhood of *currentSol*. The resulting solution is stored in *newSol*, which is then improved by means of a local search. If there is an improvement (i.e., *newSol* is preferred over *currentSol*), *newSol* is copied into *currentSol*, and the current neighborhood is set to the first. This constitutes a descendent phase aimed to find a local minimum. Otherwise, the next neighborhood is analyzed (i.e., k is set to $k + 1$). The inner loop is executed until the last neighborhood is explored (i.e., $k = K$). Finally, *currentSol* is returned.

Algorithm 5 Variable neighborhood search structure

```
1:  procedure VNS METAHEURISTIC
2:      Set K
3:      t ← 0
4:      Generate an initial solution s_0
5:      s* ← s_0
6:      repeat
7:          k ← 1
8:          while k ≤ K do
9:              s' ← Shake(s*, k)
10:             s' ← LocalSearch(s')
11:             if s' ≻ s* then
12:                 s* ← s'
13:                 k ← 1
14:             else
15:                 k ← k + 1
16:             end if
17:         end while
18:     until stopping criterion is met
19: end procedure
```

Chapter 3

Simulation and simheuristics

This chapter presents simheuristics, including a literature review and a discussion of benefits and limitations.
It is based on the following journal articles: Calvet et al. (submitted[a]), Calvet et al. (submitted[b]), Gruler et al. (2017), Gruler et al. (submitted), and Pages et al. (submitted).

3.1 Introduction

Metaheuristics constitute a powerful approach to tackle COPs, they are indeed highly popular in many research fields. However, these methodologies have been developed considering deterministic problems (i.e., ignoring stochasticity) when, in fact, real-life is plenty of uncertainty. For example, in garbage collection or stocking of vending machines, the demand is not revealed until the place is reached. Other situations in which there is unknown information are flight scheduling and capital management. Unfortunately, the oversimplification of scenarios, i.e., assuming no uncertainty, can lead to poor-quality solutions. This is the reason why there is an increasing interest in considering randomness in COPs (Bianchi et al., 2009).

Simheuristics (Juan et al., 2015a) is an approach combining metaheuristics (in a general sense, i.e., including heuristics, metaheuristics, and exact methods, among others) and simulation (Nance and Sargent, 2002; Borshchev and Filippov, 2004; Gass and Assad, 2005), specially designed to tackle COPs containing stochastic components. These components can be modeled as random variables following either theoretical or empirical probability distributions, and can be located in the objective function (for instance, random processing times) or in the set of constraints (e.g., deadlines that must be met with a given probability).

3.2 Simheuristics

A simheuristic algorithm is a particular simulation–optimization approach oriented to efficiently tackle a COP instance that typically contains stochastic components. These components can either be located in the objective function (e.g., random customers' demands) or in the set of constraints (e.g., deadlines that must be met with a given probability). In particular, the simheuristic approach is aimed at solving COPs of the form:

$$\begin{array}{ll} \text{Min} & f(s) = E[C(s)] \quad \text{or, alternatively,} \\ \text{Max} & f(s) = E[B(s)] \end{array} \tag{3.1}$$

$$\text{subject to:} \quad P(q_i(s) \geq l_i) \geq k_i \quad \forall i \in \{1, 2, \ldots, n\} \tag{3.2}$$

$$h_j(s) \leq r_j \quad \forall j \in \{1, 2, \ldots, m\} \tag{3.3}$$

$$s \in S \tag{3.4}$$

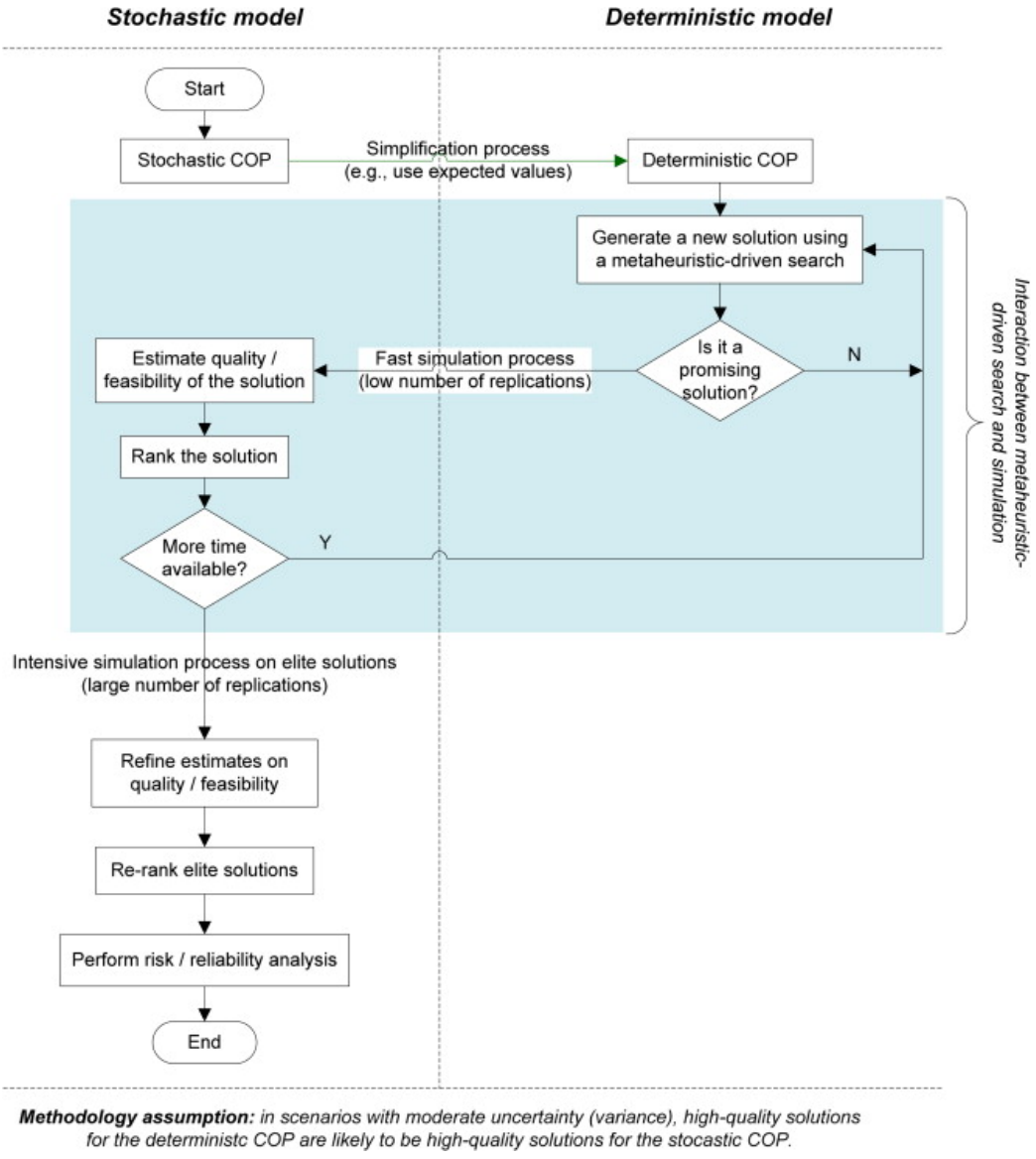
where: (*i*) S represents a discrete space of possible solutions s to the optimization problem; (*ii*) $C(s)$ represents a stochastic cost function (alternatively, $B(s)$ represents a stochastic profit or income function); (*iii*) $E[C(s)]$ represents a probabilistic measure of interest associated with the cost function (e.g., the expected value of $C(s)$); (*iv*) Equations **??** represent probabilistic constraints

related to the problem (e.g., the probability that the service quality $q(s)$ reaches a given threshold l is above a user-defined value k); and (v) Equations **??** represent typical deterministic constraints in COPs.

The simheuristic approach relies on the assumption that high-quality solutions for the deterministic version of a COP are also likely to be high-quality solutions for its corresponding stochastic version, specially in scenarios with a low or moderate uncertainty (variance).

The steps proposed to solve a SCOP instance are described next (see also Figure **??**). First, a deterministic counterpart of the instance is obtained, for example, by replacing random variables by their expected values. Afterwards, an iterative process is started. It consists in running a metaheuristic-driven algorithm to perform an efficient search inside the solution space associated with the deterministic COP first, and then estimating the quality or feasibility of each of the promising solutions when being considered as solutions of the SCOP instance. These estimations are computed using simulation techniques. The estimated values can be employed to keep a ranked list of the best solutions for the SCOP instance. Once a stopping criteria is met, more accurate estimates are obtained for the best solutions with an intensive simulation process. The advantages of this approach are numerous: it benefits from the extensive literature research related to solve deterministic COPs, and is simple, easy-to-understand and to-implement, efficient, and capable of solving realistic problems.

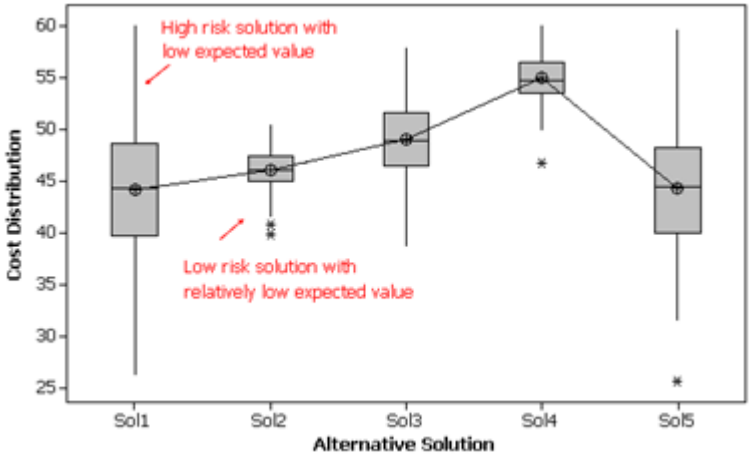
FIGURE 3.1: Scheme of the simheuristic approach. Source: Juan et al. (2015a)



Methodology assumption: *in scenarios with moderate uncertainty (variance), high-quality solutions for the determinstc COP are likely to be high-quality solutions for the stocastic COP.*

After identifying a set of high-quality solutions (those that provide the highest average performance), the risk aversion of the decision-maker can be taken into account by performing a risk analysis. It is done by studying the empirical probability distribution functions of the quality and feasibility measures computed during the intensive simulation process. An easy and fast procedure can be to provide a multiple box-plot including the solutions returned by the algorithm or only a subset of them, the non-dominated. It provides information (quartiles and outliers) about the functions. Figure **??** shows an example where there is a trade-off between solution risk and expected value.

FIGURE 3.2: Risk analysis of alternative solutions. Source: Juan et al. (2015a).



The described procedure is proposed when simulation is to be used as an evaluation function technique. Nevertheless, simheuristics can also be applied in the context of analytical model enhancement methods. The first case is the most developed so far. An example of the second can be found in Figueira et al. (2013).

Despite the fact that this approach is relatively new, there are many relevant lines of investigation being explored. In Juan et al. (2011b), the authors address the VRP with stochastic demands (VRP-SD) employing safety-stocks to reduce the route failure-risk. The single-period stochastic inventory routing problem (stochastic IRP) with stock-outs is tackled in Juan et al. (2014b). Another routing problem, the arc routing problem (ARP) with stochastic demands is studied by Gonzalez et al. (2016). Juan et al. (2014a) present a methodology to solve the PFSP. In Cabrera et al. (2014), the authors address the SCOP of determining a minimum-cost configuration of non-dedicated resources able to support a specific service while maintaining its availability over a user-defined threshold. Focusing on the home service industry, Fikar et al. (2016) propose a flexible discrete-event driven metaheuristic to deal with dynamic routing and scheduling scenarios using combined trip sharing and walking. It facilitates real-world operations, enabling rescheduling and rerouting.

3.2.1 Benefits

According to Chica et al. (submitted), the most relevant benefits of simheuristics are:

- Embracing reality by a validated simheuristic
  They allow the construction and study of valid complex system models. Indeed, new simulation paradigms can better represent the complexity of reality, and there are computational resources to run demanding simulations for addressing models that are too complicated for analytical models.

- Risk assessment of alternative solutions and sensitivity analysis
  The outputs of the simulations can be employed to generate information about the probability distribution of the quality of each solution. These outputs can also be used to perform a sensitivity analysis, which identifies the parameters having a higher effect on the model. These analyses aim to gain insights into existing or prospective systems, which could lead to better decisions and, as a consequence, to better managerial outcomes.

- System understanding and output analysis
  An innovization process (Deb et al., 2014) consists in analyzing a set of trade-off optimal or near-optimal solutions to decipher useful relationships among problem entities. Visualization methods promote design innovations. An analysis of the input/output variables space of a model may strengthen trust in the solving approach. All these analyses provide a better understanding of the behavior of the optimization and simulation models.

3.2.2 Limitations

The most important limitations are:

- Results are not expected to be optimal
  Metaheuristics do not guarantee the optimality of the solution provided. Additionally, simulation in simheuristics represents a nonlinear complex system which cannot be analytically treated. Thus, simheuristics should be used when simple and flexible methods are needed to address complex problems.

- Additional stakeholders effort is demanded to define the system
  Simheuristics require additional effort when defining the simulation system and analyzing the results.

- More computational resources are required with respect to traditional methods
  Running a simheuristic algorithm requires a high computational effort, which depends on the selected type of simulation paradigm.

Chapter 4

Statistical learning

This chapter introduces statistical learning. It summarizes the basic learning approaches describing the most popular methods and highlighting the main applications.
It is based on the following journal articles: Calvet et al. (2017), Calvet et al. (submitted[c]), and Calvet and Juan (2015).
This work has been presented at the following seminar: De Armas et al. (2016).

4.1 Introduction

The term "statistics" was originally created in the 18th century to denote the systematic collection of demographic and economic data of a state. Since then, its meaning has been increasingly broadened. Some more updated informal definitions proposed in Hahn and Doganaksoy (2012) are: *(i)* the science of learning from data; *(ii)* the theory and methods of extracting information from observational data for solving real-world problems; and *(iii)* the science of uncertainty. Statistics plays an important role in numerous economics sectors. The world of statistics[1] remarks the most visible: business and industry, health and medicine, learning, research, social sciences and natural resources.

The following subsections present the most basic learning methods classified into supervised and unsupervised learning (Hastie et al., 2009).

4.2 Supervised learning

Supervised learning encompasses a set of procedures for function approximation. Given data pairs $\{x_i, y_i\}\, \forall i = \{1, \ldots, n\}$, in a $(p+1)$-dimensional Euclidean space, there is a function $f(x_i)$ that has a domain equal to the p-dimensional input subspace, and is related to the data via a model such as $y_i = f(x_i) + \varepsilon_i$. The goal of these procedures is to obtain a useful approximation to $f(x_i)$ for all x in some region of $\mathbb{R}^p$.

Functions are estimated to describe a relation between variables, and to predict a response variable based on explanatory variables. Despite the fact that linear models with few explanatory variables are usually robust and powerful, sometimes a more complex model as a neural network can be required. In this case, the user can hardly explain the specific role of each explanatory variable in the model, i.e., the main aim of the model is to make predictions.

It is essential to validate a model before using it. This is done by splitting the dataset into two subsets: a training set, containing the data pairs used to build the model, and the test set, which is used to assess its performance. This split should be random, in order to obtain two representative subsets. Sometimes, it is required to have three subsets: a training, a validation and a test set. The validation test is employed to determine the best model between a set, or to estimate a model-specific parameter like the number of hidden units in a neural network or the parameter that determines the shrinkage penalty in a ridge regression. Often, specially with high-dimensional data ($p >> n$), it is undesirable or unfeasible to perform those splits. Then, a common procedure consists in applying cross-validation, a method for estimating the prediction error of a model based on the following steps: *(1)* generate a given number of disjointed training

[1] The world of statistics is a global network of more than 2.350 organizations worldwide committed to increasing public awareness of the power and impact of statistics, nurturing statistics as a profession, and promoting the development of probability and statistics. Its official web is: www.worldofstatistics.org.

sets from the dataset and define the corresponding test sets (all data pairs except those included in the associated training set); *(2)* for each training set, build a model and compute the prediction error with the corresponding test set; and *(3)* estimate the prediction error of the model as the average. The most popular methods of supervised learning are introduced below.

- Linear regression
  A linear regression model assumes that the relationship between the response variable and the explanatory variables is linear. This relationship is modeled through an error variable ε_i, which is an unobserved random variable. The corresponding model is:

  $$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \varepsilon_i \qquad \forall i = \{1, \ldots, n\}$$

- Tree-based methods
  Tree-based methods partition the feature space with splits, and fit a simple model for each subset of data. There are methods both for regression and classification.

- Neural networks
  Neural networks (NNs) extract linear combinations of explanatory variables as derived features, and model a response variable as a non-linear function of these features. Hidden layers are layers between the input and the output layers. Increasing the number of hidden layers and/or hidden neurons adds complexity and improves computational capacity. Having too few hidden neurons, the model might not have enough flexibility to capture the non-linearities in data. NNs tend to have many weights, which might cause problems of overfitting. Weight decay is a method of regularization to prevent it. This method adds a penalty to the error function that shrinks the weights toward zero. A tuning parameter allows weighting the penalty in the error function.

  There are several types of NN, both in supervised and unsupervised learning. They may be employed for prediction, classification, clustering, and ranking. The most popular is the feed-forward NN for prediction in supervised learning.

- Support vector machines
  The basic support vector machine (SVM) algorithm employs a training subset with a binary response variable to build a model capable of assigning new observations into one category. The algorithm constructs a hyperplane so that categories are separated by the widest margin possible. The model may include penalizations in case observations are not separable. This method may perform non-linear classification by employing the kernel trick, and mapping the explanation variables into high-dimensional spaces.

4.3 Unsupervised learning

In unsupervised learning, there is a set of observations $x_i, \forall i = \{1, \ldots, n\}$, of a random p-vector X having joint density $Pr(X)$. The aim is to infer the properties of this probability density. The most popular methods are introduced below.

- Cluster analysis
  Cluster analysis consists in grouping a collection of observations into subsets or clusters, such that those within each cluster are more closely related to one another than those assigned to different clusters. The grouping is based on the definition of similarity / dissimilarity between two observations. The dissimilarity between two clusters is defined by the linkage. The most used types are: complete, single, average, centroid, and medoid. The result of the clustering highly depends on the linkage selected.

  Hierarchical clustering is an approach that aims to build a hierarchy of clusters. The related strategies fall into two types: agglomerative and divisive. The first group starts with each observation being considered a cluster, and pairs of clusters are merged as one moves up the hierarchy. On the other hand, in the divisive approach all observations start in a cluster, and splits are done recursively as one moves down the hierarchy.

- Self-organizing maps
  A self-organizing map (SOM) is a type of NN. It produces a low-dimensional and discretized representation of the input space of the dataset, which is called map. An important characteristic of this NN is that preserves the topological properties of the input space by employing a neighbourhood function. A SOM represents a mapping from the input space to the map space with a lower dimension.

- Principal components analysis
  Principal components analysis (PCA) consists in transforming a dataset to a new coordinate system by applying orthogonal linear transformation in such a way that the greatest variance by some projection of the data comes to lie on the first coordinate or first principal component, the second greatest variance on the second coordinate, and so on. The number of principal components is less or equal to the number of original variables. PCA is usually performed by eigenvalue decomposition of the covariance or correlation matrix of the original data.

There is a third method so-called semi-supervised learning (Chapelle et al., 2006), which employs both labeled and unlabeled observations (i.e., not all have associated an output value). It is extremely powerful for problems in which large amounts of unlabelled observations are available, and only a few of them can be manually labelled. Typical examples are visual object recognition, where milions of untagged images are publicly available, or natural language processing.

Chapter 5

Learnheuristics: statistical learning and metaheuristics

This chapter reviews works combining statistical learning and metaheuristics, and proposes a hybrid approach for tackling optimization problems with dynamic inputs. It is based on the following journal articles: Calvet et al. (2017).
This work has been presented at the following seminar: Calvet (2015a) and Calvet (2015b).

5.1 Introduction

The OR community shows a growing interest in coping with increasingly challenging COPs, such as SCOPs and dynamic COPs (in which some of the problem inputs evolve over time). This might be due to several factors, including: *(i)* the rich characteristics of real-life problems frequently faced by modern companies in sectors such as logistics and transportation (Caceres et al., 2014); *(ii)* the technological development; *(iii)* the availability of vast amounts of Internet-based data; and *(iv)* a shift to a more data-driven culture. During the last years, hybrid approaches have been extensively employed due to their success when dealing with realistic problems, among others: those combining different metaheuristics (Talbi, 2013), matheuristics (i.e., metaheuristics combined with mathematical programming) (Maniezzo et al., 2009), and simheuristics.

The hybridization of metaheuristics with statistical learning is an emerging research field. In this context, the main contributions of this chapter are: *(i)* providing a classification on works combining metaheuristics with statistical learning; and *(ii)* proposing a novel 'learnheuristic' framework, combining a heuristic-based constructive procedure with statistical learning, to deal with COPDIs. In these problems, the inputs are deterministic (i.e., non-stochastic) but, instead of being fixed in advance, they vary according to the structure of the solution (i.e., they change as the solution is being constructed following a heuristic-based iterative process). In this sense, these COPDIs represent an extension of the classical deterministic COPs in which all inputs are given in advance and are immutable. An example of such a COPDI is given next for illustrative purposes. Suppose there is a set of heterogeneous radio access technologies (RATs) that provide pay-per-use services to a group of users. Each user has to be assigned to just one RAT, and each RAT can serve only a limited number of users. The goal is to maximize the total benefit, which depends on the customers' demands. Several scenarios may be described based on the nature of the demands (Figure **??**): *(i)* they are deterministic and static; *(ii)* they contain some degree of uncertainty but can be modeled as random variables or using fuzzy techniques; and *(iii)* they are dynamic in the sense that they depend on the solution characteristics (e.g., the number of users connected to the same RAT, which has an effect on the service quality and, therefore, on the customers' demands). While the first case corresponds to a classical deterministic COP, the second case introduces a level of uncertainty that usually requires the use of stochastic programming, simulation-optimization, or fuzzy methods. Focusing on the third case, the learnheuristic algorithms have a learning mechanism that updates the input values as the solution is iteratively constructed using the heuristic logic (Calvet et al., 2016d).

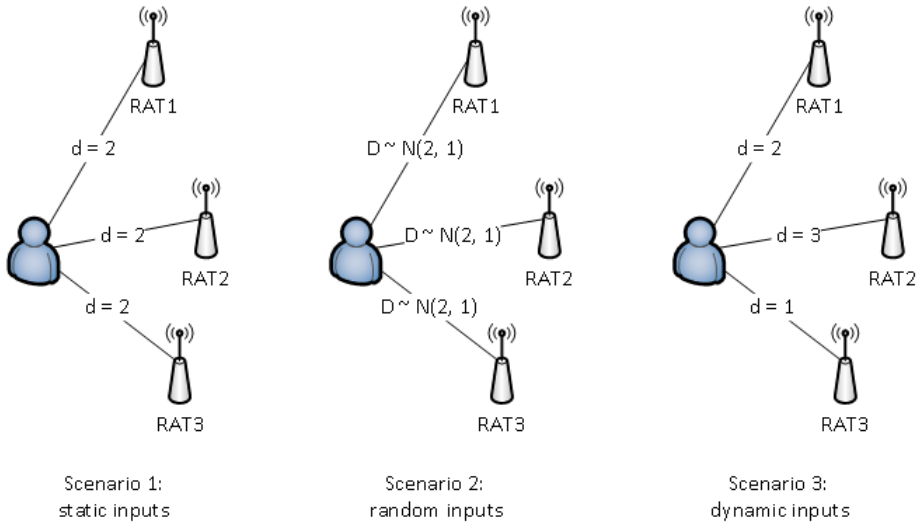


FIGURE 5.1: Type of problem according to the nature of the inputs.

5.2 Related reviews

The existing literature analyzing the hybridization of metaheuristics and statistical learning may be mainly divided into two groups: works were statistical learning is employed to enhance metaheuristics, and those in which metaheuristics are used to improve the performance of statistical learning techniques.

Regarding the first group, there are several works providing overviews. For instance, the emergence of hybrid metaheuristics is studied in Talbi (2013), which includes the combination of metaheuristics and: *(i)* complementary metaheuristics; *(ii)* exact methods; *(iii)* constraint programming; or *(iv)* statistical learning. The author distinguishes between low-level hybridizations, in which a given internal function of a metaheuristic is replaced by another optimization method, and high-level hybridizations, where the different optimization methods are self-contained. In a second phase, these algorithms can be further classified into relay (where techniques are applied one after another) or teamwork hybridization. Jourdan et al. (2006) describe applications of data mining techniques to help metaheuristics. A survey on the integration of statistical learning in evolutionary computation can be found in Zhang et al. (2011). The work presented in Corne et al. (2012) gathers the synergies between OR and data mining, highlighting three benefits of employing data mining in OR: *(i)* increasing the quality of the results; *(ii)* speeding up algorithms; and *(iii)* selecting an algorithm based on instance properties. This chapter builds on the classification in Jourdan et al. (2006) and extends it by proposing more categories and analyzing a higher number of works. Works are classified into specifically-located hybridizations (where statistical learning is applied in a specific procedure) and global hybridizations (in which statistical learning has a higher effect on the metaheuristic design).

Similarly, there are a few reviews on metaheuristics used to improve the performance of statistical learning techniques. For instance, Freitas (2008) focuses on two evolutionary algorithms (EAs), namely GAs and genetic programming (GP), and discusses their application to discovery of classification rules, clustering, attribute selection and attribute construction. Corne et al. (2012) analyze the role of OR in data mining discussing the relevance of exact methods, heuristics and metaheuristics in supervised classification, unsupervised classification, rule mining and feature selection. More recently, Dhaenens and Jourdan (2016) provides an overview of the use of optimization in Big Data focusing on metaheuristics. The book introduces the role of metaheuristics in clustering, association rules, classification, and feature selection in classification. Building on these reviews, here the literature works are arranged into the following categories: classification, regression, clustering, and rule mining.

5.3 Statistical learning for enhancing metaheuristics

This section describes works employing statistical learning for enhancing metaheuristics. They are first grouped into specifically-located hybridizations and global hybridizations.

5.3.1 Specifically-located hybridizations

The ***fine-tuning of metaheuristic parameters*** is known to have a significant effect on the algorithm performance. However, this issue is not always properly addressed and many researchers still continue selecting parameter values by performing exhaustive testing or copying values recommended for similar instances or problems. Basically, there are three approaches:

1. *Parameter control strategies* (De Jong, 2007) apply a dynamic fine-tuning of the parameters by controlling and adapting the parameter values during the solving of an instance. The main types of control are: *(i)* deterministic, which modifies the parameter values by some deterministic rule; and *(ii)* adaptive, which employs feedback from the search. For instance, there are works relying on fuzzy logic (Jeong et al., 2009), SVMs (Zennaki and Ech-Cherif, 2010), and linear and SVM regression (Lessmann et al., 2011).

2. *Parameter tuning strategies* assume that the algorithms are robust enough to provide good results for a set of instances of the same problem with a fixed set of parameter values. Frequently, researchers focus on a subset of the instances and analyze their fitness landscapes. Popular techniques are: response surface (Gunawan et al., 2013), logistic regression (Ramos et al., 2005), and tree-based regression (Bartz-Beielstein et al., 2004).

3. *Instance-specific parameter tuning strategies* present characteristics from the previous approaches. While the parameter values are constant as in the second approach, they are specific for each instance as in the first one. These strategies employ a learning mechanism able to return recommended sets of parameter values given a number of instance features. Techniques employed are: Bayesian networks (Pavón et al., 2009), case-based reasoning (CBR) (Pereira et al., 2013), fuzzy logic (Ries et al., 2012), linear regression (Caserta and Rico, 2009), and NNs (Dobslaw, 2010).

Typically, metaheuristics generate their ***initial solutions*** randomly, using design of experiments (Leung and Wang, 2001), or via a fast heuristic. There are also works employing statistical learning techniques. For instance, some of them apply CBR to initialize GAs (Ramsey and Grefenstette, 1993; Louis and McDonnell, 2004; Li et al., 2011b), while others explore the use of Hopfield NNs (Yalcinoz and Altun, 2001). In De Lima et al. (2008) the authors suggest using the Q-learning algorithm in the constructive phase of a GRASP and a reactive GRASP metaheuristics. In this line, the hybridization of data mining and the GRASP metaheuristic is discussed in Santos et al. (2008).

In real-life applications it is common to find objective functions and constraints that are computationally expensive to ***evaluate*** (Lim et al., 2010; Tenne and Goh, 2010). In these cases, it is required to build an approximation model to assess solutions employing polynomial regression (Zhou et al., 2005), NNs (Adra et al., 2005; Pathak et al., 2008), SVMs (Yang et al., 2009), Markov fitness models (Brownlee et al., 2010), kriging (Díaz-Manríquez et al., 2011) or radial basis functions (Regis, 2014), for example. Some authors combine their use with that of real objective functions (Rasheed and Hirsh, 2000; Zhou and Zhang, 2010). A survey on model approximation in evolutionary computation may be found in Jin (2005). Another option to reduce evaluation costs is to evaluate only representative solutions. Following this idea, Yoo and Cho (2004) apply fuzzy clustering, while Jin and Sendhoff (2004) use clustering techniques and NNs ensembles.

Regarding ***population management***, many authors attempt to extract information from solutions already visited and employ it to build new ones, aiming to explore more promising search spaces. A number of works rely on the Apriori algorithm (to identify interesting subsolutions) (Dalboni et al., 2003; Santos et al., 2005; Ribeiro et al., 2006; Santos et al., 2006) or on CBR (Louis, 2003). Another important issue in PMs is the population diversity, since maintaining it may lead to better performances. The most common technique for promoting diversity is clustering analysis. In Streichert et al. (2003), for instance, individuals in a GA are separated in different sub-populations based on their features and only those in the same cluster compete for survival. The selection operator is applied independently to each cluster.

The search of a metaheuristic may be improved by introducing knowledge in ***operators*** such as mutation or crossover operators in PMs. For example, Michalski (2000) design a class of evolutionary computation processes called learnable evolution model (LEM), which uses symbolic learning methods to create rules that explain why certain individuals are superior to others. These rules are then employed to create new populations by avoiding past failures, using recommendations or generating variants. In Jourdan et al. (2005), this class is extended to address multi-objective problems.

Some statistical learning techniques have been used as ***local searches***. For instance, Gaspar-Cunha and Vieira (2004) employ a multi-objective EA (MOEA) combined with an inverse NN. The authors test their approach on a set of benchmark bi-objective functions. A similar approach is suggested in Adra et al. (2005) to be applied to an aircraft control system design application.

5.3.2 Global hybridizations

A few works have attempted to ***reduce the search space*** in order to make more effective and efficient searches. Statistical learning techniques used are: clustering techniques (Hu and Huang, 2004; Senjyu et al., 2005; Barreto et al., 2007; Adibi and Shahrabi, 2013), NNs (ChangYoon and Way, 2001; Marim et al., 2003) and PCA (Auger and Hansen, 2005).

The ***algorithm selection problem*** (ASP) aims to predict the algorithm from a portfolio that will perform best, employing a given set of instance features. Its framework was proposed by Rice (1976), where it was applied to partial differential equation solvers. More recently, Smith-Miles (2009) presents it in the context of optimization algorithms. Kanda et al. (2011) design an approach to select the best optimization method for solving a given travelling salesman problem (TSP) instance. Initially, 14 TSP properties and the performance values obtained with each metaheuristic analyzed (GRASP, TS, SA and GA) are stored. Then, a rank of metaheuristics is determined by using a multi-layer perceptron network. Several network architectures are assessed. In Smith-Miles et al. (2014), the authors construct a methodology to compare the strengths and weaknesses of a set of optimization algorithms. First, the instance space is generated. This step includes selecting a subset of features providing a good separation of easy and hard instances. Afterwards, classification techniques are used to identify the regions where an algorithm performs well or poorly. The experiment is carried out with 8 algorithms for solving the graph coloring problem.

According to Burke et al. (2010), ***hyperheuristics*** may be described as search methods or learning mechanisms for selecting or generating heuristics to solve computational search problems. Typically, these methods do not aim to obtain better results than problem-specific metaheuristics, but to be able to automate the design of heuristic methods and/or deal with a wide range of problems. The authors propose a classification taking into account the following dimensions: *(i)* the nature of the heuristic search space (either heuristic selection or generation); and *(ii)* the feedback, since hyperheuristics may learn (following online or offline learning strategies) or not. A comprehensive survey on hyper-heuristics may be found in Burke et al. (2013). Reinforcement learning is highly popular in methodologies selecting heuristics employing an online learning strategy (e.g., see Berberoğlu and Uyar, 2010). In Asta and Ozcan (2014) an apprenticeship learning hyperheuristic is proposed for vehicle routing. Taking a state-of-the-art hyperheuristic as an expert, the authors follow a learning approach that yields various classifiers, which capture different actions that the expert performs during the search. While this approach relies on a C4.5 algorithm, in Tyasnurita et al. (2015) it is improved by using a multilayer perceptron.

During the last decades, a new trend in optimization has emerged based on ***cooperative strategies***. It consists in combining several algorithms/agents to produce a hybrid strategy in which they cooperate in parallel or sequentially. Communication among them can be either many-to-many (direct) or memory-based (indirect). Agents may share partial or complete solutions and models, among others. It is broadly accepted that strategies based on agents with unrestricted access to shared information may experiment premature convergence. Commonly, there is an agent that coordinates the search of the others, organizing the communication. For example, Cadenas et al. (2009) develop a centralized hybrid metaheuristic cooperative strategy, where knowledge is incorporated into the coordinator agent through fuzzy rules. These rules have been defined from a knowledge extraction process applied to the results obtained by each metaheuristic. The strategy is tested on the knapsack problem, employing a TS, a SA, and a GA. In Martin et al. (2016), a cooperative strategy relying on different metaheuristic / local search combinations is put forward.

The architecture makes use of two types of agents: the launcher and the metaheuristic agent. Each metaheuristic agent continuously adapts itself according to a cooperation protocol based on reinforcement learning and pattern matching. This proposal is tested on the PFSP and the capacitated VRP (CVRP).

There are several ***new metaheuristics*** based on learning procedures. Most rely on the fact that a set of pseudo-optimal solutions may be considered a sample drawn from an unknown probability distribution. This distribution may be estimated by employing a selected set of promising solutions and used to generate new solutions. A review of these metaheuristics, called estimation of distribution algorithms (EDAs), can be found in Pelikan et al. (2002). These metaheuristics have been employed in a wide range of fields such as routing (Euchi, 2014; Wang et al., 2015), scheduling (Ceberio et al., 2012), and nutrition (Gumustekin et al., 2014).

5.4 Using metaheuristics to improve statistical learning

Metaheuristics have been extensively employed to improve statistical learning tasks. Briefly, some of the most successful approaches are reviewed in the supervised learning topic, both in classification and regression, and in the unsupervised learning topic, including clustering and rule mining.

In ***classification***, metaheuristics have been mainly applied for feature selection, feature extraction and parameter fine-tuning. Escalante et al. (2016) suggest that the bags of visual words algorithm could be improved when non linear combinations of weighted features obtained with GP are considered. The approach is successfully applied to the object recognition field, learning both the weights of each visual word (feature) and the non linear combination of them. Fernández-Caballero et al. (2010) present a multi-classification algorithm relying on multi-layer perceptron NN models. In order to obtain high levels of sensitivity and accuracy (which may be conflicting measures), a Pareto-based multi-objective optimization methodology based on a memetic EA is proposed.

In ***regression***, the use of statistical learning is typically related to the training of complex regression models. Neuroevolution is an emergent field which employs EAs to train NNs. Thus, Yao (1999) provides a literature review on elements evolved: connection weights, architectures, learning rules, and input features. In Stanley and Miikkulainen (2002), the authors develop the neuroevolution of augmenting topologies (NEAT) method, which evolves topologies and weights at the same time. Carvalho et al. (2011) present a methodology to find the best architecture of a NN using metaheuristics. The authors tested the following ones: generalized extremal optimization, VNS, SA, and canonical GA.

Regarding ***clustering***, centroid models are based on an $\mathcal{NP}$-hard optimization problem (thus, only approximated solving methods such as metaheuristics may be employed). For instance, Shelokar et al. (2004) use ACO to cluster objects, obtaining faster results in terms of the number of objective functions evaluations. Gene clustering is performed in Banu and Andrews (2015), where a comparative study is presented based on the following metaheuristics: GA, PSO, cuckoo search and levy flight cuckoo search. More recently, Ferone et al. (2016) present a GRASP metaheuristic for biclustering of gene expression data. The reader can find more details in the applications of metaheuristics to unsupervised learning in these surveys: Hruschka et al. (2009) and Kurada et al. (2013).

Related to ***rule mining***, Freitas (2002) presents data mining tasks and paradigms, and describes the application of GAs and GP for rule discovery, and EAs for generating fuzzy rules. After modeling association rules discovery as an optimization problem, Khabzaoui et al. (2004) explore the use of a GA to obtain associations between genes from DNA microarray data. Noticing that most approaches tend to seek only frequent rules, Khabzaoui et al. (2008) propose a multi-objective approach combining a GA and exact methods to discover interesting rules in large search spaces.

5.5 Learnheuristics

The learnheuristic framework aims at solving COPs in which the model inputs (either located in the objective function or in the set of constraints) are not fixed in advance. Instead, these inputs might vary in a predictable way according to the current status of the partially-built solution at

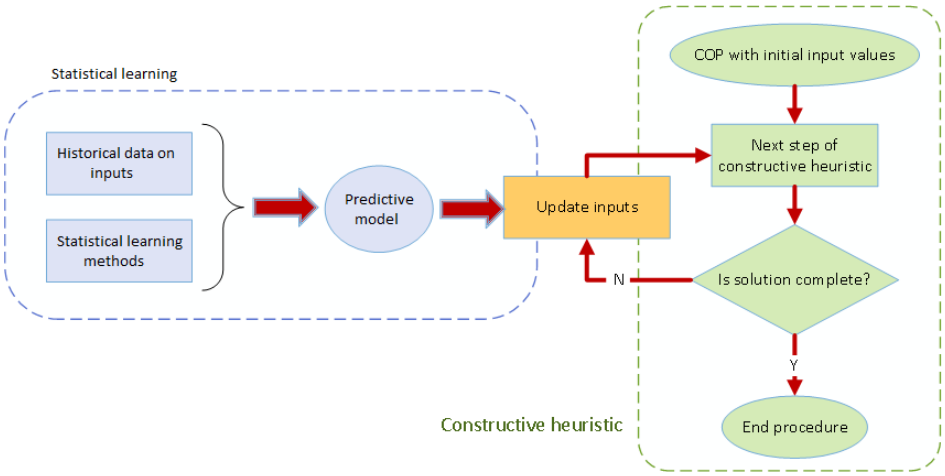


FIGURE 5.2: Basic scheme of a learnheuristic framework.

each iteration of the constructive heuristic. More formally, these problems might be represented as follows:

$$\begin{array}{ll} \text{Min} & C(s, I_{OF}(s)) \quad \text{or, alternatively,} \\ \text{Max} & B(s, I_{OF}(s)) \end{array} \tag{5.1}$$

$$\text{subject to:} \quad Q_j(s, I_C(s)) \leq r_j \quad \forall j \in J \tag{5.2}$$

$$s \in S \tag{5.3}$$

where: *(i)* S refers to a discrete space of possible solutions s; *(ii)* $C(s)$ represents a cost function (alternatively, $B(s)$ represents a benefits function); *(iii)* $I_{OF}(s)$ and $I_C(s)$ refer to inputs in the objective function or the constraints, respectively; and *(iv)* Equations **??** represent a set of constraints. Thus, the aim of this type of problems is to minimize a function of costs (or, alternatively, maximize a function of benefits) subject to a number of constraints. The novel characteristic is that inputs in the objective function and/or the constraints may depend on the solution structure, which makes them to be dynamic as the partially-built solution evolves, and not fixed in advance.

In order to deal with these COPDIs, the use of a learnheuristic framework is proposed as explained next (see Figure **??**). Initially, historical data on different system states (e.g., different assignments of users to RATs) and their associated inputs (e.g., users' demands observed for the corresponding assignments) are employed to generate predictive models. Then, these models are iteratively used during the heuristic-based constructive process in order to obtain updated estimates of the problem inputs (e.g., users' demands) as the structure of the solution (e.g., users-to-RAT assignment map) varies. Eventually, once the construction process is finished, a complete solution is generated. Without the use of the learning mechanism, the heuristic-based construction process will not take into account the variations in the inputs due to changes in the solution structure, which will lead to sub-optimal solutions.

Algorithm **??** contains a more detailed description of the basic learnheuristic framework. Notice that the main loop iterates over a list of elements that are provided by the constructive heuristic (e.g., next user-to-RAT assignment). At each iteration, the algorithm evaluates the current status of the partially-built solution, makes use of the predictive model to update the problem inputs according to this status, and follows the heuristic logic to take another solution-building step based on the new problem inputs.

As any other heuristic procedure, the aforementioned learnheuristic approach can be integrated into a more complex metaheuristic framework. For instance, it can be easily integrated into MS, GRASP, or ILS frameworks. In order to do so, the learnheuristic algorithm may be combined with biased-randomization strategies as the ones proposed in Juan et al. (2011b).

Algorithm 6 Learnheuristic algorithm.

Learnheuristics(*historicalData*, *inputs*)
% *historicalData*: historical data on different system states and their associated inputs
% *inputs*: problem instance
  model ←buildPredictiveModel(*historicalData*)
  sol ← *empty*
  while (*sol* is not completely built) **do** % *iterative learning-heuristic process*
    inputs ←updateInputs(*model*, *inputs*, *sol*)
    sol ← nextHeuristicStep(*inputs*, *sol*)
  end while
  return *sol*

Algorithm 7 Learnheuristic algorithm based on the MS metaheuristic.

Multi-start(*historicalData*, *inputs*, *distribution*, *maxTime*)
% *distribution*: probability distribution and parameters for the biased-randomization process
% *maxTime*: maximum computing time allowed
  initInputs ← *inputs* % *copy of initial inputs*
  elapsedTime ← 0
  initTime ← *currentTime*
  bestSol ← biasedRandLearnheuristic(*historicalData*, *inputs*, *distribution*)
  inputs ← *initInputs* % *reset inputs*
  while (*elapsedTime* ≤ *maxTime*) **do**
    newSol ← biasedRandLearnheuristic(*historicalData*, *inputs*, *distribution*)
    newSol ← localSearch(*newSol*)
    if (cost(*newSol*) ≤ cost(*bestSol*)) **then**
      bestSol ← *newSol*
    end if
    inputs ← *initInputs* % *reset inputs*
    elapsedTime ← *currentTime*−*initTime*
  end while
  return *bestSol*

5.6 Applications

This section provides a series of examples in which the use of learnheuristics might facilitate the solving process of more realistic and rich models.

- ***Transportation:*** In the transportation area and, in particular, in VRPs and ARPs, inputs such as the customers' demands might be dynamic in the sense that they might depend upon the delivery time and whether or not certain time-windows are satisfied. It is a function of the solution structure, e.g., the order in which the customers are visited, the number and type of vehicles employed, etc. Similarly, the traveling times, which affect the distribution cost, might also be dynamic and dependent on the solution structure, specially in large cities where traffic jams occur frequently.
- ***Logistics:*** As discussed in Calvet et al. (2016d), the assignment of customers to certain distribution centers might have a significant effect on the customers' willingness to spend (i.e., on their demands). Therefore, in realistic facility location problems and similar ones, modelers might have to face dynamic inputs influenced by the shape of the solution (i.e., which facilities are open and how customers are assigned to them).
- ***Production:*** In scheduling problems, for instance, processing times of jobs into machines might not be fixed but, instead, they may be a function of the order in which they are processed by the machine (e.g., due to 'fatigue' issues or to breaks). A similar situation can happen in project scheduling, where some working teams might be more efficient than others and assigning them to a given sub-project could cause the delay of others.
- ***Finance:*** In problems such as portfolio optimization, the covariance matrix that measures the risk associated with each pair of assets could also be a function of the current portfolio structure (i.e., which other assets are already included and which percentage of investment has been assigned to each of them). Likewise, the expected return for each asset might depend on the current composition of the portfolio. This dynamic behavior of the inputs can be extended to different risk-management problems which include some sort of portfolio optimization.

5.7 Example

This section describes a simple numerical experiment based on a VRP in which each customer's demand will depend on the order in which the customer is visited. For each customer, its initial demand value is an upper-bound of the real demand. In other words, this value will be valid only if the customer is visited by a vehicle as the first stop in its route. Then, as the position in which the customer is visited increases, the customer's demand will be reduced. Therefore, the use of a constructive heuristic to solve the VRP considering the initial demands as fixed inputs will overestimate the real demands. This, in turn, will lead to higher costs, since the number of routes employed to satisfy the real demands will be higher than necessary. Likewise, vehicles will be carrying more load than strictly required. On the contrary, if the real customers' demands are predicted based on their position inside a route, then each route might be able to cover additional customers and the total distance-based costs will be reduced.

In order to compare both cases, the CWS heuristic have been applied to a random instance belonging to the well known benchmarks for the VRP, particularly to the instance P-n70-k10 (http://neo.lcc.uma.es/vrp/wp-content/data/instances/Augerat/P-VRP.zip). On the one hand, fixed demands are considered, i.e., the original demands are used to obtain the solution through the heuristic in the standard way. On the other hand, a predictive model is created to calculate dynamic demands in order to apply a learnheuristic algorithm. In this case, for illustrative purposes, the following linear regression model has been considered:

$$d = \max\{k_1 \cdot d_0,\ d_0 - k_2 \cdot d_0 \cdot (p-1)\} \tag{5.4}$$

where d is the predicted demand of a given customer, d_0 is the initial demand of the same customer, k_1 and $k_2 \in (0, 1)$, and p is the position order in the route of the aforementioned customer. In particular, k_1 and k_2 are set to 0.20 and 0.05, respectively.

The regression model aims at predicting a customer's demand taking into account the position in which the customer is served in the route, so that the demand decreases as the position increases or until a certain demand lower-bound is reached. Thus, each time the heuristic performs a step, incorporating a new customer in a route or moving a customer from one route to another, the customer's demand is predicted and updated according to its new position in the corresponding route. As mentioned before, the total demand in a route is limited by the capacity of the vehicle. Therefore, this prediction affects the next steps that can be performed.

When fixed demands are considered, the best solution the constructive heuristic is able to obtain has an associated cost of 896.86, and it involves 11 routes. However, if demands are predicted taking into account the delivery order, the same heuristic obtains a solution with 8 routes and a cost of 791.26. Therefore, the savings might be noticeable when dynamic demands are considered.

Part II

APPLICATIONS

Chapter 6

Applications in transportation

This chapter studies several rich and realistic routing problems. It proposes different hybrid algorithms relying on metaheuristics, Monte Carlo simulation and regression models.
It is based on the following journal articles: Calvet et al. (submitted[a]), Calvet et al. (2016d), Gruler et al. (2017), Reyes et al. (submitted), and Calvet et al. (2016a). This work has been presented at the following conferences: Juan et al. (2015d), Juan et al. (2015b), Calvet et al. (2015a), and Calvet et al. (2015b).

6.1 Introduction

As a consequence of the growing flows of freight, the development of efficient and sustainable transportation and logistics activities has become a priority for Europe (European Union, 2011a; European Union, 2011b). The globalization, the growth of population, their purchase capacity and the efficiency of production systems are the main reasons of the increase in this sector. According to Eurostat (2015), emissions of greenhouses gases, air pollutants and noise from transport have significant impacts on the climate, the environment and human health. The need of flexible and efficient optimization tools affects both the public and the private sectors, since a huge number of companies daily address problems related to the transportation of people and/or goods.

In this context, the design of intelligent approaches is key for: *(i)* company competitiveness; *(ii)* good functioning of the labour market; *(iii)* cohesion within and between regions; *(iv)* reduction in the fossil energy importation, by the decrease of the consumption of petroleum derivatives; *(v)* decrement in the pollution, which improves people health; and *(vi)* reduction in traffic accidents.

The VRP is the most classical and simple formulation employed to describe logistic problems dealing with physical distribution. It constitutes the most popular research line in combinatorial optimization because of its practical relevance and its scientific interest due to its $\mathcal{NP}$-hardness. This chapter studies five realistic and rich extensions of the VRP (RVRP): the MDVRP-SD, the MDVRP-HD, the sustainable MDVRP, the WCP and the HSAVRP.

The MDVRP is a two-stage decision process, since assignation and routing issues are often interrelated, i.e., the assignment map may affect the quality of the posterior routing. Montoya-Torres et al. (2015) highlight a noticeable growing interest in the MDVRP during the last decade, with over 103 publications between 2006 and 2014. In the stochastic and capacitated MDVRP, a set of customers with random demands must be served by a fleet of homogeneous capacitated vehicles departing from one among several capacitated depots. The main goal is to determine the set of routes that minimizes the expected total routing cost, including recursive actions, subject to a number of capacity-related constraints. The problem has numerous applications in real-life, e.g.: garbage collection, gas distribution, stocking of vending machines, and other similar activities in which the specific amount of goods to leave or pick up is not known until the place is reached. Despite its relevance, there are few works on the stochastic MDVRP and, to the best of our knowledge, Calvet et al. (submitted[a]) is the first one addressing the stochastic and capacitated MDVRP.

Most works on the MDVRP has focused on minimizing distance-based distribution costs. However, no attention has been given so far to potential variations in demands due to the fitness of the customer-depot mapping in the case of heterogeneous depots. In the MDVRP-HD, the depots are heterogeneous in terms of their commercial offer, and customers show different willingness to consume depending on how well the assigned depot fits their preferences. As a consequence, market-segmentation strategies need to be considered to increase sales and total income while accounting for the distribution costs.

The increasing social concern is compelling companies to change purely commercial objectives in order to consider sustainability. This new vision seeks to compensate the negative impacts of transport activities without neglecting economic profits. In this context, it is essential to introduce approaches considering the impacts of the three pillars of sustainability for routing problems such as the MDVRP. For instance, travelling time and distance are related to economical impacts, carbon emissions to environmental impacts, and risk of accidents to social impacts.

In the face of rising population densities in urban areas around the world, a large number of cities are currently reorganizing their municipal responsibilities (Nations, 2015). As a consequence, the WCP is of high practical importance, especially in the context of smart city initiatives (Neirotti et al., 2014). On the one hand, uncollected garbage can lead to pollution of the environment and health-issues, while noise and road congestions through extensive use of waste collection vehicles decrease urban living standards. On the other hand, waste collection represents up to two thirds of operational waste management costs (Malakahmad et al., 2014; Son, 2014; Tavares et al., 2009). As waste generation and travel times of vehicles cannot be predicted with full certainty, there is a need for fast and risk-aware solutions of high quality which are able to take stochastic input variables into account.

The HSAVRP-SD considers a heterogeneous fleet. This diversity usually comes from two facts: different customers and locations may require different types of vehicle (e.g., due to narrow roads, available parking spaces, and vehicle weight restrictions), and the vehicle acquisitions may be made in different times and places. In addition, this problem describes a scenario where some customers cannot be accessed with all types of vehicle, which is known as site-dependency. Regarding the cost matrix, the classical assumption about its symmetry is relaxed, since there can be cost differences associated to the direction of a route (for instance, differences between driving uphill or downhill in mountainous regions). Moreover, the HSAVRP-SD also accounts for uncertainty in demands. It has several real-life applications such as the fuel oil distribution, which can be associated to petrol station replenishment or to the delivery of domestic heating oil. In these cases, the exact demand is not known until the time of the delivery, and cost between nodes (based on energy consumption) is asymmetric due to the presence of important road grades. The optimization of domestic heating oil distribution has been less studied, even though the high dependence on heating oil of some isolated regions. It could be of particular interest in mountainous regions where in absence of a gas pipeline, domestic oil is frequently the predominant fuel for heating.

6.2 Literature review

This section reviews related works. First, it focuses on the MDVRP and the WCP. Afterwards, routing works considering stochasticity and sustainability are introduced. More references can be found in the articles introduced in this chapter.

6.2.1 The MDVRP

As commented before, the MDVRP has been intensively studied in the last decades. Table **??** summarizes the information of the main related works.

6.2.2 The WCP

Probably the first work to address municipal solid waste collection is Beltrami and Bodin (1974). Since then, various solution techniques for different variants of the WCP have been proposed. While some works formulating the WCP as an ARP can be found (Ghiani et al., 2005; Bautista et al., 2008), the following discussion refers to recent publications using VRP formulations. More extensive literature reviews are provided by Beliën et al. (2014), Ghiani et al. (2014), and Han and Ponce-Cueto (2015).

Most works on the WCP focus on case studies with some problem extension. For example, Baptista et al. (2002) elaborate an extension of the Christofides and Beasley heuristic for the multi-period WCP (Christofides and Beasley, 1984), modeled as a periodic VRP to combine vehicle scheduling over multiple time periods with route planning. Also addressing a multi-period WCP, Teixeira et al. (2004) develop a cluster-first route-second heuristic to schedule and plan

TABLE 6.1: Works on the MDVRP and extensions.

Work	Problem	Approach
Tillman and Cain (1972)	MDVRP	Branch and bound methods
Golden et al. (1977)	TSP, Multiple TSP, CVRP and MDVRP	Heuristics
Raft (1982)	MDVRP	Heuristic-based technique, which is decomposed into: route assignment, depot assignment, vehicle assignment, delivery period, and route design
Renaud et al. (1996)	Capacitated MDVRP with a maximum route length	TS
Cordeau et al. (1997)	Periodic VRP, periodic TSP and capacitated MDVRP	TS
Salhi and Sari (1997)	MDVRP determining the vehicle fleet composition	Three-level methodology, including composite heuristics
Thangiah and Salhi (2001)	MDVRP	GA, an insertion heuristic and a post-optimization method
Wu et al. (2002)	Capacitated multi-depot LRP with a heterogeneous fleet and a limited number of available vehicles	SA
Bae et al. (2007)	Capacitated MDVRP	GA and GUI-type programming
Crevier et al. (2007)	MDVRP with inter-depot routes	TS, a solution pool, a route generation algorithm, a set partitioning algorithm, and a post-optimization process
Pisinger and Ropke (2007)	CVRP, CVRP with time windows, capacitated MDVRP, site-dependent CVRP, and open CVRP	LNS
Chen and Xu (2008)	Capacitated MDVRP	GA and Metropolis acceptance rule of the SA
Ho et al. (2008)	MDVRP	Two hybrid GAs
Dondo and Cerdá (2009)	MDVRP with time windows	Mixed-integer linear programming formulations, a spatial decomposition scheme and a local search improvement algorithm exploring large neighborhoods
Mirabi et al. (2010)	Capacitated MDVRP	Three hybrid heuristics including nearest depot method, CWS heuristic, nearest neighbor heuristic, and SA
Gulczynski et al. (2011)	Multi-depot split delivery VRP	Integer programming-based heuristic
Surekha and Sumathi (2011)	Capacitated MDVRP	Clustering, CWS heuristic, and a GA
Cordeau and Maischberger (2012)	Capacitated MDVRP among other routing problems	ILS and TS
Vidal et al. (2012)	Capacitated MDVRP, periodic VRP, and capacitated multi-depot periodic VRP	Hybrid GA
Juan et al. (2015c)	Capacitated MDVRP	ILS and biased randomization
Escobar et al. (2014)	Capacitated MDVRP	Hybrid granular TS
Karakatic and Podgorelec (2015)	MDVRP	Survey of GAs
Li et al. (2015)	Capacitated MDVRP with simultaneous deliveries and pickups	ILS

waste collection routes for different waste types. Nuortio et al. (2006) present a guided variable thresholding metaheuristic to solve a multi-period WCP. Hemmelmayr et al. (2013) address the periodic VRP with different waste types, which they solve with a VNS metaheuristic. The landfills are considered intermediate facilities, which are inserted in pre-constructed routes using dynamic programming. The authors also discussed the single period WCP with multiple depots, in which the landfills serve as vehicle depots and disposal sites at the same time. Later, Hemmelmayr et al. (2014) discuss the integrated vehicle routing- and bin allocation problem using the same real-life problem set, which they solve with a combination of a VNS metaheuristic for the routing part and a mixed integer linear programming-based exact method for the bin allocation. Ramos et al. (2014) extend the typical objective of minimizing routing costs in order to include environmental concerns, considering multiple waste types and numerous vehicle depots.

Only focusing on waste collection routing, Kim et al. (2006) develop an extension of Solomon's insertion algorithm (Solomon, 1987) to optimize routes of a waste management service provider, considering a capacitated vehicle fleet, time windows, and driver lunch breaks. A benchmark set of 10 realistic instances based on the original case study ranging from 102-2100 nodes is provided. This benchmark set has been later employed by Ombuki-Berman et al. (2007) to test a multi-objective GA. Furthermore, the same benchmark set has been used by Benjamin and Beasley (2010) and Buhrkal et al. (2012) to test their metaheuristic solution methods. Benjamin and Beasley (2010) combine the TS and the VNS metaheuristics. By exchanging containers and landfills within and between routes, the solution search space is systematically increased. Buhrkal et al. (2012) put forward an adaptive LNS metaheuristic. Based on an initial solution, this approach applies a range of destroy-and-repair methods to examine several solution neighborhoods. It is called adaptive since the choice of methods depends on the solution quality obtained during the construction of earlier solutions. Moreover, an acceptance criterion for new solutions based on the SA metaheuristic is included. Recently, Markov et al. (2016) present a multiple neighborhood search heuristic for a real-word application of the WCP with intermediate facilities. The authors consider a heterogeneous vehicle fleet and flexible depot destinations in their approach.

6.2.3 Stochasticity

Regarding the VRP-SD, the first works appear in the 80s. Table **??** shows some related works. It is worth highlighting a few outstanding contributions. In Dror and Trudeau (1986), the concept of route failure is introduced, and its effects on the expected cost of a route are illustrated. A review on the stochastic CVRP is presented in Gendreau et al. (1996), where the main variants are presented. Yang et al. (2000) suggest anticipating possible stock-outs by incorporating preventive breaks or restocking in the route design. The aim is to reduce the probability of route failure and, as a result, the cost. During the last decades, the scientific community has focused on the implementation of metaheuristics. In this context, Bianchi et al. (2006) compare the performance of several methodologies embedding one of the following metaheuristics: SA, TS, ILS, ACO, and EA.

The stochastic MDVRP

The number of works analyzing the stochastic MDVRP is rather limited. Tillman (1969) expands the CWS heuristic to address it. The procedure proposed may be applied to demands with Poisson, exponential, normal, binomial or chi-squared distributions. In Chan et al. (2001), a multi-depot, multiple-vehicle LRP with stochastically processed demands is formulated. The probable demands are estimated by stochastic processes before the vehicle location-routing decisions. Tatarakis and Minis (2009) study the stochastic MDVRP considering both the case in which products are stored dedicatedly or together in a compartment. Dynamic programming algorithms are proposed to determine the minimal routing cost, and an optimal routing policy is derived to decide whether a vehicle has to return to the depot for a reload after serving the current customer or should continue to the next customer. Tauhid et al. (2012) solve the stochastic MDVRP in three phases: first a nearest neighbor classification method is used for grouping the customers; then, the sum-of-subsets method is applied for routing; and finally, the routes are optimized throughout a greedy method. They aim to minimize the number of routes and, accordingly, the number of vehicles needed.

TABLE 6.2: Works on the VRP-SD and related problems.

Work	Problem	Approach	Modeling
Stewart and Golden (1983)	VRP-SD	Formulations and heuristics	Demands follow normal or Poisson distributions
Dror and Trudeau (1986)	VRP-SD	Modification of the CWS heuristic	Demands follow specific distributions from a limited set
Bastian and Kan (1992)	VRP-SD	Formulation for a penalty model, a chance-constrained model, and a full-service model	Demands are independent and identically distributed with known probability distributions
Gendreau et al. (1995)	VRP-SD with stochastic customers	Stochastic integer formulation and integer L-shaped method	Demands are independent, discrete and identically distributed with known probability distributions
Yang et al. (2000)	VRP-SD	Two heuristics	Demands are independent and identically distributed with known probability distributions
Tan et al. (2007)	VRP-SD with time windows	MOEA and simulation	Demands follow normal distributions
Ismail and Irhamah (2008)	VRP-SD	GA and TS	Demands follow independent and discrete uniform distributions
Moghaddam et al. (2012)	CVRP with uncertain demands	PSO	No assumptions are made
Goodson (2015)	Multi-compartment VRP-SD	Method to calculate the expected cost of a solution integrated in a cyclic-order-based SA	Demands follow discrete distributions

None of the aforementioned works deals with the stochastic and capacitated MDVRP analyzed here. In particular, Tillman (1969) and Tauhid et al. (2012) consider unlimited capacities at each depot -which significantly reduces the difficulty of the problem and constitutes an unrealistic assumption. In addition, Tauhid et al. (2012) do not really consider stochastic demands. Also, Tillman (1969) makes strong assumptions on the probability distributions of these demands. Chan et al. (2001) and Tatarakis and Minis (2009) deal with problems that, although somewhat related, can not be considered stochastic and capacitated MDVRPs. While the former focuses on location issues, in the latter a single vehicle must deliver multiple products given a predefined customer sequence.

The stochastic WCP

Concerning the WCP with stochastic demands, the literature is scarce. The ACO metaheuristic and a hybrid approach based on a GA and TS for a case study with 50 containers is presented in Ismail and Irhamah (2008), and Ismail and Loh (2009). After planning a priori routes, waste levels are simulated according to a discrete probability distribution. Routes undergo a recourse action (i.e., an additional disposal trip) whenever actual demand exceeds the planned collection amount. Nolz et al. (2014) formulate a collector-managed IRP for a case study on the collection of infectious waste. By using real information obtained through radio frequency identification, their ALNS algorithm is able to consider stochastic inputs. Alshraideh and Abu Qdais (2016) combine a multi-period WCP with time windows and stochastic demands. They use a GA and a probability constraint regarding a pre-defined service level to solve the problem.

6.2.4 Sustainability

The increasing social concern for the environment and a sustainable growth requires the transformation of cities. During the last decade, the green VRP (GVRP) and the pollution VRP (PVRP) have become increasingly popular. While the former is focused on the environmental impact caused by the fuel or energy consumption of transport, the latter takes into account the pollution and different emissions generated. Thus, both problems analyze the emissions and fuel/energy consumption levels, which depend on traffic congestion, speed, acceleration, type of road, type of vehicle, and load, among other internal and external factors of the operation (Bektaş and Laporte, 2011; Koç et al., 2014).

Regarding environmental impacts, the distance and vehicle weight play a crucial role in the fuel/energy consumption and carbon emissions, thereby Ubeda et al. (2011) aim at reducing transport costs and emissions, considering the distance and some variations in the vehicle maximum capacity. It is concluded that enhancing load factors (which may be achieved by using heterogeneous fleets) is an efficient way to get significant savings and environmental benefits. The authors also discuss negative externalities of transport such as noise, air pollution, congestion, accident rate, energy consumption and land use, among others. There are studies tackling the negative impacts from three perspectives: negative externalities, emissions released and fuel consumption. Faulin et al. (2011), Liu et al. (2014), and Zhang et al. (2015) consider environmental indicators for the CVRP; they state that the load variation defines fuel consumption and emissions caused by transport. Besides, the load variation influences the distribution processes profitability. In this line, Kuo (2010), Demir et al. (2014), and Xiao and Konak (2015) develop methodologies for the green heterogeneous VRP, considering traffic congestion, road gradient, speed variations, and distance traveled as variables that influence fuel consumption and as elements that characterize the urban transport dynamics (Jabbarpour et al., 2015). More recently, Niknamfar and Niaki (2016) study the MDVRP with time windows to optimize the customers-depots allocation and the vehicles selection aiming to minimize the environmental impacts. They demonstrate that an optimal allocation and coordination between stakeholders not only reduce the negative impacts but also enhance the total profit. Juan et al. (2014c) consider a supply chain with multiple suppliers for minimizing the empty trips and the travel distance in each route. They conclude that it is possible to reduce the CO2 emission to 23% when the distribution process is carried out in collaboration with multiple suppliers. Wang et al. (2014b) demonstrate that considering environmental criteria allows a saving up to 10% of the operation costs. The authors develop an algorithm to integrate the economic and environmental goals based on the MDVRP with backhauls. Demir et al. (2014)

consider the MDVRP with freight pick-up and delivery to ensure that any customer demand can be met from any depot and thus reducing the operation cost.

Some studies focus on the analysis of environmental impacts caused by transport activities in urban zones, however there is no characterization for getting a rough estimation of the real impact of these activities. For instance, about 60% of transport activities take place in urban regions at where around 80% population is concentrated, making people the main harmed (European Commission, 2015). Social impact refers to health problems and other factors such as quietness, air quality, urban esthetic, accessibility and urban safety. Penalties, taxes or willingness to pay as a means to reduce the social impacts constitute the costs associated. It is estimated that about 0.4%, 0.2%, 1.5% and 2% of the gross domestic product is related to air pollution problems, noise, accidents and traffic congestion, respectively (Caceres et al., 2014). Therefore, the sustainability concept has started to take part in the decision-making process but there is a lack of structured tools that allow the integration of the three dimensions and support decision-makers (Chen et al., 2013).

There are only a few works on sustainability criteria. Chibeles-Martins et al. (2016) pose ecological criteria to determine an optimal structure of distribution networks. They solve a bi-objective problem determining the suitable locations, capacities and attributes in factories, warehouses and a distribution center. The solution method is based on the SA metaheuristic, and Pareto optimality is considered to get a balancing between economic and ecological concerns. In the same sense, Zhang et al. (2016) implement EAs to determine the optimal design of supply chains considering two possible scenarios: first, the transport is outsourced and second the transport is leased. It is a multi-objective problem aimed at minimizing CO2 emissions, fine dust and costs. The authors implement the non-dominated sorting GA-II (NSGA-II) and the strength Pareto EA2 (SEAP2) to compare their performance, both methods take into account Pareto optimality through a scalarization method computed by a weighted sum. Later, Kadziński et al. (2017) define a sustainable objective to design an optimal distribution structure considering a supply chain with multi-distribution channels. Objectives are maximizing customer coverage, and minimizing cost and environmental impacts. Notice that social objectives do not respond to problems highlighted by the society, besides these approaches belong to strategic levels without considering the synergy among tactical levels, operative levels and stakeholders' particular objectives.

6.3 The MDVRP-SD

The stochastic and capacitated MDVRP is characterized by the randomness of at least one of its parameters or structural variables. These random variables follow specific probability distributions. This problem may be seen as a non-trivial extension of the stochastic CVRP (Gendreau et al., 1996; Stewart and Golden, 1983). There are three problems belonging to this family: the CVRP with stochastic demands, which is the most popular (Bianchi et al., 2006); the CVRP with stochastic customers (Bertsimas, 1988); and the CVRP with stochastic times (Laporte et al., 1992; Kenyon and Morton, 2003). The MDVRP may be described as follows. Let $G = \{V, E\}$ be a complete directed graph, where $V = \{V_d, V_c\}$ is the set of vertices including the depots (V_d) and the customers (V_c), and E is the set of edges connecting all vertices in V. Each customer $i \in V_c$ has a positive demand d_i. Each depot $p \in V_d$ has assigned a maximum number of vehicles, m. All vehicles are supposed to have the same capacity W. Each edge in E has an associated cost $c_{ij} = c_{ji} \geq 0$. A solution is a set of routes in which each route starts at one depot in V_d, connects one or more customers in V_c, and ends at the same depot (Figure **??**). Moreover, each customer must be visited only once. The MDVRP-SD differs in the following two consideration: *(i)* each customer has a positive demand D_i that follows a probability distribution, either theoretical or empirical, with an existing mean denoted as $E[D_i]$; and *(ii)* each customer is visited once except in the undesirable case in which a route failure occurs. While the demands' distribution is known beforehand, the exact demand cannot be revealed until the vehicle reaches the customer.

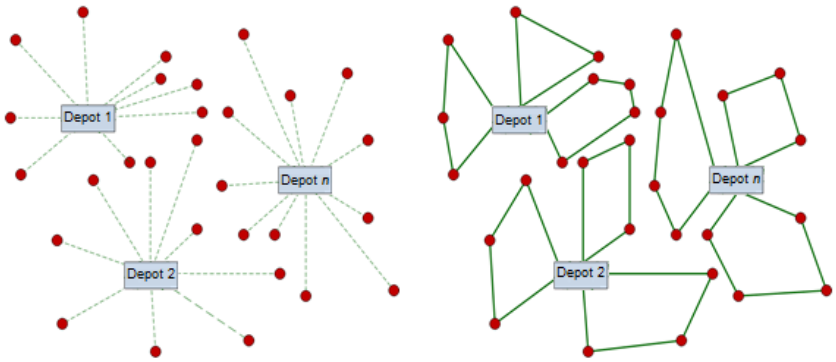


FIGURE 6.1: Customer-depot assignment and posterior routing processes in the MDVRP.

The classical goal of the MDVRP-SD is to find a solution that minimizes the expected routing cost while satisfying the customer demands, and the constraints related to the number of vehicles, and the vehicles' capacity. However, other constraints may also apply, e.g.: a maximum allowable cost for a route, time windows for visiting each customer, etc. In addition, different goals may be proposed such as solution balance or minimization of environmental costs. Even in its simplest version, this problem represents a challenge since it integrates a combinatorial assignment problem, in which each customer is assigned to one depot, with several stochastic CVRPs, one per depot. The additional complexity lies in the interrelation between assigning and routing issues.

One way to model the MDVRP-SD is as a two-stage problem. In the first stage (design stage), a set of routes is designed considering the probability distributions associated with each customer's demand. The second stage (corrective stage) specifies the actual route of each vehicle, which may include corrective actions if the route fails, i.e., if the demand of a customer visited by a given vehicle is higher than the remaining vehicle capacity. In this case, the vehicle must return to the depot to reload. Often, the possibility of re-stocking is allowed, that means that a vehicle may return to the depot before it has run out of capacity. For instance, if the remaining vehicle capacity is not enough to satisfy the expected demands of the customers that still have to be served. The solution must minimize the expected total cost, which is the sum of the costs of the routes planned in the first stage (fixed cost), and the expected costs due to corrective actions (variable cost).

6.3.1 Methodology

The proposed approach relies on two facts: *(i)* the MDVRP-SD can be considered a generalization of the MDVRP, i.e., the MDVRP can be seen as a MDVRP-SD in which the random demands have zero variance; and *(ii)* despite the fact that the MDVRP-SD has not been intensively studied, there exists efficient algorithms for solving the MDVRP.

The general ideas behind the approach are described next. Initially, given an instance of the MDVRP-SD, it is transformed into a deterministic one by replacing each random variable by its mean. A set of high-quality solutions for the deterministic version is then obtained by using an efficient algorithm. While the search takes place, MCS techniques are employed to assess the performance of these promising solutions for the stochastic version. The best solution is defined as the one with the lowest expected total cost. Safety stocks are employed as suggested in Juan et al. (2011b). A safety stock is a certain amount of the vehicle capacity that is not considered while designing the routes. Then, if the final routes' demands surpass their expected values, this stock can be employed to try to satisfy them. Thus, the aim of considering safety stocks is to reduce the probability of a route failure.

Proposed steps

The flowchart diagram is depicted in Figure **??** and described next:

1. Consider a MDVRP-SD instance defined by a set of n customers. Each customer i has associated a demand D_i ($1 \leq i \leq n$) that follows a known probability distribution with an existing mean $E[D_i]$.

2. Determine a set K of percentages, where each element k_l is the percentage of the vehicle capacity (W) that can be used during the route design phase; in other words, $1-k_l$ represents a fixed level of safety stock. For each of these elements, follow the steps 3 to 9.

3. Consider the capacitated MDVRP(k_l) with a total vehicle capacity of $W_l^* = k_l \cdot W$ and deterministic demands $d_i = E[D_i]$.

4. Generate an initial solution for the MDVRP(k_l). This solution is also an aprioristic solution for the MDVRP-SD. It will be employed "as it is" as long as there is no need of corrective actions (routes failures and re-stockings). Therefore, the cost associated to this solution, $C_{MDVRP}(k_l)$, can be considered a base or fixed cost of the MDVRP-SD solution. In the case of the stochastic problem, there is also a variable cost $C_{CA}(k_l)$ that depends on the corrective actions undertaken. Consequently, for a given value of k_l, the total cost of the 'stochastic' solution (the one associated with the MDVRP-SD) is the sum of the fixed cost corresponding to the 'deterministic' solution (the one associated with the MDVRP) and the variable cost due to corrective actions, $C_{MDVRP-SD}(k_l) = C_{MDVRP}(k_l) + C_{CA}(k_l)$.

5. Use MCS to estimate the expected cost due to corrective actions for each route j of the aprioristic solution, $E[C_{CA}^j(k_l)]$ $(1 \leq j \leq m)$. Then, aggregate the expected total cost for all routes, $E[C_{CA}(k_l)]) = \sum_{j=1}^{m} E[C_{CA}^j(k_l)]$. In this phase, a short simulation is used to quickly get that estimate. Then, the expected total cost of the solution is calculated as follows: $E[C_{MDVRP-SD}(k_l)] = C_{MDVRP}(k_l) + E[C_{CA}(k_l)]$.

6. Set a base solution as the initial solution.

7. Employ a metaheuristic algorithm, which starts an improvement process that will continue until a stopping condition, based on time or a fixed number of iterations, is reached. At each iteration the following steps are implemented. First, a perturbation is applied to the base solution to generate a new one. If the fixed cost of the new solution is lower than the fixed cost of the current base solution, then the list of the best deterministic solutions is updated (only if it is not full or if the worst solution has a higher cost, then a swap is performed) and the expected total cost of the new solution is estimated with a short simulation. If this cost is lower than the expected total cost of the base solution, the latter is replaced and the list of the best stochastic solutions is updated. Otherwise, an acceptance criterion is used to decide whether the base solution is deteriorated to the new one. Before that, if the fixed cost of the new solution is higher, then that solution is discarded. This iterative process will provide, after analyzing many possible solutions, a list of promising solutions for the MDVRP-SD.

8. Try to improve all promising solutions with an intensive routing algorithm.

9. Use a long simulation to generate a sample of total costs for each promising solution. Large samples are required to obtain estimates with small confidence intervals.

10. Finally, return the top best stochastic solutions (considering all solutions found with the different values in K), and the corresponding samples (they will be used for completing a risk analysis).

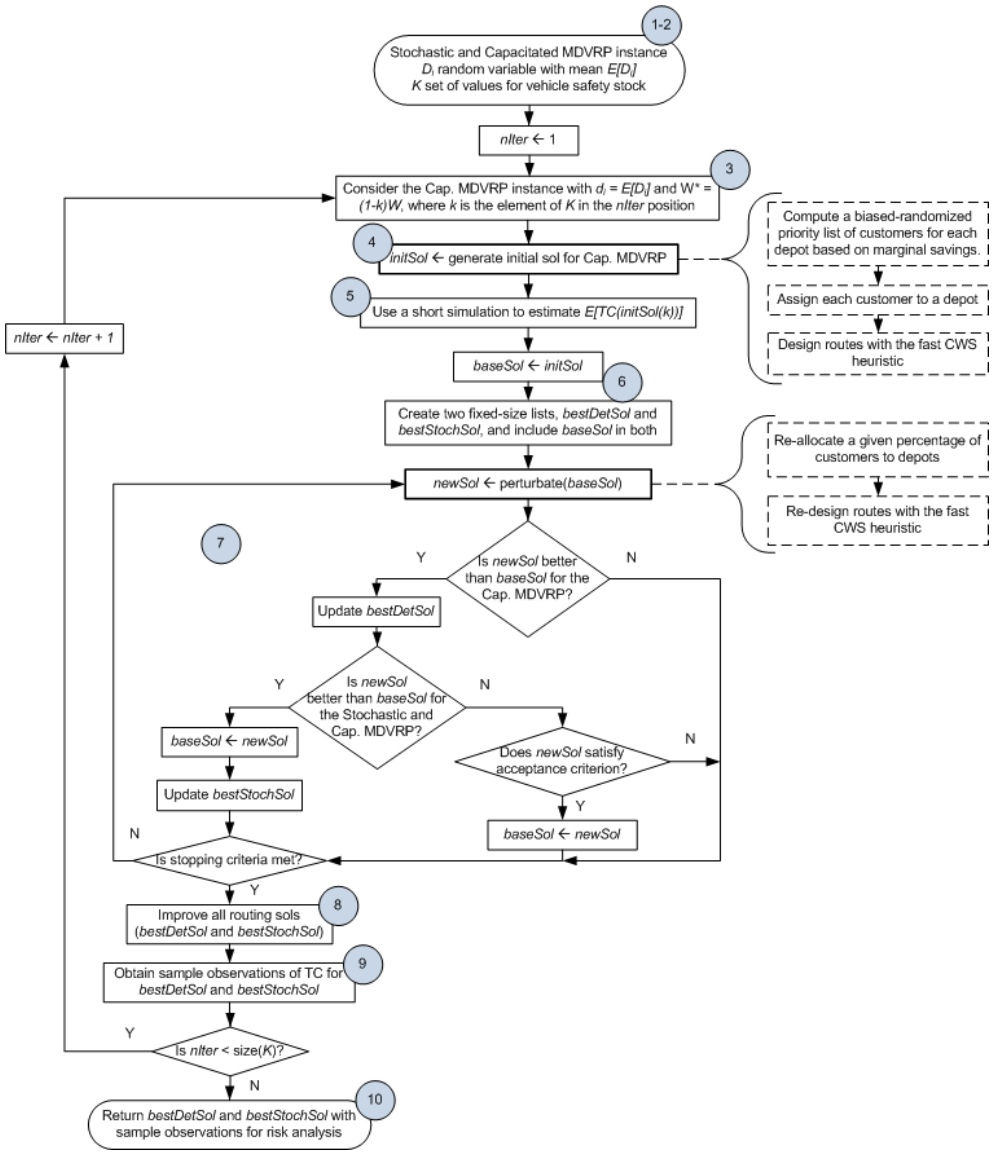


FIGURE 6.2: Flowchart of the proposed approach for the MDVRP-SD.

Details and further considerations

The algorithm used to get the initial solution for the MDVRP is the one proposed in Juan et al. (2015c). Initially, a customer-depot assignment map is set, and then the CWS heuristic is applied to obtain a fast routing plan. Regarding safety stocks, it is expected that lower values of k_l will provide more reliable routes, as a high percentage of the vehicle capacity will be reserved as safety stock. However, a high fixed cost will result too, since more vehicles will be needed to cover all the customers' demands. On the other hand, a high value of k_l is related to a lower fixed cost but a higher variable cost due to the elevated risk of having to return to the depot to reload. Considering the trade-off between these two costs, different values are tested as indicated in step 2.

The cost due to corrective actions is computed as follows. In case of route failure, it includes the cost of returning to the depot first and then to the customer being served. It is assumed that the vehicle delivers all the remaining stock before going back to reload. A re-stocking is carried out when the expected demand of the next customer is higher than the current remaining stock. The cost of this strategy incorporates the costs on the edges that link a customer with the depot and the depot with the next customer minus the cost of the edge linking both customers.

The perturbation operator modifies the current solution by reallocating a given percentage p of customers randomly selected, considering the remaining capacity of the depots, and the distance-based cost for each pair of customer and depot. The savings heuristic is applied again to design the routes. A Demon-like acceptance criterion (Talbi, 2009) is used to diversify the search.

In order to improve the most promising solutions found within the metaheuristic framework, the routing algorithm proposed by Juan et al. (2011a) is applied to each one. This algorithm is based on a randomized version of the savings heuristic that employs a geometric distribution to guide the random search, and a cache and splitting techniques to make it more efficient. This algorithm has been adapted for the stochastic solutions.

Finally, it is interesting to observe that, considering the parameters (not the estimates), the fixed cost and the expected total cost of the best deterministic solution represent a lower and an upper bound, respectively, of the expected total cost associated to the best stochastic solution. The set of samples will allow us to compare the solutions not only focusing on the expected total cost, but also on the distribution of the total cost.

6.3.2 Computational experiments

The algorithm has been implemented as a Java application and tested on 23 MDVRP benchmark instances: the first seven were proposed by Christofides and Eilon (1969), the following four were created by Gillett and Johnson (1976) and the remaining are described in Chao et al. (1993). Vidal et al. (2012), Escobar et al. (2014), and Juan et al. (2015c) are some recent works using them. These instances have been adapted as described next. The demand of each customer (d_i) has been considered as a random variable D_i following a log-normal distribution with mean d_i and variance vd_i. Three different scenarios have been considered, each one with a respectively different variance: 0.1 $E[D_i]$, 0.5 $E[D_i]$, and 1 $E[D_i]$, where $E[\cdot]$ represents the mean or expected value. In order to choose the percentage of the vehicle capacity in the route design phase ($1 - k_l$), 5 equally-spaced values varying from 0.90 to 1.00 have been tested.

The computational time is limited to 30 seconds. The number of seeds is set to 10, and only the best result is stored. Concerning the number of iterations in each simulation, 200 runs have been employed for the short simulations and 2,000 runs for the long simulations. The selection of these values, as well as of the number of solutions stored in the list of top solutions (4), is mainly driven by the total computing time available. Biased randomization techniques rely on two geometric distributions (one for mapping and one for routing) and, therefore, they require distribution parameters: bM and bR, respectively. Additionally, there is a parameter p which controls the percentage of nodes that may be reallocated in a solution when perturbing it. They have been tuned by performing a full factorial experiment. bM, bR and p follow uniform distribution between $[0.3, 0.4]$, $[0.2, 0.3]$ and $[0.3, 0.4]$.

Results are displayed in Tables **??**, **??**, and **??**. Each of these tables represents a specific scenario. The first column identifies the instance and the second shows the best known solution (BKS) for the MDVRP. The next five columns are associated with the solution with the lowest fixed cost: the first, the *best deterministic solution - fixed cost* (BDS-FC), represents the fixed cost; the second calculates the gap between the BKS and the BDS-FC, which reveals the performance of the algorithm for the deterministic version of the problem; the third, the *best deterministic solution - total expected cost* (BDS-TEC), is the expected total cost; the fourth, the *best deterministic solution - reliability* (BDS-R), has been computed as one minus the number of route failures divided by the number of routes; and the fifth, *best deterministic solution - k* (BDS-K), provides the percentage of the vehicle total capacity chosen. The following column represents the gap between the BDS-TEC and the BDS-FC. The next three columns are associated with the solution with the lowest expected total cost: the *best stochastic solution - total expected cost* (BSS-TEC) contains the expected total cost; the following two columns, the *best stochastic solution - reliability* (BSS-R) and the *best stochastic solution - k* (BSS-K), show the associated reliability, and the k-value respectively. The next two columns are the gaps between the BSS-TEC and the BKS, and between the BSS-TEC and the BDS-FC, respectively. It is important to highlight that, provided a 'large' number of simulation iterations is used, the BSS-TEC is bounded by the BDS-FC and the BKS (lower bounds), and the BDS-TEC (upper bound). Therefore, the previous gaps show the difference between the BSS-TEC and its lower bounds. The last column is the gap between the expected total costs of both solutions.

TABLE 6.3: Table of results for the MDVRP benchmark instances with a low demand variability.

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	583.11	1.08%	594.85	1.00	1.00	2.01%	594.85	1.00	1.00	3.12%	2.01%	0.00%
p02	473.53	480.72	1.52%	487.43	1.00	1.00	1.39%	483.63	1.00	0.925	2.13%	0.60%	-0.78%
p03	641.19	648.60	1.16%	668.71	1.00	1.00	3.10%	654.34	1.00	0.95	2.05%	0.88%	-2.15%
p04	1001.04	1043.05	4.20%	1108.63	1.00	1.00	6.29%	1073.77	1.00	0.95	7.27%	2.95%	-3.14%
p05	750.03	775.13	3.35%	792.15	1.00	1.00	2.19%	778.85	1.00	0.975	3.84%	0.48%	-1.68%
p06	876.50	897.16	2.36%	966.85	1.00	1.00	7.77%	924.28	1.00	0.925	5.45%	3.02%	-4.40%
p07	881.97	899.96	2.04%	961.25	1.00	1.00	6.81%	940.16	1.00	0.975	6.60%	4.47%	-2.19%
p08	4371.66	4495.17	2.83%	5058.74	0.99	1.00	12.54%	4623.25	1.00	0.975	5.75%	2.85%	-8.61%
p09	3858.66	3962.64	2.69%	4273.86	1.00	1.00	7.85%	4056.49	1.00	0.975	5.13%	2.37%	-5.09%
p10	3629.60	3738.80	3.01%	3925.71	1.00	1.00	5.00%	3794.06	1.00	0.975	4.53%	1.48%	-3.35%
p11	3545.18	3612.26	1.89%	3815.84	1.00	1.00	5.64%	3677.32	1.00	0.975	3.73%	1.80%	-3.63%
p12	1318.95	1318.95	0.00%	1326.89	1.00	1.00	0.60%	1326.74	1.00	0.975	0.59%	0.59%	-0.01%
p13	1318.95	1318.95	0.00%	1326.71	1.00	0.95	0.59%	1326.68	1.00	1.00	0.59%	0.59%	0.00%
p14	1360.12	1360.12	0.00%	1361.93	1.00	0.90	0.13%	1361.50	1.00	0.975	0.10%	0.10%	-0.03%
p15	2505.42	2557.53	2.08%	2666.42	1.00	1.00	4.26%	2590.15	1.00	0.925	3.38%	1.28%	-2.86%
p16	2572.23	2587.86	0.61%	2616.13	1.00	0.975	1.09%	2616.13	1.00	0.975	1.71%	1.09%	0.00%
p17	2709.09	2714.66	0.21%	2718.27	1.00	1.00	0.13%	2718.27	1.00	1.00	0.34%	0.13%	0.00%
p18	3702.85	3812.30	2.96%	3841.51	1.00	1.00	0.77%	3833.52	1.00	0.975	3.53%	0.56%	-0.21%
p19	3827.06	3876.15	1.28%	3918.52	1.00	1.00	1.09%	3918.52	1.00	1.00	2.39%	1.09%	0.00%
p20	4058.07	4085.91	0.69%	4091.26	1.00	0.90	0.13%	4091.26	1.00	0.90	0.82%	0.13%	0.00%
p21	5474.84	5681.16	3.77%	5849.68	1.00	1.00	2.97%	5741.34	1.00	0.925	4.87%	1.06%	-1.85%
p22	5702.16	5808.74	1.87%	5864.13	1.00	0.975	0.95%	5864.13	1.00	0.975	2.84%	0.95%	0.00%
p23	6078.75	6140.01	1.01%	6147.81	1.00	0.90	0.13%	6147.47	1.00	0.90	1.13%	0.12%	-0.01%

TABLE 6.4: Table of results for the MDVRP benchmark instances with a medium demand variability.

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	580.49	0.63%	621.78	0.99	1.00	7.11%	607.92	1.00	0.925	5.38%	4.73%	-2.23%
p02	473.53	481.05	1.59%	495.72	0.99	1.00	3.05%	484.47	1.00	0.925	2.31%	0.71%	-2.27%
p03	641.19	648.80	1.19%	672.36	1.00	1.00	3.63%	663.64	1.00	0.925	3.50%	2.29%	-1.30%
p04	1001.04	1039.64	3.86%	1194.31	0.98	1.00	14.88%	1136.68	0.99	0.95	13.55%	9.33%	-4.83%
p05	750.03	775.40	3.38%	814.19	0.99	1.00	5.00%	797.13	1.00	0.95	6.28%	2.80%	-2.10%
p06	876.50	896.83	2.32%	991.64	0.98	1.00	10.57%	947.47	1.00	0.925	8.10%	5.65%	-4.45%
p07	881.97	901.93	2.26%	988.31	0.98	1.00	9.58%	975.88	0.99	0.975	10.65%	8.20%	-1.26%
p08	4371.66	4496.87	2.86%	5173.76	0.99	1.00	15.05%	4726.09	1.00	0.95	8.11%	5.10%	-8.65%
p09	3858.66	3981.77	3.19%	4290.58	0.99	1.00	7.76%	4116.24	1.00	0.95	6.68%	3.38%	-4.06%
p10	3629.60	3754.90	3.45%	4032.78	1.00	1.00	7.40%	3866.11	1.00	0.95	6.52%	2.96%	-4.13%
p11	3545.18	3607.04	1.75%	3953.10	0.99	1.00	9.59%	3767.31	1.00	0.925	6.27%	4.44%	-4.70%
p12	1318.95	1318.95	0.00%	1356.83	1.00	0.975	2.87%	1356.83	1.00	0.975	2.87%	2.87%	0.00%
p13	1318.95	1318.95	0.00%	1355.84	1.00	0.975	2.80%	1355.84	1.00	0.975	2.80%	2.80%	0.00%
p14	1360.12	1360.12	0.00%	1393.37	1.00	0.90	2.45%	1392.10	1.00	0.925	2.35%	2.35%	-0.09%
p15	2505.42	2549.17	1.75%	2722.47	0.99	1.00	6.80%	2635.44	1.00	0.90	5.19%	3.38%	-3.20%
p16	2572.23	2587.86	0.61%	2671.31	1.00	0.975	3.22%	2671.31	1.00	0.975	3.85%	3.22%	0.00%
p17	2709.09	2714.66	0.21%	2783.04	1.00	1.00	2.52%	2783.04	1.00	1.00	2.73%	2.52%	0.00%
p18	3702.85	3814.73	3.02%	3930.95	1.00	1.00	3.05%	3916.54	1.00	0.925	5.77%	2.67%	-0.37%
p19	3827.06	3875.40	1.26%	4002.83	1.00	1.00	3.29%	4002.83	1.00	1.00	4.59%	3.29%	0.00%
p20	4058.07	4085.91	0.69%	4187.05	1.00	0.90	2.48%	4187.05	1.00	0.90	3.18%	2.48%	0.00%
p21	5474.84	5690.22	3.93%	5891.98	1.00	1.00	3.55%	5870.96	1.00	0.925	7.24%	3.18%	-0.36%
p22	5702.16	5796.48	1.65%	5980.72	1.00	0.975	3.18%	5980.72	1.00	0.975	4.89%	3.18%	0.00%
p23	6078.75	6140.01	1.01%	6290.98	1.00	0.90	2.46%	6289.53	1.00	0.90	3.47%	2.44%	-0.02%

TABLE 6.5: Table of results for the MDVRP benchmark instances with a high demand variability.

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	587.18	1.79%	633.62	0.99	1.00	7.91%	620.93	1.00	0.925	7.64%	5.75%	-2.00%
p02	473.53	479.45	1.25%	491.96	0.99	1.00	2.61%	485.77	1.00	0.90	2.58%	1.32%	-1.26%
p03	641.19	649.87	1.35%	679.88	1.00	1.00	4.62%	671.69	1.00	0.925	4.76%	3.36%	-1.20%
p04	1001.04	1042.05	4.10%	1197.08	0.97	1.00	14.88%	1177.35	0.98	0.95	17.61%	12.98%	-1.65%
p05	750.03	777.41	3.65%	821.53	0.98	1.00	5.67%	806.97	0.99	0.95	7.59%	3.80%	-1.77%
p06	876.50	897.48	2.39%	1021.06	0.97	1.00	13.77%	971.84	0.99	0.925	10.88%	8.29%	-4.82%
p07	881.97	907.38	2.88%	1011.02	0.97	1.00	11.42%	991.80	0.97	1.00	12.45%	9.30%	-1.90%
p08	4371.66	4498.65	2.90%	5267.60	0.98	1.00	17.09%	4772.74	1.00	0.925	9.17%	6.09%	-9.39%
p09	3858.66	3962.30	2.69%	4398.51	0.99	1.00	11.01%	4185.01	1.00	0.95	8.46%	5.62%	-4.85%
p10	3629.60	3747.91	3.26%	4106.36	0.99	1.00	9.56%	3940.02	1.00	0.95	8.55%	5.13%	-4.05%
p11	3545.18	3625.26	2.26%	4010.29	0.99	1.00	10.62%	3788.72	1.00	0.925	6.87%	4.51%	-5.53%
p12	1318.95	1318.95	0.00%	1377.66	1.00	1.00	4.45%	1377.66	1.00	1.00	4.45%	4.45%	0.00%
p13	1318.95	1318.95	0.00%	1376.28	0.99	1.00	4.35%	1376.28	0.99	1.00	4.35%	4.35%	0.00%
p14	1360.12	1360.12	0.00%	1417.01	0.99	0.90	4.18%	1414.06	0.99	0.925	3.97%	3.97%	-0.21%
p15	2505.42	2553.90	1.93%	2755.74	0.98	1.00	7.90%	2673.11	1.00	0.90	6.69%	4.67%	-3.00%
p16	2572.23	2590.77	0.72%	2712.30	0.99	0.975	4.69%	2712.30	0.99	0.975	5.45%	4.69%	0.00%
p17	2709.09	2714.66	0.21%	2823.97	1.00	0.90	4.03%	2823.97	1.00	0.90	4.24%	4.03%	0.00%
p18	3702.85	3813.22	2.98%	4005.61	0.99	0.975	5.05%	3971.55	1.00	0.925	7.26%	4.15%	-0.85%
p19	3827.06	3876.15	1.28%	4054.76	0.99	1.00	4.61%	4054.76	0.99	1.00	5.95%	4.61%	0.00%
p20	4058.07	4085.91	0.69%	4252.71	0.99	0.90	4.08%	4252.71	0.99	0.90	4.80%	4.08%	0.00%
p21	5474.84	5677.62	3.70%	5974.19	0.99	1.00	5.22%	5957.66	0.99	0.925	8.82%	4.93%	-0.28%
p22	5702.16	5812.03	1.93%	6079.27	0.99	0.975	4.60%	6079.27	0.99	0.975	6.61%	4.60%	0.00%
p23	6078.75	6140.01	1.01%	6388.07	1.00	0.90	4.04%	6386.58	1.00	0.975	5.06%	4.02%	-0.02%

TABLE 6.6: Summary of results for the MDVRP benchmark instances.

Scenario	G. BDS-FC - BKS	G. BDS-TEC - BDS-FC	G. BSS-TEC - BKS	G. BSS-TEC - BDS-FC	G. BSS-TEC - BDS-TEC
Var: $0.10E[D_i]$	1.83%	3.10%	3.12%	1.26%	−1.69%
Var: $0.50E[D_i]$	1.83%	5.89%	5.53%	3.62%	−2.06%
Var: $1.00E[D_i]$	1.74%	7.48%	7.12%	5.27%	−1.97%

6.3.3 Analysis of results

The results obtained show that assuming a problem being deterministic can lead to solutions with poor performance even in scenarios characterized by demands with a relatively low variance. In all experiments, the expected total cost obtained with the best stochastic solution is better than the one obtained with the best deterministic solution. There is a case in which the gap reaches the −9.39% (instance 'p08' with high variance). The reason is that the deterministic solution is not balanced, and a high variance results in an increasing of the expected total cost. Figure **??** illustrates the case of the instance 'p02' with a high variance. The vehicle capacity is 160. The left and right plots represent the best deterministic solution and the best stochastic solution, respectively. The numbers in the nodes reveal the expected customer demands, while the numbers in the center of each route are the total demands. Although the routes are similar, notice that the best stochastic solution seems more 'balanced' in terms of demands, which explains why it is also more reliable.

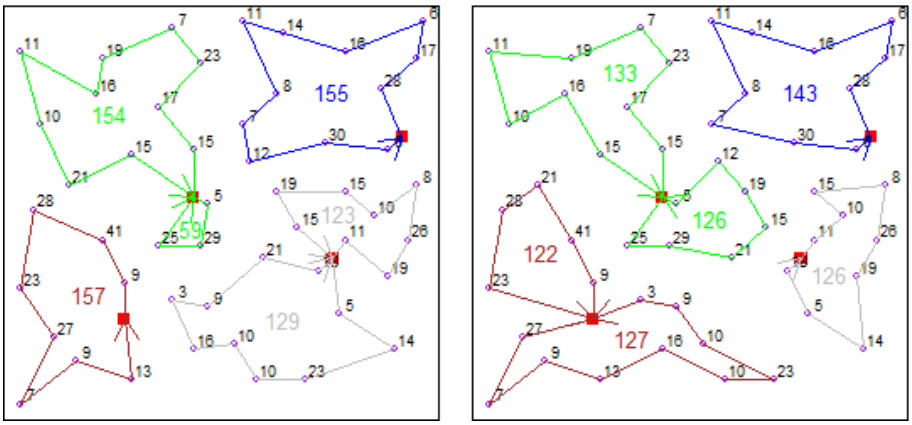


FIGURE 6.3: Best deterministic (left) and stochastic solutions (right) for the MDVRP instance 'p02'.

Table **??** summarizes the results described in Tables **??**, **??** and **??**. For each scenario, it shows the mean gaps. The mean gaps between the BDS-FC and the BKS, which ranges from 1.74% to 1.83%, show that our approach is relatively competitive for finding the best solution to the deterministic problem. The third column reveals that the difference between the BDS-TEC and the BDS-FC (i.e., the total cost if there was no stochasticity) is positive and positively correlated with the variability of the scenario. Next two columns quantify the gaps between the BSS-TEC and its lower bounds, the BKS and the BDS-FC. They both increase as the variability of the scenario gets higher. Finally, the last gap shows the benefit of using the simheuristic approach. Thus, it can be concluded that the higher the variability the higher the benefit.

Here a risk analysis is presented in which the four best stochastic solutions and the best deterministic solution are compared. It is illustrated on a specific case, the instance 'p09' with high variance. Thus, Figure **??** shows a boxplot of the total costs obtained by means of MCS. It can be stated that the variability of total costs associated to the best deterministic solution is the highest, and all distributions present a positive skew. In Figure **??**, the empirical cumulative distributions functions (CDFs) for the best deterministic and stochastic solutions are drawn. The probability distribution function of the best stochastic solution is above the other almost for the entire domain. In other words, the probability of having a total cost equal to or lower than a given value is usually higher with this solution. As a consequence, a risk-averse decision-maker would prefer it. Nevertheless, the minimum values are provided by the deterministic solution, which makes sense since

this solution will be the one selected in scenarios where the customer demands are similar to the corresponding mean of the distributions.

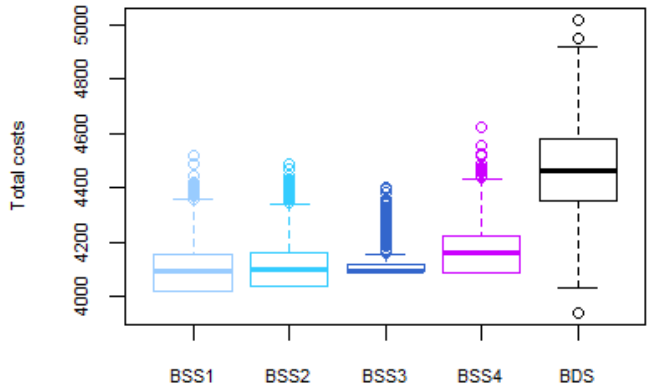


FIGURE 6.4: Boxplots of best solutions for the MDVRP instance 'p09' with high variability.

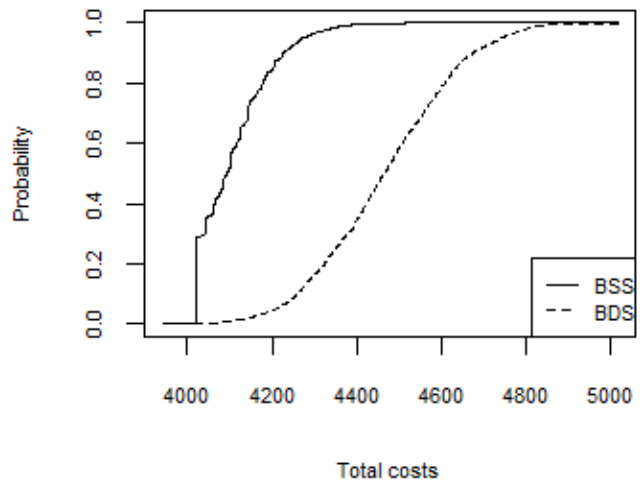


FIGURE 6.5: CDFs of best deterministic and stochastic solutions for the MDVRP instance 'p09' with high variability.

6.4 The MDVRP-HD

The problem addressed here is an extension of the MDVRP, already introduced in this chapter. When adopting a marketing perspective, companies focus on market segmentation to group customers according to their features and preferences. Considering the heterogeneity of markets,

segmentation attempts to divide customers into subsets that behave in a similar way. This extension of the MDVRP aims at assigning customers to depots based not only on distribution costs but also on customers' features and preferences. The goal is to optimize expected benefits by considering both distribution costs and expected incomes.

6.4.1 Methodology

In order to assign customers to depots, the heterogeneity of the depots is taken into account. It is a realistic approach, since depots belonging to the same organization usually have different characteristics related to products, trade credit policies, and complementary services, among others. This diversity leads to consider customer preferences. Specifically, the willingness to consume (or expenditure) of each customer depends on how well the assigned depot fits his/her preferences. Market segmentation techniques are applied to identify subsets of customers with similar profiles and assign them to the particular depot that better fits their preferences, considering the restrictions of the problem. Accordingly, it is proposed to study the relationship between expenditure and customers' features from data of existent customers by employing statistical learning methodologies (e.g., prediction techniques). It will enable the assignation of new customers in such a way that the expected benefits (expected incomes minus distribution costs) is maximized. The phases of the proposed approach are represented in Figure **??** and described next:

1. Data collection. The approach requires several inputs: database of historical sales, description of new customers, location of depots, vehicle maximum capacity, number of available vehicles at each depot, and maximum distribution costs per route. The sales database includes the following information for each existent customer: personal features, geographical location, expenditure level, and depot to which he/she has been assigned (randomly or according to a metric not related to personal features such as distribution costs). The description of new customers gathers personal features and geographical locations. Regarding the information of both existent and new customers, an initial selection of variables has to be performed by assessing which ones may be valuable. Besides explaining the differences of expenditures among depots, they should be easy to obtain, estimate or compute, and store.

2. Statistical learning. Given the database of existent customers, a statistical model exploring the relationship between customers' features and expenditure is performed for each group of customers assigned to a specific depot. Considering several groups, it is allowed the existence of a different trend in each one. A high number of methodologies are available to carry out regression analysis (Hastie et al., 2009; Lantz, 2013).

3. Prediction of expenditure for new customers. Once a methodology has been selected and the different functions have been fitted, the expenditure is predicted for each new customer given his/her features if assigned to each depot.

4. Assignment of customers to depots. In order to perform an efficient and feasible assignation, it is necessary not only to consider the predicted expenditure but also the distribution costs, the maximum number of vehicles per depot, and their capacity. Taking a decision for each customer individually may provide non-feasible and poor-quality solutions. Consequently, a global and iterative strategy is presented in which customers are selected one at a time to be assigned to a specific depot. It prioritizes the assignments of those customers that have associated a relatively high expected benefits only for a particular depot, and is based on the procedure developed in Juan et al. (2015c). In particular, the following steps are proposed:

   - For each depot k and customer i,
     - Compute the expected benefits μ_i^k as the difference between the predicted expenditure p_i^k and the distribution costs c_i^k (computed as the cost of moving from k to i).
     - Compute the difference between the expected benefits of assigning i to k and the maximum expected benefits of assigning i to a depot l other than k, i.e.:

$$s_i^k = \mu_i^k - max_{l \in V_d \setminus \{k\}} \mu_i^l \quad \forall i \in V_c, \ \forall k \in V_d$$

This measure is referred to as "marginal savings". Accordingly, s_i^k will be high in the case customer i reports relevant expected benefits only if assigned to k, low (in absolute terms) if the expected benefits are similar for k and at least one other depot, presenting both depots the highest expected benefits, and very low (negative) when there is at least one depot where the expected benefits are larger than those estimated for k.

- For each depot k, create a priority list of customers and sort it in descending order according to the marginal savings s_i^k.
- Create a list of unassigned customers. Then, select a depot and choose the next customer to assign from its priority list. Update the list of unassigned customers and repeat these steps while there are unassigned customers. Different policies may be applied to determine which depot selects the next customer, as: *(i)* allowing the depot with the highest remaining capacity to choose, *(ii)* using a round robin-based criterion, or *(iii)* selecting it randomly.

5. Routing. Having an assignment map, the MDVRP can be solved as a set of independent CVRPs. However, the most important challenge when addressing a MDVRP instance is the interrelation between assignation and routing. Therefore, algorithms are required to take the decisions associated to both phases 'simultaneously'. Thus, instead of finding an optimal or near-optimal solution for the customer-to-depot assignment phase and then use this unique solution as a starting point to solve the routing phase, an iteration process combines 'good' and fast computed solutions for the first stage with 'good' and fast computed solutions for the second one in order to find a near-optimal solution for the overall problem.

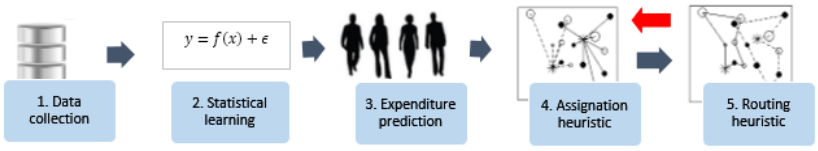


FIGURE 6.6: Scheme of the proposed approach for the MDVRP-HD.

Detailed algorithm

Figure **??** summarizes the proposed approach, highlighting the main differences between the classical version of the problem and the proposed one.

Since the phase of data collection is company-specific, it is assumed to be already done. The second and the third phases are related to the development and use of predictive statistical learning models. First, the database of existent customers is split into two subsets: a training set, which will be used to build the models, and a test set, to assess their performance. These subsets are generated by means of random sampling: 75% of customers are assigned to the training set and 25% to the test set. Having different alternatives to explore the relationship between expenditure and customers' features, three well-known methodologies are employed in the experiments: multiple linear regression (MLR), multi-layer feedforward network (MFN), and model tree.

- Regarding MLR, the ordinary least squares method is applied to estimate the parameters, and the stepwise regression approach with a bidirectional elimination procedure is chosen to perform the variable selection.
- The MFN with one hidden layer is considered. The number of hidden units (4, 5, 6, 7, or 8) and the decay value for regularization (0.2, 0.3, 0.4, 0.5 or 0.6) are set using 10-fold cross validation based on the metric R^2 (Kuhn, 2008). The back propagation method is employed to estimate the parameters.
- The algorithm selected to implement a model tree is the standard M5P (Wang and Witten, 1996).

The mean squared error (MSE) for each model (the number of models is the number of depots multiplied by the number of methodologies tested) using the same problem instance is computed. The total MSE is computed by aggregating the values of the models corresponding to the same methodology. In the experiments, the methodology selected is the one with the lowest total MSE. Thus, during the third phase, the expenditure that each new customer would make if he/she was assigned to each one of the depots is predicted using the selected methodology and the customer's features.

For the assignation and the routing phases, an existing methodology described in Juan et al. (2015c) has been adapted. The authors propose an efficient algorithm based on the ILS metaheuristic framework. Firstly, an initial solution is generated assigning customers to depots according to the marginal savings (only the distribution costs are considered) and designing the routes by implementing the classical CWS heuristic. Afterwards, an iterative procedure is started in which the base solution (the initial solution in the first iteration) is perturbed. If the new solution is better than the base solution, then the latter is replaced. In case no improvement is achieved, a Demon-based acceptance criterion is considered to avoid entrapment at local optimum. These steps are repeated until a termination condition is met. Finally, the top best solutions are improved by means of a post optimization process, and the best one is returned. The described algorithm includes biased randomization techniques to further diversify the search (Juan et al., 2009). They are implemented both in the assignation phase, to randomize the sorted priority list of customers of each depot in such a way that the reasoning behind the sorting is not erased but many orderings are provided, and in the routing phase, where the CWS heuristic is randomized.

6.4.2 Computational experiments

An algorithm based on the described approach has been implemented and employed to solve a number of generated instances. The computational experiments compare the results of our approach for the analyzed version of the MDVRP and for the classical version (i.e., the one assuming homogeneous depots).

Set of instances

A total of 15 instances have been generated. Each of them consists in three datasets: the first two gather data concerning existent and new customers, respectively, and the third includes depots' locations and information related to restrictions. Regarding data of existent customers, four variables have been created: age (a discrete variable following a uniform distribution with parameters 16 and 80), sex (a categorical variable with two equally probable values), estimated income (it follows a normal distribution with a mean of 1500 and standard deviation of 300), and preferred article (a categorical variable including four equally probable values). Initially, each customer has been assigned to his/her closest depot, while the expenditure level has been determined by a given function that depends on the depot, the aforementioned variables and a white noise term. For a total of 100 new customers, the variables age, sex, estimated income and preferred article have been generated using the same distributions. Customers' and depots' locations have been randomly generated in a square of 100 x 100. In order to simplify the instances' generation, Euclidean distances are employed as distribution costs. Different values have been chosen for the number of depots, existent customers and vehicles, the maximum cost per route and vehicles' capacity. This information is shown in Table **??**.

Test

Each instance has been adapted by modifying the expenditure of existent customers to analyze the following scenarios: *(1)* low ratio (LR), the average ratio between average expenditure of existent customers and average distribution costs is similar; *(2)* medium ratio (MR), average expenditure is relatively higher than average distribution costs; and *(3)* high ratio (HR), average expenditure is much higher than average distribution costs. The target ratio has been reached multiplying expenditures by a coefficient. The analysis of these scenarios will allow the comparison of the expected benefits (expected incomes, defined as the sum of predicted expenditures, minus distribution costs) associated to solutions considering only distribution costs and those taking into account also customer preferences (predicted expenditure), thus exploring the consequences of having different weights of expenditure in the objective solution. For the first scenario, it is expected that the gap

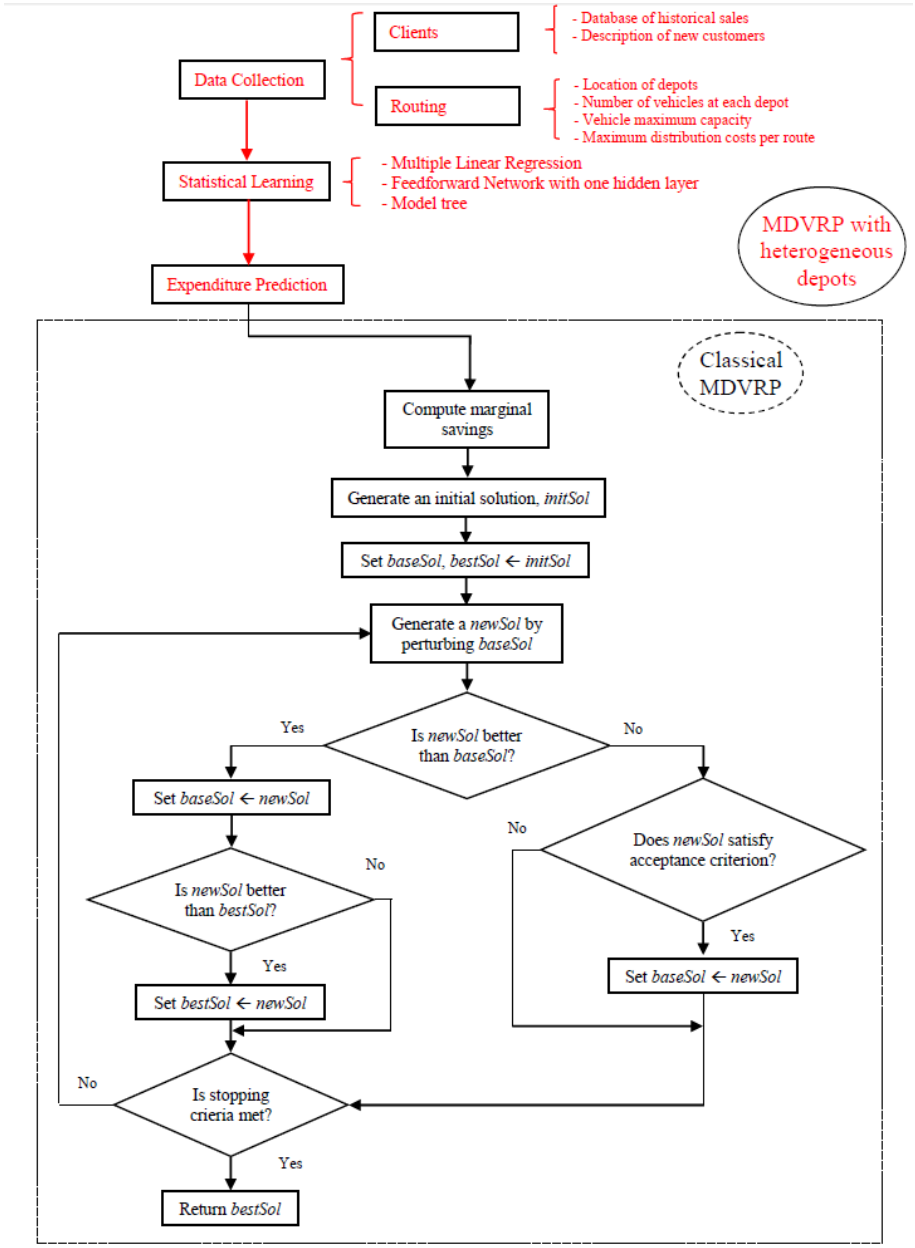


FIGURE 6.7: Flowchart of the proposed approach for the MDVRP-HD.

between distribution costs will be low (i.e., solutions are expected to be relatively similar). Likewise, it is expected that this gap will be higher as the ratio increases. Similarly, it is also expected that the higher the ratio, the higher the gap between the expected benefits of the solutions. The code has been implemented with Java and R (Team, 2008). The ILS process runs for 4,000 iterations, and all executions are solved for 10 different seeds. Only the best values obtained after the 10 runs are reported.

Results

Tables **??**, **??** and **??** show the results. The information gathered is the following: instance name; methodology selected for prediction; distribution costs, expected incomes, expected benefits and time associated to the best solution found considering only distribution costs (classical MDVRP) and to the best solution found when maximizing expected benefit (MDVRP with heterogeneous

Instance	Numb. depots	Numb. existent cust.	Numb. vehicles	Vehicle capacity	Max. cost
1	3	300	3	250	200
2	3	300	3	225	200
3	3	300	3	225	150
4	3	300	3	225	200
5	3	300	3	200	150
6	3	400	3	350	225
7	3	400	3	300	200
8	3	400	3	200	175
9	5	400	4	325	175
10	5	400	4	200	150
11	5	400	4	275	175
12	5	400	4	275	150
13	5	400	4	225	200
14	5	400	4	175	125
15	5	400	4	250	175

TABLE 6.7: Description of the generated instances.

depots); and gaps between distribution costs, expected incomes and expected benefits of both solutions.

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.1	MLR	898.6	961	62.4	82	930.6	1006	75.4	123	31.9	45.0	13.1
p02.1	M5P	834.3	943	108.7	112	834.5	947	112.6	335	0.1	4.0	3.9
p03.1	MFN	944.0	911	-33.0	143	964.4	939	-25.4	159	20.4	28.0	7.6
p04.1	MFN	891.8	852	-39.8	79	923.4	884	-39.4	165	31.6	32.0	0.4
p05.1	MFN	909.7	824	-85.7	189	914.4	829	-85.4	66	4.8	5.0	0.2
p06.1	MFN	868.5	1425	556.5	655	870.2	1429	558.8	613	1.7	4.0	2.3
p07.1	MFN	923.4	1073	149.6	103	925.7	1093	167.3	383	2.3	20.0	17.7
p08.1	M5P	898.2	867	-31.2	105	900.9	872	-28.9	122	2.7	5.0	2.3
p09.1	MLR	1039.2	2008	968.8	91	1127.5	2218	1090.5	33	88.3	210.0	121.7
p10.1	MFN	1029.6	1404	374.4	63	1062.5	1462	399.5	40	32.9	58.0	25.1
p11.1	MLR	880.7	1469	588.3	47	939.1	1609	669.9	464	58.4	140.0	81.6
p12.1	MFN	1858.4	1699	-159.4	108	1864.2	1709	-155.2	328	5.8	10.0	4.2
p13.1	MLR	1428.3	1495	66.7	437	1568.0	1691	123.0	144	139.6	196.0	56.4
p14.1	MFN	930.0	1163	233.0	43	930.0	1163	233.0	40	0.0	0.0	0.0
p15.1	M5P	1268.1	1401	132.9	374	1375.0	1512	137.0	59	107.0	111.0	4.0
								Average		35.2	57.9	22.7

TABLE 6.8: Table of results for the MDVRP-HD instances considering a low ratio.

6.4.3 Analysis of results

Given the flexibility of neural networks, and despite the basic topology and parameter fine-tuning, and the medium size of the training set, they have been selected to solve more than half of the instances (57.8%). MLR has provided the best results in a high number of cases (31.1%).

The gaps related to the distribution costs and the expected incomes are strictly positive except in one case. It confirms the trade-off decision-makers face between both measures; that is to say, higher distribution costs are required to obtain an increase in expected incomes. Regarding the gap of expected benefits, it is strictly positive for all instances except for two where both solutions are equal. Therefore, attempting to achieve the highest benefits studying only distribution costs in instances with heterogeneous depots results in sub-optimal solutions. As expected, all average gaps increase with the ratio, i.e., the difference between solutions (in terms of distribution costs, expected incomes or expected benefits) is positively correlated to the average expenditure for fixed average distribution costs. However, this rule does not apply for all cases. In some of them, despite the fact that the gap of expected incomes increases, so does the gap of distribution costs. As a consequence, the gap of expected benefit may be reduced.

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.2	MLR	925.3	1383	457.7	277	978.0	1483	505.0	173	52.7	100.0	47.3
p02.2	MLR	901.2	1334	432.8	301	921.9	1385	463.1	254	20.7	51.0	30.3
p03.2	MLR	959.3	1405	445.7	134	979.1	1438	458.9	89	19.8	33.0	13.2
p04.2	MFN	942.5	1280	337.5	124	947.8	1292	344.3	101	5.3	12.0	6.7
p05.2	MFN	919.0	1264	345.0	51	921.3	1269	347.8	221	2.3	5.0	2.7
p06.2	MFN	945.6	2103	1157.4	106	948.6	2122	1173.4	327	3.1	19.0	15.9
p07.2	MFN	962.8	1581	618.2	394	992.3	1617	624.7	139	29.5	36.0	6.5
p08.2	MFN	969.9	1302	332.1	300	969.9	1302	332.1	296	0.0	0.0	0.0
p09.2	MFN	1169.6	2897	1727.4	36	1336.1	3335	1998.9	173	166.5	438.0	271.5
p10.2	MFN	1165.1	2109	943.9	161	1222.9	2222	999.1	97	57.8	113.0	55.2
p11.2	MLR	1001.8	2212	1210.2	80	1054.4	2288	1233.7	253	52.5	76.0	23.5
p12.2	MFN	1050.0	2571	1521.0	75	1070.5	2620	1549.5	41	20.6	49.0	28.4
p13.2	MLR	1633.4	2178	544.6	106	1778.2	2446	667.8	270	144.8	268.0	123.2
p14.2	MFN	1020.2	1703	682.8	63	1026.8	1717	690.2	67	6.6	14.0	7.4
p15.2	M5P	1419.6	2090	670.4	69	1560.2	2257	696.8	106	140.5	167.0	26.5
								Average		48.2	92.1	43.9

TABLE 6.9: Table of results for the MDVRP-HD instances considering a medium ratio.

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.3	MLR	1060.3	1930	869.7	199	1153.7	2132	978.3	42	93.4	202.0	108.6
p02.3	M5P	1070.7	1803	732.3	253	1097.0	1864	767.0	174	26.3	61.0	34.7
p03.3	MFN	1042.7	1864	821.3	23	1067.1	1923	855.9	162	24.4	59.0	34.6
p04.3	MFN	1043.2	1701	657.8	54	1080.5	1755	674.5	393	37.2	54.0	16.8
p05.3	MFN	994.0	1621	627.0	174	1011.0	1657	646.0	68	17.0	36.0	19.0
p06.3	MFN	1068.1	2856	1787.9	109	1102.7	2906	1803.3	208	34.6	50.0	15.4
p07.3	MFN	1064.1	2115	1050.9	152	1081.2	2139	1057.8	71	17.1	24.0	6.9
p08.3	M5P	1069.6	1741	671.5	32	1069.6	1741	671.5	261	0.0	0.0	0.0
p09.3	MLR	1420.5	4269	2848.5	37	1690.6	4825	3134.4	138	270.1	556.0	285.9
p10.3	MFN	1434.8	2913	1478.2	113	1734.8	3396	1661.2	33	299.9	483.0	183.1
p11.3	MLR	1238.0	3020	1782.0	25	1486.3	3407	1920.7	265	248.3	387.0	138.7
p12.3	MFN	1195.7	3385	2189.3	37	1216.1	3452	2235.9	125	20.3	67.0	46.7
p13.3	MLR	1843.3	2801	957.7	79	2321.4	3387	1065.6	101	478.1	586.0	107.9
p14.3	MFN	1198.9	2297	1098.1	17	1251.0	2351	1100.0	23	52.1	54.0	1.9
p15.3	M5P	1416.0	2086	670.0	164	1595.5	2311	715.6	210	179.5	225.0	45.5
								Average		119.9	189.6	69.7

TABLE 6.10: Table of results for the MDVRP-HD instances considering a high ratio.

The results are summarized in Figures **??**. The boxplots on the left show the expected benefits per scenario and version of the problem: considering heterogeneous depots (rich) and assuming homogeneous ones (traditional). Even if the medians associated to each ratio level do not differ significantly, the third and second quartile values do present a higher value for the extended version of the problem. This behavior is caused by the long right tails of the corresponding distributions, which indicate that for some instances the rich version results in better solutions in terms of expected benefits. The second figure displays the variables in which expected benefits are decomposed per scenario and considering the rich version. It can be observed that differences of expected benefits between scenarios are mainly due to differences between expected incomes.

6.5 Sustainable urban freight transport

This section studies a MDVRP considering the sustainability concept as optimality criteria. The three-axis of sustainability (measured as economic, environmental and social impacts) are represented by traveling distances and times, carbon emissions and risk of accidents. These measures are monetized and aggregated. Several studies have addressed the economic impacts as a variable mainly influenced by traveling distances; therefore most existing models seek to minimize them. However, doing this does not guarantee the minimum impact because many elements such as congestion, speed limits, traffic signs and vehicles crashes make longer the time of the distribution

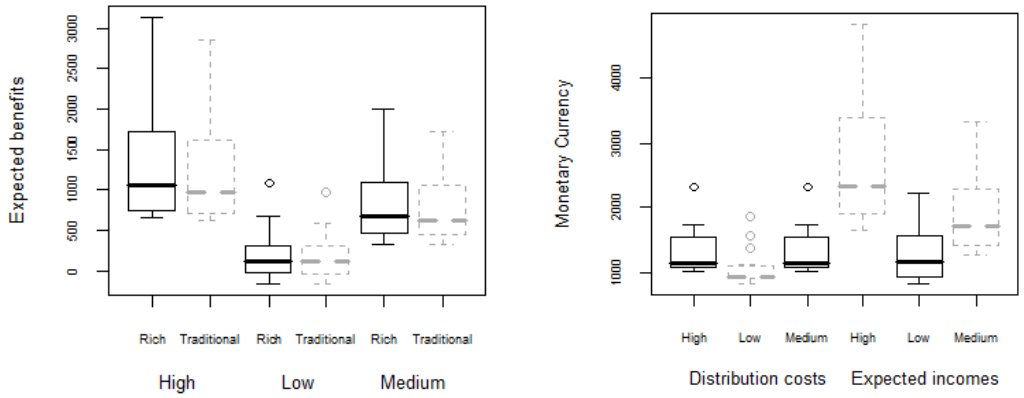


FIGURE 6.8: Boxplots of the expected benefits for the MDVRP-HD instances per scenario and version (left), and of the distribution costs and expected incomes for the rich version (right).

routes (Wang et al., 2016). In fact, the shortest paths in urban zones tend to have more traffic signs since these are the most frequented and, as a result, main streets may be the slowest paths.

- *Economic dimension:* It is composed by the classical measures total traveling times and distances, which are monetized based on the driver wage, vehicle fixed cost and oil price.
- *Environmental dimension:* CO2 emissions estimates assume that the internal combustion process of vehicles burns the carbon of the fuel and it is released as carbon dioxide. Thus, emissions are assumed to depend on fuel consumption. The fuel consumption is estimated as suggested in Kuo (2010) and Zhang et al. (2015).
- *Social dimension:* Accidents are an externality caused by speed variations on roads, among other factors. These variations represent the state and stability of the roads, and are associated to an accident risk for pedestrian and vehicles (Wang et al., 2016).

6.5.1 Methodology

The methodology proposed (Algorithm **??**) is based on the VNS metaheuristic. The inputs are the problem instance and the number of neighborhoods (K). It is usual to set K to two or three, and to design nested neighborhoods.

First, an initial solution is generated and stored in *initSol* and *baseSol*. Then, the cost of all the impacts associated are computed. *bestSol* will store the best solution found. An outer loop is started, which sets the current neighborhood to the first one. Inside, another loop builds and assesses new solutions. Within this loop, the base solution is initially shaken (i.e., it is partially destroyed and reconstructed in a random way), generating a solution from the k-th neighborhood of *baseSol*. The total cost of this solution (*newSol*) is computed (Algorithm **??**). The variable *rpd* measures the relative percentage difference between the total cost of *newSol* and *baseSol*. If there is an improvement (i.e., $rpd < 0$), a local search is applied to *newSol*, the resulting solution is copied into *baseSol*, and the current neighborhood is set to the first. In addition, *bestSol* is updated if it applies. This constitutes a descendent phase aimed to find a local minimum. Otherwise, *newSol* is accepted and the current neighborhood is set to the first with a probability of $exp(-rpd)$. This acceptance criterion, first proposed in Hatami et al. (2015), aims to avoid entrapment at local optimum. In case of not accepting *newSol*, the next neighborhood is analyzed (i.e., k is set to $k+1$). The inner loop is executed until the last neighborhood is explored (i.e., $k = K$). Finally, *bestSol* is returned.

The generation of solutions for the MDVRP has two sequential and interrelated stages: *a)* the assignment of customers to depots, and *b)* the design of distribution routes for each depot. Both stages employ biased randomization techniques. The first stage relies on a measure called "marginal savings" (Juan et al., 2015c), which is computed for each pair depot-customer as follows: the distance between each depot and the customer is obtained, and the difference of assigning the customer to the specific depot instead of the closest depot among the others is computed. A

Algorithm 8 Approach for the MDVRP considering externalities.

```
1:  procedure MDVRP WITH SUSTAINABILITY INDICATORS (inputs, impactsParameters)
2:      initSol ← genInitSol (inputs) # generate solution based on the BR-CWS heuristic
3:      baseSol ← clone (initSol)
4:      computeTotalCost(baseSol, impactsParameters)
5:      bestSol ← clone (baseSol)
6:      while (stopping criterion is not met) do
7:          k ← 1
8:          while (k ≤ K) do
9:              newSol ← shake(baseSol, k) # destruction-construction stages
10:             computeTotalCost(newSol, impactsParameters)
11:             rpd ← (getTotalCost(newSol) - getTotalCost(baseSol))/getTotalCost(baseSol)· 100
12:             if (rpd < 0) then # newSol improves baseSol
13:                 newSol ← localSearch(newSol)
14:                 baseSol ← newSol
15:                 k ← 1
16:                 if (getTotalCost(newSol) - getTotalCost(bestSol) < 0) then
17:                     bestSol ← newSol
18:                 end if
19:             else
20:                 u ← generateU()
21:                 if (u < exp(−rpd)) then #acceptance criterion
22:                     baseSol ← newSol
23:                     k ← 1
24:                 else
25:                     k ← k + 1
26:                 end if
27:             end if
28:         end while
29:     end while
30:     bestSol ← localSearch(bestSol)
31:     return bestSol
32: end procedure
```

Algorithm 9 Function to monetize the impacts of a given solution.

```
1:  procedure COMPUTE TOTAL COST(MDVRPSol, impactsParameters)
2:      distance ← 0
3:      time ← 0
4:      emissions ← 0
5:      social ← 0
6:      for each (cvrpSol in MDVRPSol) do
7:          distance ← distance + getDistance(cvrpSol)
8:          time ← time + getTime(cvrpSol)
9:          for each (edge in cvrpSol) do
10:             emissions ← emissions + getDistance(edge)/getKPL(edge)
11:             social ← social + getDistance(edge) · getLoad(edge, cvrpSol)
12:         end for
13:     end for
14:     distanceCost ← distance· getDistUnitCost(impactsParameters)
15:     timeCost ← time· getTimeUnitCost(impactsParameters)
16:     emissionsCost ← emissions· getEmissionsUnitCost(impactsParameters)
17:     socialCost ← social· getSocialUnitCost(impactsParameters)
18:     totalCost ← distanceCost + timeCost + emissionsCost + socialCost
19:     return totalCost
20: end procedure
```

priority list of customers is created for each depot and sorted according to the marginal savings. Thus, high marginal savings are prioritized, since assigning the corresponding customer to another depot (which would be farther) could lead to a poor-quality solution. These lists are randomized

assigning probabilities according to a geometric distribution. Three different policies are iteratively applied to choose the depot to select the next customer to be assigned: *i)* all depots choose the first node in their list at a time, following consecutive turns (known as round robin criteria); *ii)* randomly; *iii)* the depot with the highest remaining capacity is selected. Thus, using biased randomization and different policies promotes the generation of different assignation-maps. The second stage is based on the randomized version of the CWS heuristic (Juan et al., 2011a), which also depends on a geometric distribution applied to the savings to iteratively choose one merge among all possible. However, the classical distance-based savings are replaced by "rich savings" including all costs.

The performance of each solution is computed and the sum of the costs associated to the impacts considered: economic, environmental and social. The shaking procedure randomly selects a percentage p_k of customers to be assigned to a different depot. Afterwards, the procedure to construct solutions is applied to repair the solution. This movement is introduced to diversify the search. The search is guided by the base solution, since the shaking procedure applied at each iteration works with that solution. It is set to the initial solution at the beginning and replaced by the new solution if the acceptance criterion is met. The stopping criterion is based on the number of iterations. Two local searches are used: the first is applied to solutions improving the current base solution and is based on the classical 2-opt operator defined for the CVRP (Lin, 1965), while the second is a routing extensive improving search described in Juan et al. (2011a), and applied only to the best solution found at the end.

6.5.2 Computational experiments

The algorithm proposed has been implemented in JAVA and run on a personal computer with 8 GB of RAM and an Intel Core i7 of 1.8 GHz. In order to test it, illustrate its use and the analysis of results that may be carried out, 4 MDVRP benchmark instances (p10, p11, p12 and p13) called here instance 1, 2, 3 and 4, respectively, are employed. They have been extensively used (see Vidal et al., 2012; Escobar et al., 2014).

Each instance has been adapted as follows. Vehicles' efficiency parameters are based on a type of light duty vehicle used for freight distribution in urban zones. The cost for CO2 emissions (0.1 USD/L) is suggested by Zhang et al. (2015). Regarding the time cost, it is defined by Koç et al. (2014) as the sum of a vehicle fixed cost and driver wage, which are set to 1.4 USD/h and 6.3 USD/h, respectively. The distance cost is based on the price of fuel (1.1 USD/L) and the average miles per fuel liter (5.56 km/L). Delucchi and McCubbin (2010) propose an the interval $[1 \cdot 10^{-4}, 1.3 \cdot 10^{-3}]$ USD per kg-km for the coefficient to estimate the social cost. Without loss of generality, times t_{ij} are generated from distances d_{ij} using this formula $t_{ij} = \alpha \cdot d_{ij} + \varepsilon_{ij}$, where α is a constant based on an estimated speed ($\alpha^{-1} = 35$ km/h) and ε_{ij} represents external factors that define the correlation between traveling time and distance. It is set to follow a truncated normal distribution with a lower bound and mean equal to 0 and a standard deviation equal to 3.5, 2, and 0.5. These deviations are set in order to get a correlation around 0.5, 0.7 and 0.9, which may represent a high, medium and low congested zone, respectively. Thus, three scenarios are generated per instance. For example, for $d_{ij} = 10$ km, t_{ij} fall in the following intervals considering a probability of 95%: (0.59, 1.69), (0.64, 5.06), and (0.69, 8.42).

Each instance has been solved 10 times (employing a different seed) and only the best solutions are reported. 300,000 iterations are considered. The parameter fine-tuning is performed by using design of experiments and testing reasonable ranges. The parameters for the geometric distributions related to the allocation and the routing process are randomly chosen in the intervals (0.5, 0.8) and (0.1, 0.2), respectively. The degree of shaking, which defines the neighborhoods, is set to 10%, 30% and 50% (for the first, second and third neighborhood, respectively).

The experimentation process consists in analyzing how the solution space changes according to the optimization criterion and how it influences the other indicators. Thus, five options are considered: optimization criterion is based on minimizing each component of the objective function or the sum of them. In a real-life application, the choice will depend on the particular interests of the decision-maker.

6.5.3 Analysis of results

This section analyzes the solutions found considering all the indicators or each one of them as optimization criterion. This comparison aims to determine a solution subspace representing an equilibrium between the economic, environmental, and social dimensions.

Table **??** shows the total cost of the best solutions found according to the objective pursued for each instance and scenario. As described before, the total cost is computed as the sum of the costs associated to traveling distance, traveling time and CO2 emissions, and the social cost. In instance 1 and 2, the solution minimizing the traveling time matches the solution minimizing the total cost, which means that these objectives converge to the same solution. Similarly, the same solution ensures the minimum traveling distance cost and emissions cost (this is due to the way in which the emissions are estimated). Obviously, the total cost is higher in the zones where the traveling time and the traveling distance have a low correlation (i.e., in congested zones, based on the description of the scenarios). The solution with the minimum social cost is the most expensive, because the other costs are significantly increased.

TABLE 6.11: Total cost by scenario, instance and optimization criterion.

		Scenarios			
		Low	Medium	High	
Instance	Objective	Total cost	Total cost	Total cost	Run Time (s)
1	Total cost	7597.4	5669.1	3763.3	1665.8
	Distance	8601.8	6054.1	3763.3	1572.9
	Time	7597.4	5770.8	3867.3	1688.3
	CO2 emissions	8601.8	6054.1	3763.3	1572.9
	Social cost	8686.3	6393.7	3977.7	1485.0
2	Total cost	7625.8	5979.4	3645.0	1368.4
	Distance	9087.9	6392.3	3645.0	1625.2
	Time	7625.8	6060.9	3741.2	1603.2
	CO2 emissions	9087.9	6392.3	3645.0	1625.2
	Social cost	9096.5	6651.7	3898.9	1130.3
3	Total cost	2475.6	1913.3	1197.9	200.2
	Distance	2949.0	1944.5	1199.6	190.8
	Time	2475.6	1949.0	1197.9	198.5
	CO2 emissions	2949.0	1944.5	1199.6	190.8
	Social cost	2979.8	1999.8	1241.6	186.0
4	Total cost	2757.8	1904.3	1217.0	185.9
	Distance	2871.1	1913.3	1217.0	187.3
	Time	2813.3	1927.9	1222.7	189.3
	CO2 emissions	2871.1	1913.3	1217.0	183.9
	Social cost	3144.2	2103.1	1385.6	182.8

Regarding the social cost, it is important to determine the customer sequence and the direction of the route. Figure **??** illustrates and quantifies their effect on the total cost of a given route. Accordingly, high-quality solutions visit first the customers with higher demands, minimizing the amount of freight transported over long stretch of roads. On the other hand, the scenario influences the total cost. Table **??** suggests that the gap between solutions with minimum total cost and minimum social cost is higher in congested zones. This happens because minimizing the social cost involves reducing the traveling distance, which leads to optimize also the traveling time if there is a high correlation between time and distance.

Figures **??** and **??** provide information regarding the behavior of solutions for each scenario. The first represents the average weight of each cost component per scenario considering the four instances. It can be observed that traveling time represents the main cost and its magnitude is the most sensitive to the scenario. Figure **??** shows the ranges of total cost and its components per scenario for instance 1. In this case, the time cost increases at a higher rate than the distance cost, which causes differences among scenarios.

Table **??** shows the cost of each indicator when the main objective is to minimize the total cost for all instances and scenarios. The gaps reflect the difference between the solution with minimum total cost and the best solution for each indicator. For example, the solution with the minimum total cost for instance 1 in the low scenario has a social cost 9.49% higher than the best solution found when the objective is to minimize the social cost. This table demonstrates that the solution

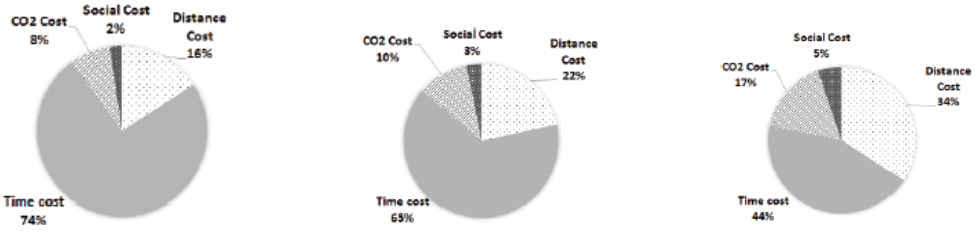
Route	1	2	3	4	5	6	7	8	9	10	11	1
Demand		70	71	56	14	50	79	57	50	22	14	
Distance		1.2	0.5	0.7	0.9	1.3	0.8	0.9	0.3	0.7	0.6	
a_{ij}		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	
Social Cost		555.5	198.2	236.0	263.1	359.0	166.5	124.4	25.8	24.1	7.8	

Total Cost **$1,961**

Route	1	11	10	9	8	7	6	5	4	3	2	1
Demand		14	22	50	57	79	50	14	56	71	70	
Distance		0.6	0.7	0.3	0.9	0.8	1.3	0.9	0.7	0.5	1.2	
a_{ij}		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	
Social Cost		270.5	314.2	134.1	345.4	255.0	344.5	194.1	135.9	67.7	80.5	

Total Cost **$2,142**

FIGURE 6.9: Effect of the customer sequence and the direction for a given route.



(A) Low Scenario (B) Medium Scenario (C) High Scenario

FIGURE 6.10: Weight of each sustainability component in the total cost by scenario considering all instances.

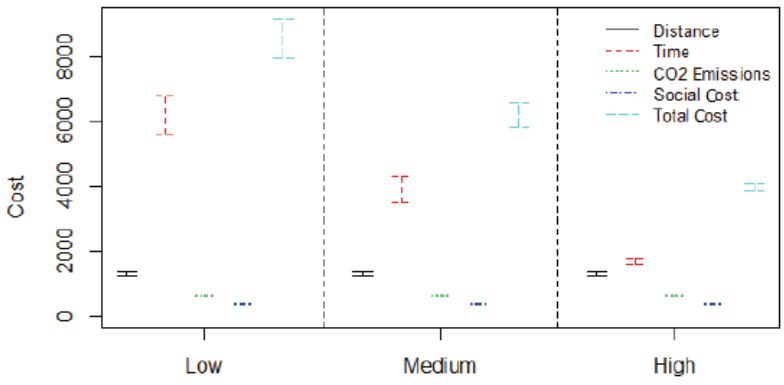


FIGURE 6.11: Total cost and component per scenario for instance '1'.

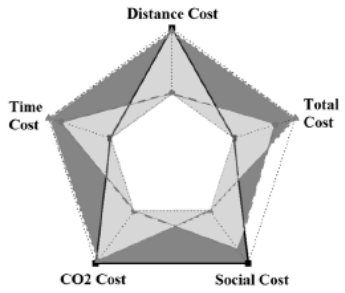
with the minimum total cost does not tend to be the best when applying another optimization criterion.

Figure **??** displays radar plots for instances 1 and 4, and the scenarios low and high using the best solutions found for each indicator and the total cost. These plots identify the desirable and sustainability regions. The desirable region is defined as the union of the subspaces associated to the solutions. Solutions falling outside the figure may be discarded since they are dominated by at least one solution (i.e., there is at least one solution with the same or better values for all indicators). Similarly, the intersections of at least two solution subspaces form the sustainability region, which contains the best solutions, i.e., solutions that achieve a suitable balance considering

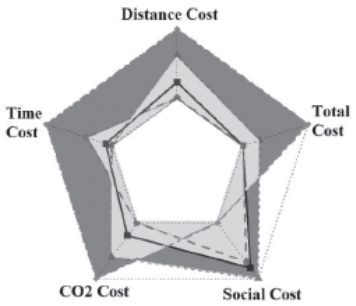
Table 6.12: Comparison among solutions for each instance and scenario.

Objective: Minimizing Total cost										
Scenario		Instance	Traveling distance		Traveling time		CO2 emissions		Social cost	
		(V, C, D)	Cost	Gap (%)	Cost	Gap (%)	Cost	Gap (%)	Cost	Gap (%)
Low	1	(8,249,4)	1386.35	-12.88%	5136.15	0.00%	672.10	-12.88%	402.78	-20.37%
	2	(6,249,5)	1376.06	-14.96%	5192.52	0.00%	667.11	-14.96%	390.11	-21.38%
	3	(5,80,2)	560.84	-23.09%	1624.93	0.00%	271.89	-23.09%	17.90	-31.55%
	4	(5,80,2)	460.19	-5.66%	2059.82	-1.16%	223.10	-5.66%	14.69	-28.31%
		Average	945.86	-14.15%	3503.36	-0.29%	458.55	-14.15%	206.37	-25.40%
Medium	1	(8,249,4)	1328.31	-9.07%	3325.69	-2.38%	643.96	-9.07%	371.14	-13.58%
	2	(6,249,5)	1312.04	-10.81%	3671.20	-2.96%	636.08	-10.81%	360.05	-14.82%
	3	(5,80,2)	434.16	-0.65%	1254.84	-12.01%	210.48	-0.65%	13.78	-11.05%
	4	(5,80,2)	442.88	-1.97%	1232.64	-0.26%	214.71	-1.97%	14.09	-25.23%
		Average	879.35	-5.63%	2371.09	-4.40%	426.31	-5.63%	189.77	-16.17%
High	1	(8,249,4)	1204.67	0.00%	1626.95	-4.72%	584.02	0.00%	347.65	-7.74%
	2	(6,249,5)	1177.17	-0.59%	1563.70	-3.79%	570.69	-0.59%	333.45	-8.03%
	3	(5,80,2)	435.35	-0.92%	538.16	0.00%	211.06	-0.92%	13.33	-8.06%
	4	(5,80,2)	434.16	0.00%	558.53	-0.48%	210.48	0.00%	13.78	-23.54%
		Average	812.84	-0.38%	1071.84	-2.25%	394.06	-0.38%	177.05	-11.84%

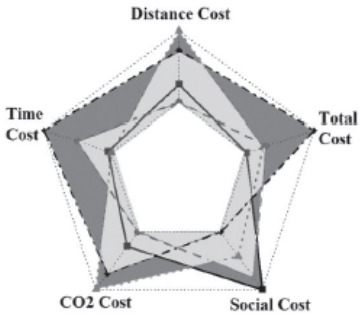
a number of indicators.



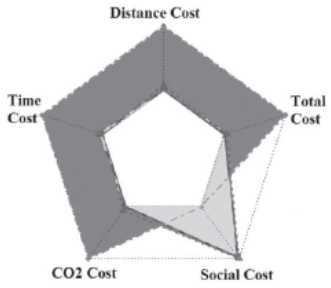
(a) Instance 1: Low Scenario



(b) Instance 1: High Scenario



(c) Instance 4: Low Scenario



(d) Instance 4: High Scenario

Figure 6.12: Solution spaces for decision-making considering sustainability indicators.

6.6 The WCP

The WCP (Figure **??**) can be described on a graph $G = (V, A)$, where the set of nodes $V = V^d \cup V^f \cup V^c \cup V^b$ includes: *(i)* a set of starting and ending depots $V^d = \{0, 0'\}$ (in practice both depots could be the same), with the starting depot being the initial location of a fleet of homogeneous vehicles $K = \{1, 2, \ldots, k\}$, each of them having a capacity C; *(ii)* a set $V^f = \{1, 2, \ldots, m\}$ describing m landfills at which collected waste must be disposed at least once before visiting the ending depot; *(iii)* a set of waste containers (customers) $V^c = \{m+1, \ldots, m+n\}$ with associated waste levels $q_i > 0$ ($\forall i \in V^c$); and *(iv)* a set $V^b = \{0^*\}$ representing a virtual lunch-break node that has to be included in each route. Each node $i \in V \setminus V^d$ has an associated time window represented by $[a_i, b_i]$ (with

$0 \leq a_i < b_i$). Necessary service times for emptying any container and the duration of the lunch break are formulated as $r_i > 0$ ($\forall i \in V^c \cup V^b$). Likewise, the set $A = \{(i, j)/i, j \in V, i \neq j\}$ describes the arcs connecting any pair of different nodes. Each pair is characterized by its respective travel costs, $c_{ij} = c_{ji} \geq 0$, and travel times, $t_{ij} = t_{ji} \geq 0$. The travel time associated with going from any node $i \in V \cup V^b$ to the virtual lunch-break node (and vice versa) is equal to zero, i.e.: $t_{i0^*} = t_{0^*i} = 0$. Notice, however, that the travel cost associated with 'crossing' the lunch-break virtual node is given by the travel cost of the origin and destination nodes, i.e.: $c_{i0^*} + c_{0^*j} = c_{ij}$. The decision variables x_{ijl} ($\forall (i, j) \in A, \forall l \in K$) equal 1 if arc (i, j) is employed by vehicle l and 0 otherwise. The aim is to minimize total travel costs.

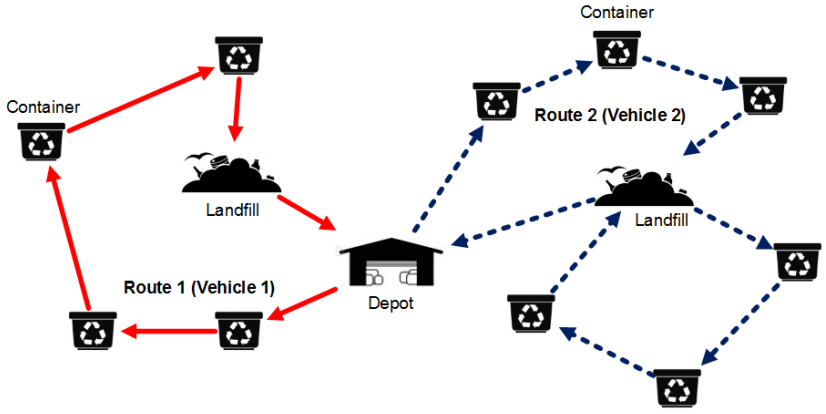


FIGURE 6.13: Representation of a WCP instance.

The following restrictions are considered: *(i)* the number of vehicles used is not predetermined, only the maximum number of available vehicles is given; *(ii)* the lunch break is automatically included in a route whenever a certain time window is reached; *(iii)* there is a maximum number of stops at containers and landfills per route; *(iv)* there is a maximum amount of waste that can be collected on a single vehicle route; and *(v)* the depot also has a time window. Methodologies for both the deterministic and the stochastic versions are presented.

6.6.1 Methodology

The WCP

A VNS metaheuristic is proposed to solve the deterministic WCP. An initial solution is obtained by applying the biased-randomized extension of the CWS heuristic. This procedure is adapted to the special case of waste collection by changing the calculation of savings values used for merging two customers i and j, originally calculated as $s_{ij} = c_{i0} + s_{0j} - c_{ij}$. In the WCP, the costs of traveling between a customer and the depot are asymmetric due to the additional landfill visit. To address this new situation, the average savings associated to each arc are employed.

Based on the initial solution $baseSol$, different neighborhood structures $N_k(\{k = 1, ..., k_{max}\})$ are created. The shaking procedures applied to create new solution structures are outlined in Table **??**. Within each neighborhood $N_k(baseSol)$, different local descent heuristics described in Table **??** are randomly applied to find the local minimum of $N_k(baseSol)$. To conclude the local search phase, a quick solution improvement procedure based on a cache memory technique (Juan et al., 2013a) is implemented: the best-known order of traveling between a set of nodes establishing a sub-route –i.e., starting at the depot or a landfill and ending at a disposal site– is stored in a hash-table data structure, thus allowing new solutions to benefit from previously constructed ones. Whenever the local search phase leads to a more competitive objective function value than that of $baseSol$, $baseSol$ is updated and k is returned to its initial value of 1. If $baseSol$ cannot be improved through the local minimum of N_k, k is incremented by 1 and the next shaking operator is applied. Once each neighborhood has been constructed ($k = k_{max}$), the process is repeated until a certain predefined stopping criterion (e.g.: time, iterations, etc.) has been reached. Note that we shuffle the list of neighborhood operators every time $k > k_{max}$. A description of the VNS procedure for the deterministic WCP can be seen in Algorithm **??**.

TABLE 6.13: Shaking operators for the WCP.

Operator (k)	Description
Customer Swap Inter-Route	Swaps two random customers between different routes.
2-Opt Inter-Route	Interchanges two chains of randomly selected customers between different routes.
Reinsertion Inter-Route	Inserts a random customer in a different route.
Cross-Exchange	Interchanges positions of 2-4 random, non-consecutive customers from different routes.

TABLE 6.14: Local search operators for the WCP.

Operator (LS-Scheme)	Description
Best Position Insertion	Reinserts the container with the highest objective function increase into the best available position of any route.
Re-allocate all	Iteratively calculates the objective function increase of each container and reinserts it at the best possible position.
Random Swaps	Randomly selects and interchanges two nodes (from the same or different routes) if the objective function improves.

The stochastic WCP

Waste levels cannot be predicted with full certainty when solving a more realistic stochastic version of the problem. The fact that actual waste levels in containers are only known when reaching designated pick-up points can lead to route failures whenever collected garbage exceeds the planned collection amount. In these cases, the collection vehicle needs to add an additional and expensive landfill visit to its route. The proposed simheuristic methodology (Algorithm **??**) allows an estimation of the solution quality of previously created outputs using the VNS metaheuristic proposed before by integrating MCS into the solution procedure.

The methodology starts by transforming the stochastic input variables into their deterministic counterpart, which is used to establish initial WCP solutions. The waste levels can be modeled according to some kind of probability distribution. This allows the (stochastic) waste levels w_i at each container i to be replaced with expected values $E[w_i]$. Using these deterministic values, an initial solution $baseSol$ is constructed. In the following, the solution quality in a stochastic environment is tested by randomly simulating the waste levels of each container i for a certain number of iterations (or simulation runs) within the predefined distribution. During each run the occurring route failure costs are estimated by penalizing situations in which vehicle capacities are reached before a scheduled landfill trip. More specifically, route failure costs are calculated as corrective actions to the predefined routes. Finally, the sum of all route failure costs of all simulation runs are divided by the number of simulation runs. Thus, the expected total costs of $baseSol$ consist not only of the deterministic routing costs, but rather in the addition of the deterministic routing costs with the expected route failure costs. At this stage a small number of iterations $shortSimIter$ is proposed. On the one hand, a larger number of simulation runs lead to more reliable estimates of the stochastic route costs. On the other hand, at this stage a shorter simulation procedure can be used to keep the computational effort through the simulation reasonable.

Once $detCosts(baseSol)$, $stochCosts(baseSol)$, and $totalCosts(baseSol)$ have been defined, new deterministic solution neighborhoods are constructed and locally improved as described previously. A newly constructed solution $newSol$ is considered as promising whenever it yields lower deterministic costs than the current base solution. The behavior of each promising solution under waste level uncertainty is then evaluated by applying a short simulation run, leading to a first estimation of the total solution costs. Whenever $totalCosts(newSol) < totalCosts(baseSol)$, the current base solution is updated and k is returned to its initial value. Furthermore, the solution is stored as elite stochastic solution. With each elite solution, a more extensive simulation run is started for $longSimIter$ iterations once the metaheuristic stopping criteria has been reached. The number of stored $eliteSols$ is limited to 10.

In addition to calculating the stochastic objective function value of promising deterministic solutions, the methodology allows the estimation of a solution reliability by considering the proportion of runs where the solution plan can be implemented without any route failure. Thus,

Algorithm 10 VNS procedure for the WCP

```
1: baseSol ← solve biased randomized CWS for the WCP
2: while stopping criteria not reached do
3:     shuffle(ListOfShakingOperators)
4:     k ← 1
5:     repeat
6:     newSol ← shake(baseSol, k)
7:     improving ← true
8:     while improving do
9:         newSol* ← localDescent(newSol, randomLSoperator)
10:        if costs(newSol*) ≤ costs(newSol) then
11:            newSol ← newSol*
12:        else
13:            improving ← false
14:        end if
15:        cacheSubRoutes(newSol)
16:        if costs(newSol) < costs(baseSol) then
17:            baseSol ← newSol
18:            k ← 1
19:        else
20:            k ← k + 1
21:        end if
22:    end while
23:    until k > k_max
24: end while
25: bestSol ← baseSol
26: return bestSol
```

the reliability $reliab_r$ of each route r of any solution S is computed as the quotient of the number of runs in which a route failure occurs divided by the total number of simulation runs, i.e. $reliab_r = simRunsWithRouteFailue/simRuns$. Notice that each route in a solution can be seen as an independent component of a series system (i.e., the proposed solution will fail if, and only if, a failure occurs in any of its routes). Therefore, the overall reliability of a solution with R routes can be computed as $\prod_{r=1}^{R} reliab_r$.

6.6.2 Computational experiments

To test the deterministic approach, the 10 WCP benchmark instances provided by Kim et al. (2006), which were later adopted by Benjamin and Beasley (2010) and Buhrkal et al. (2012), are employed. Furthermore, the clustered instances presented by Buhrkal et al. (2012) are used. A clustering procedure is applied to nodes with the same location and time windows to change the total number of nodes. The algorithm was implemented as Java application and run on a personal computer with an Intel®Xeon™CPU E5-2630 v2 @ 2.60GHz processor. The initial solutions constructed with the biased randomized version of the savings heuristic are based on a distribution parameter randomly chosen within the range (0.4, 0.5) at each solution construction step.

The results are summarized in Table **??**. Column (1) reports the BKS for each instance, column (2) the computational times (CT) in seconds, and column (3) the average results with 10 different random number seeds. The VNS metaheuristic is tested with two different stopping criteria. On the one hand, our best solution (achieved with 10 seeds) is reported in column (4). Furthermore, our average solution (5) and our best solution (6) with a stopping criterion of 300 seconds per instance are reported, as suggested by Benjamin and Beasley (2010). It can be seen that the proposed algorithm outperforms current BKSs by an average of -0.85% and -2.65%. Moreover, it reaches 9 new BKSs (11 with the extended algorithm running time).

Algorithm 11 Simheuristic approach for the WCP-SW

```
replace stochastic waste levels by expected values
baseSol ← solve biased randomized CWS for the WCP
shortSimulation(baseSol)
while stopping criteria not reached do
    k ← 1
    repeat
        newSol ← shake(baseSol, k)
        localSearch(newSol)
        if detCosts(newSol) < detCosts(baseSol) then          ▷ Solution is promising
            shortSimulation(newSol)
            if totalCosts(newSol) < totalCosts(baseSol) then
                update(eliteSols)
                baseSol ← newSol
                k ← 1
            else
                k ← k + 1
            end if
        end if
    until k > k_max
end while
for each eliteSol do
    longSimulation(eliteSol)
    estimateReliability(eliteSol)
end for
```

Table 6.15: Table of results for the WCP benchmark instances.

Instance	(1) BKS	(2) CT BKS (s)	(3) BKS average	(4) Our best sol[1]	(5) Our sol average[2]	(6) Our best sol[2]	(7) CT our best sol (s)	%-Gap (1)-(4)	%-Gap (1)-(6)
Kim102	174.5	3	176.03	158.61	158.64	154.62	5	-9.11	-11.39
Kim277	447.6	8	455.7	472.73	457.14	450.6	299	5.61	0.67
Kim335	182.1	10	196.49	189.79	187.36	184.22	298	4.22	1.16
Kim444	78.3	18	78.99	80.22	80.09	79.49	292	2.45	1.52
Kim804	604.1	72	650.65	603.17	601.14	593.2	300	-0.15	-1.80
Kim1051	2250.6	194	2387.7	2128.37	2119.50	2077.37	294	-5.43	-7.70
Kim1351	871.9	105	891.17	929.5	929.40	910.6	238	6.61	4.44
Kim1599	1337.5	252	1385.3	1184.67	1208.54	1182.58	292	-11.43	-11.58
Kim1932	1162.5	285	1192.2	1149.45	1169.95	1136.34	273	-1.12	-2.25
Kim2100	1749	356	1916.8	1595.48	1622.29	1603.93	293	-8.78	-8.29
Clustered Instances									
Kim86	174.5	3	176.6	155.68	158.35	155.68	10	-10.79	-10.79
Kim267	450.7	8	456.4	460.4	455.96	449.41	294	2.15	-0.29
Kim322	182.4	10	190.7	189.78	185.93	184.26	298	4.05	1.02
Kim444	78.6	18	79.2	80.22	80.09	79.49	292	2.06	1.13
Kim602	586.2	72	647.8	610.52	593.25	586.11	297	4.15	-0.02
Kim1011	2295.2	116	2370.5	2151.51	2131.00	2102.23	299	-6.26	-8.41
Kim536	850	105	850.9	885.83	877.69	850.46	292	4.22	0.05
Kim870	1170.2	252	1230.6	1156.15	1180.07	1145.83	286	-1.20	-2.08
Kim1860	1128.7	285	1180.9	1129.89	1154.48	1138.6	295	0.11	0.88
Kim1877	1594.2	266	1650.8	1620.89	1642.20	1604.33	186	1.67	0.64
Average	*868.44*	*122*	*908.27*	*846.64*	*849.65*	**833.47**	*257*	*-0.85*	*-2.65*

[1] Computational times per instance equal to column (2)
[2] Computational times per instance equal to column (7)

Since there is a lack of stochastic benchmark instances, the non-clustered instances of Kim et al. (2006) are used as reference. The deterministic instances are then transformed into stochastic ones by using random waste levels following a log-normal distribution with expected values equal to the original deterministic value.

Table 6.16: Comparison of elite solutions for the WCP-SW.

Elite Solutions	Best 1		Best 2		Best 3	
Instance Name	Mean	St. Dev.	Mean	St. Dev.	Mean	St. Dev.
Kim102	157.05	3.54	157.14	3.38	157.22	3.65
Kim277	498.66	4.53	499.07	4.45	499.12	4.59
Kim335	187.84	1.81	187.96	1.84	188.25	1.85
Kim444	87.79	0.84	87.80	0.79	91.35	0.82
Kim804	633.97	5.93	634.34	5.74	635.00	5.90
Kim1051	2342.85	16.67	2343.58	15.48	2345.62	16.29
Kim1351	1009.88	26.48	1012.78	26.57	1025.50	26.54
Kim1599	1290.02	24.34	1291.67	23.40	1292.07	23.83
Kim1932	1199.85	29.77	1202.21	30.50	1245.03	30.14
Kim2100	1742.47	13.97	1742.81	14.62	1748.34	13.83

The approach is tested using low ($Var[w_i] = 0.05$), medium ($Var[w_i] = 0.15$), and high variance levels ($Var[w_i] = 0.25$) concerning the waste level distribution at any container. The number of short simulation runs is set to 500, while a more extensive simulation with 5000 runs is applied only to the elite solutions. Moreover, vehicle safety stocks k are considered to better deal with unexpected demands (Juan et al., 2011b). Instead of considering the complete available vehicle capacity C in the construction of the deterministic solution, a decreased capacity $C^* = C(1-k)$ is applied. On the one hand, high levels of k will, on average, lead to higher deterministic costs, as the considered vehicle capacity during the route construction is reduced. On the other hand, it can be expected that the stochastic route failure costs will decrease. 6 different levels k are considered: 0, 0.02, 0.04, 0.06, 0.08, and 0.1. The average calculation time of all scenarios was 351.92 seconds.

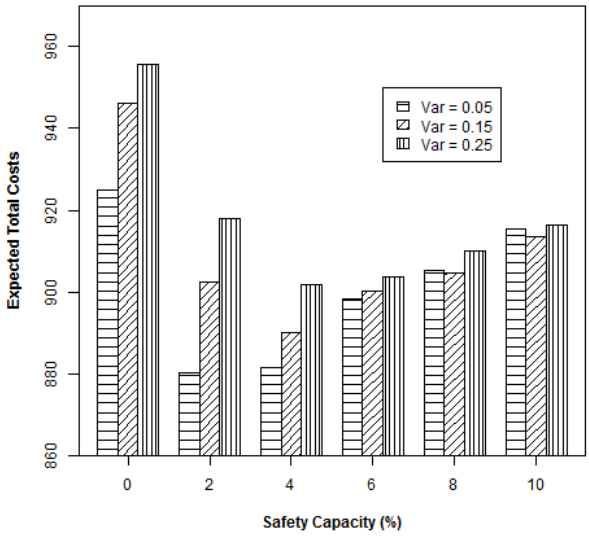
6.6.3 Analysis of results

Figure **??** shows the expected total costs and reliabilities for the average of all tested instances for each waste variance level/safety capacity factor combination. As can be observed, the highest total costs for each waste variance level is obtained when no safety capacity factor is considered as a result of high expected route failure costs. Furthermore, it can be seen that the lowest total costs over all instances for a low variance level are obtained with a safety capacity factor of 2%. For medium and high waste variance, a safety capacity factor of 4% seems to yield the most promising results concerning total costs. As expected however, the reliability levels increase for all variance levels as the vehicle safety capacity is increased. It can also be concluded that the inclusion of only a small safety capacity already significantly increases reliability levels (up to around 60% in the most extreme case).

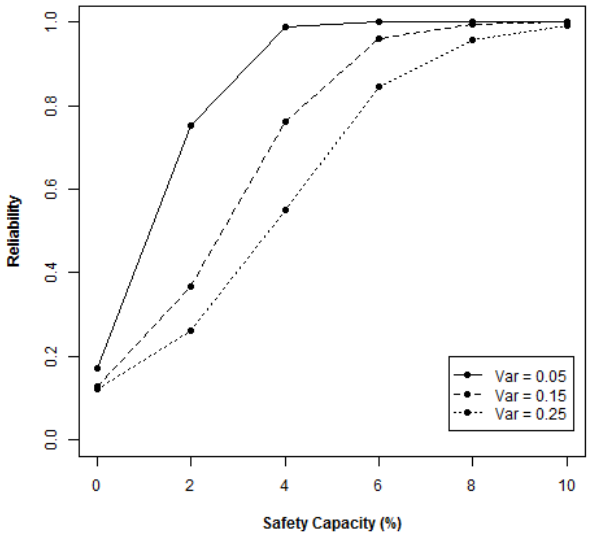
A more detailed risk analysis is done in Figure **??**, which shows a boxplot of the long simulation outputs for the three most competitive elite solutions of the 'Kim277' instance. In this specific case the first solution seems to be the most promising one, as it has the lowest mean and the lowest quartiles. However, this is not necessarily always the case. In Table **??**, the mean and standard deviation of the results from the long simulation concerning total costs of the three best solutions of each instance are listed. It can be concluded that the solution with the lowest mean does not always have the lowest standard deviation. This information can be used by decision-makers to select the solution that he/she prefers according to his/her risk preference. In a similar manner, our solution approach allows the consideration of different risk-aversion levels of decision takers by comparing solutions with different safety capacity levels. A more risk-averse route planner will choose to construct routes with higher safety capacity levels, which typically lead to higher routing costs while experiencing lower route failure, and vice versa.

6.7 The HSAVRP-SD

The HSAVRP-SD is defined over a complete graph $G = (N, A)$, where $N = \{0, 1, \ldots, n\}$ is a set of nodes representing the depot (node 0) and the n customers. Each node $i \in N$ has associated



(A)



(B)

FIGURE 6.14: Expected total costs *(a)* and reliabilities *(b)* for the WCP-SW instances.

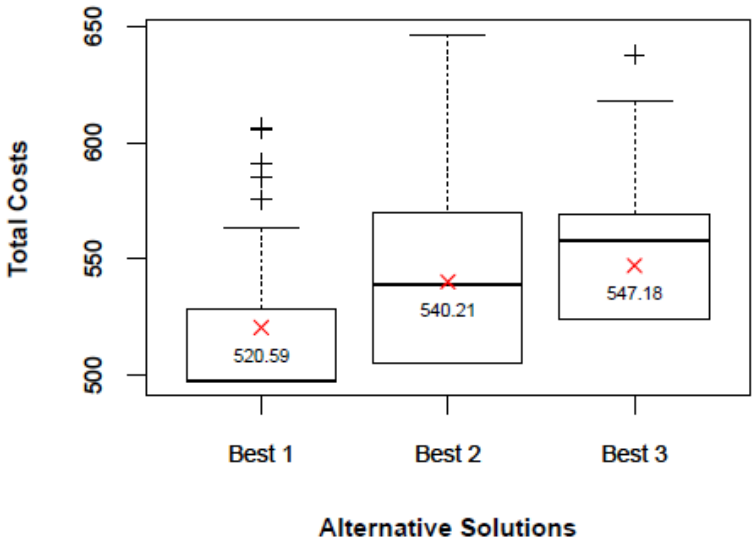


FIGURE 6.15: Boxplots of the total costs of the WCP instance 'Kim277' considering a high waste variance level and a 2% safety capacity level.

a demand D_i, which is a random variable following a given probability distribution. The actual demand of a specific customer is only known when a vehicle visits her/him. The set $A = \{(i, j) : i, j \in N, i \neq j\}$ contains the arcs connecting each pair of nodes. Moreover, there is a set $F = \{1, ..., m\}$ referring to the types of vehicle. For each type $o \in F$, there are p_o available vehicles, the parameter Q_o represents the maximum load that a vehicle can carry, and U_o ($U_o \subseteq N \setminus 0$) denotes the set of customers that can be served. Each arc has associated a cost c^o_{ij} that depends on the type of vehicle. The cost of a route is the sum of the costs of its arcs and a fixed cost for using a vehicle (f_o). The goal is to design routes that satisfy all demands and minimize the total costs.

6.7.1 Methodology

The methodology proposed is a simheuristic procedure combining the ILS metaheuristic and MCS techniques. For building solutions, the successive approximations method (SAM) (Juan et al., 2014c) (Algorithm **??**) is used. The description of the methodology is explained below and summarized in Figure **??**.

Algorithm 12 The SAM procedure

```
1:  procedure BUILDSOLUTION(customers, vehicles)
2:      globalSol ← empty
3:      nonServedCust ← customers
4:      while nonServedCust ≠ empty do
5:          vehType ← selectType(vehicles)
6:          compatCust ← getCompatibleCust(nonServedCust, vehType)
7:          sol ← solveHoSAVRP(compatCust, vehType)
8:          routes ← getRoutes(sol)
9:          numVehOfTypeK ← numberOfVehicle(vehType)
10:         if numberOfRoutes > numVehOfTypeK
11:             routes ← SelectRoutes(numVehOfTypeK, Random)
12:         end if
13:         globalSol ← addRouteToSol(routes, globalSol)
14:         vehicles ← deleteUsedVehicles(vehicles)
15:         nonServedCust ← extractCustomers(nonServedCust, globalSol)
16:     end while
17:     return globalSol
18: end procedure
```

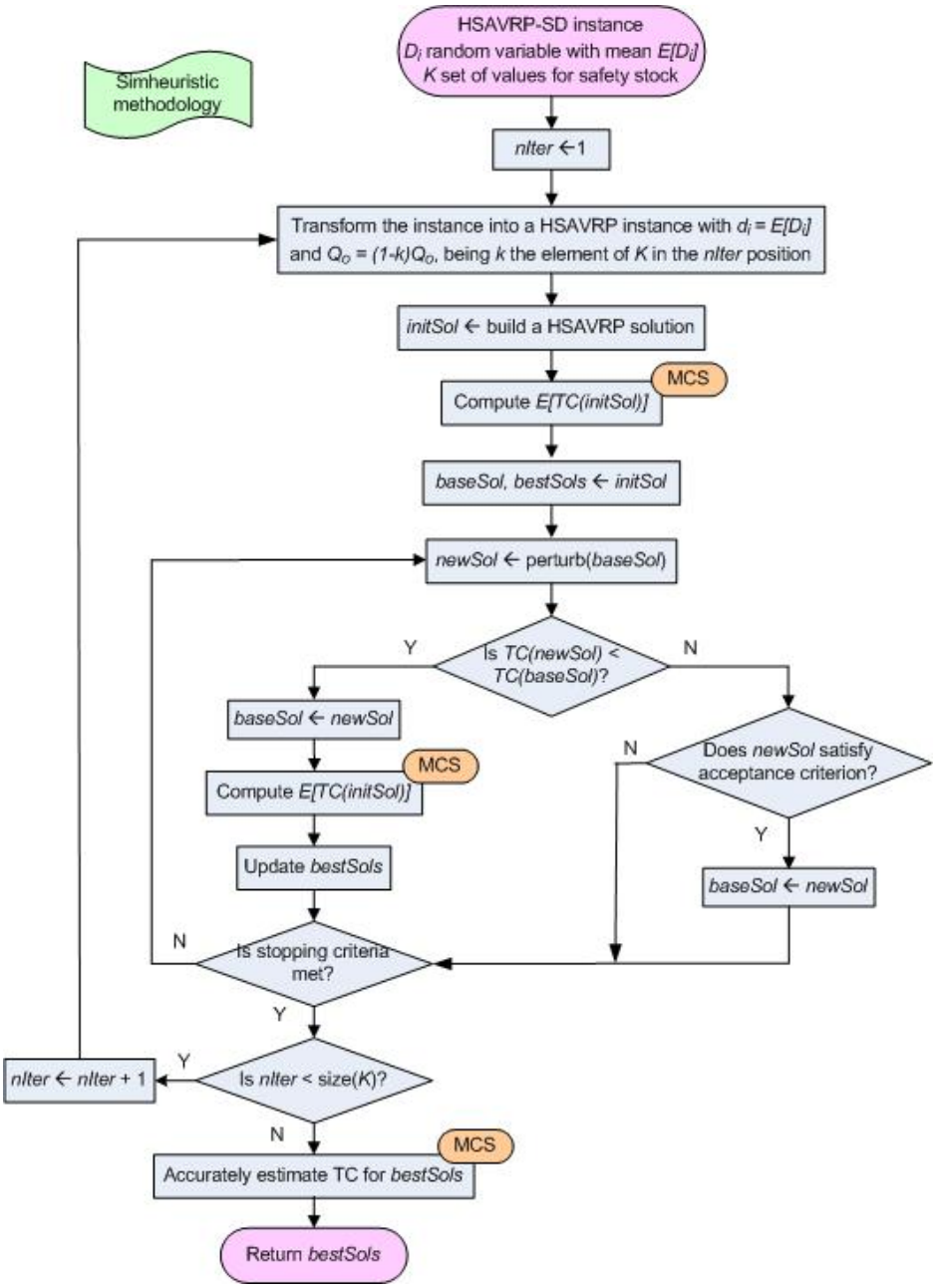


FIGURE 6.16: Flowchart of the proposed approach for the HSAVRP-SD.

The inputs are the HSAVRP-SD instance and a set K of values used to determine safety stocks. Their use leads to lower costs due to route failures (which are the costs of going from the customer being served to the depot to refill and come back to complete the delivery). However, it may also increase the number of routes needed, increasing the deterministic costs (those obtained considering that demand variances are 0). Consequently, it is required to test different values and compare expected total costs.

The algorithm starts by selecting the first value $k \in K$ and transforming the original instance into a deterministic one replacing stochastic demands by their means. Additionally, the capacities are reset to: $Q_o = (1-k)Q_o \ \ (\forall o \in F)$. The next step consists in building an initial solution (*initSol*) for the new instance and estimating the associated total costs using MCS techniques with a short number of scenarios. Afterwards, a base solution (*baseSol*) is constructed by cloning *initSol*, and a list of solutions (*bestSols*) is created, which will store the best stochastic solutions (i.e., those with the lowest expected total cost). Initially, the list includes (*initSol*). Then, a new solution (*newSol*) is obtained by perturbing *baseSol*, which involves removing a random number of routes and repairing

it. If the former has lower total costs (i.e., costs in the deterministic environment), it replaces *baseSol*, the total costs in the stochastic environment are estimated with a short MCS, and *bestSols* is updated. On the other hand, if (*newSol*) is not better than (*baseSol*), an acceptance criterion is checked to decide whether the base solution is replaced. A Demon-like acceptance criterion (Talbi, 2009) is used to allow the base solution to be deteriorated if no consecutive deteriorations take place and the degradation does not exceed the value of the last improvement. By doing this, the algorithm avoids getting stuck in a local optima. This procedure is repeated to visit different solutions until a stopping criteria is met. At this point, the algorithm is re-initialized with another value of K. When all values have been tested, the total costs of *bestSols* are accurately estimated using MCS with a larger number of scenarios. Finally, the list is returned.

Regarding the building of solutions, the SAM procedure is implemented. It can be described as follows. The procedure receives one list of customers and one of available vehicles. First, an empty global solution is created, and the list of customers is copied into a list of non-served customers. While this list is not empty, the next steps are taken. A vehicle type not used yet is selected and those customers not compatible with the selected vehicle are removed from the list. Then the problem is transformed into an homogeneous SAVRP (HoSAVRP) with no limitation on the number of vehicles that is solved with a state-of-the-art algorithm.

If the solution provided reports more routes than the number of available vehicles of the current type, some routes are discarded. This partial solution is included in the global solution. The last instructions inside the while loop update the list of available vehicles and the list of non-served customers. This process ends when all customers are assigned to a route. Finally, the global solution is returned.

The procedure for repairing solutions is exactly the same but receiving as inputs only those customers that remain to be included in a route and copying the perturbed solution into the global one when this is created.

Each HoSAVRP solution is constructed using the SR-GCWS-CS algorithm (Juan et al., 2011a). It is based on the CWS heuristic and incorporates biased randomization techniques and cache and splitting techniques, which contribute to reduce computational times. We have adapted this algorithm in order to consider asymmetric costs. For this, the easy procedure of computing savings as the mean of the two savings associated to each pair of nodes (Gruler et al., 2015) has been applied.

6.7.2 Computational experiments

A set of 4 classical CVRP instances from *Branch and Cut* are generalized to test the approach. The same location of the nodes and demand is used. They have been modified to include the characteristics of the RVRP.

A fixed cost for using a vehicle, f_o, and a variable cost, v_o, that multiplies the distance have been established. Therefore, the cost of arc $(i, j) \in A$, $c^o_{ij} = v_o d_{ij}$, where d_{ij} is the Euclidean distance. In order to account for asymmetric costs, the cost of an edge (i, j) is incremented by 10% if the y-coordinate of the destination node j is greater than the y-coordinate of the origin node j.

An heterogeneous fleet with three type of vehicles has been proposed. Large vehicles have a capacity equal to the one used in the benchmark, and medium and small vehicles have a reduced capacity of 75% and 50% respectively. All vehicles can serve all customers except for customers belonging to a randomly selected sub-area in which it is assumed that large vehicles cannot access.

The demand of a particular customer, D_i, follows a log-normal distribution, with expected value as the demand of the benchmark instance (d_i) and variance proportional to the expected value (κd_i). The results presented next are obtained with $\kappa = 0.1$.

Several measures are computed for each solution. When a solution is evaluated with deterministic demands, the cost (Z^{det}) and the distance (*dist*) are shown. When a solution is assessed with stochastic demands, route failures may happen. Therefore, the expected cost (Z^{stoch}) and the percentage of expected route failures (r^{fail}) is displayed. Finally, the safety stock is also included.

Test cases were run on a laptop with 4 cores at 2.6GHz. Experiments were run over 5 random seeds for 60 seconds except for instance A-n80-k10 which run for 300. The name of the instances indicate the number of nodes (after the letter n). Short MCS were run for 100 scenarios, and long simulations for 10000.

6.7.3 Analysis of results

Table **??** compares the solution of the original CVRP instance with the HSAVRP with deterministic demands. When the SAM method is employed to solve the CVRP, our best solution (OBS) shows to be competitive compared with the optimal (OPT) solution reported in the literature, with an average gap of 0.52%. The composition of the fleet for each solution is also reported. It can be observed that a mix fleet is used, motivated by the fact that some vehicles cannot access some customers. The performance of the deterministic solution is tested in the operational level with stochastic demands in Table **??**.

A particular solution is tested under stochastic demands using MCS techniques. According to Table **??**, the expected cost of the deterministic solution increases on average a 4% and experiences a high percentage of route failures. This is due to the fact that some routes has a filling rate of 99%. On the other hand, stochastic solutions show a filling rate more balanced among the routes, and the route failures decrease dramatically.

6.8 Conclusions

The flow of goods and products is becoming increasingly complex as a consequence of many factors such as the globalization. The weight of this sector in the gross domestic product and the employment rates of most countries require the development of inteligent algorithms to obtain efficient solutions. The constant evolution and dynamism of the sector calls for fast algorithms. Moreover, the rellevance of the social and enviromental impacts caused by this sector and the growing concern for these issues makes it necessary to study classical problems focusing on a different perspective (i.e., not analyzing only the common measures: distances or time).

While the literature on logistic transportation is extensive and varied, there are plenty of research lines to be explored. Here, both classical and novel problems have been addressed, presenting reviews, methodologies, computational experiments, and analyses of results. The main conclusions are:

- Statistical learning techniques may help to deal with uncertainty. Hybrid algorithms for routing problems allow to maximize benefits by increasing sales and total income while accounting for the distribution costs, which is a more realistic approach than the classical.

- Simheuristics are very useful to address routing problems such as the MDVRP and the WCP modeling demands as stochastic variables. Whereas solutions for the deterministic version of the problem (e.g., considering expected values to replace the random variables) tend to provide good results in scenarios characterized by a low variability, this is not true for scenarios with a higher degree of stochasticity.

- Sustainability indicators are needed to analyze the externalities of transport activities. Even if there is a high correlation between a solution's performance in terms of distance or time, and in terms of the cost associated to other sustainability indicators, it is not perfect. As a consequence, the solutions minimizing each indicator individually may be very different in some cases.

- Smart cities require efficient and clean systems of waste collection. There are plenty of works on this problem, most of them using real data. However, there are many lines of research; a version dealing with stochastic waste levels has been addressed.

- RVRPs encompass a large number of challenging problems with real-life applications. The HSAVRP has been tackled with a simple approach based on classical procedures but able to deal with the characteristics of the problem: heterogeneous fleet, site-dependency and asymmetric costs.

TABLE 6.17: Comparison between CVRP distance-based solutions and the HSAVRP cost-based solutions.

	Original instance (CVRP)				Deterministic HSAVRP - OBS									
Instance	Veh. capac.	OPT	OBS	Gap	Veh. used	Avail. vehicles			Z^{det}	dist	Δ dist	Used vehicles		
					L	L	M	S				L	M	S
P-n40-k5	140	458	461.7	*0.81*	5	4	2	3	2318.3	559.5	*21.18*	3	2	1
B-n41-k6	100	829	833.7	*0.56*	6	5	6	6	3656.7	1075.2	*28.97*	4	4	0
B-n45-k5	100	751	754.0	*0.39*	5	4	3	3	2907.6	802.2	*6.40*	4	2	0
A-n80-k10	100	1763	1768.7	*0.32*	10	8	6	6	6809.2	2040.9	*15.39*	6	5	0

TABLE 6.18: Table of results for the HSAVRP-SD instances.

	Deterministic HSAVRP				HSAVRP-SD					
Instance	Z^{det}	dist	Z^{sto} (1)	r^{fail}	Z^{det}	dist	Z^{sto} (2)	r^{fail}	Safety stock	Gap $_{((2)-(1))/(2)}$
P-n40-k5	2318.3	559.5	2320.7	3%	2318.3	560.5	2318.4	0%	100%	0.10%
B-n41-k6	3656.7	1075.2	4054.9	68%	3667.1	1078.6	3678.6	21%	99%	10.23%
B-n45-k5	2907.6	802.2	2972.9	56%	2912.4	804.1	2912.6	0%	99%	2.07%
A-n80-k10	6809.2	2040.9	7334.5	82%	6964.2	2063.0	7004.7	14%	98%	4.71%

Chapter 7

Application in production

This chapter studies PFSPs with stochastic processing times and a common due date. It proposes a simheuristic algorithm based on the ILS metaheuristic and MCS. It is based on the following journal article: Hatami et al. (submitted).
This work has been presented at the following conferences: Calvet et al. (2016c) and Calvet et al. (2016e).

7.1 Introduction

The manufacturing industry is facing important challenges, including fierce competition, short product life cycles, increasing speed of product innovation, high product variety and quality, and rising customer expectations, among others. Industries need to find proper strategies to cope with these challenges and remain successful in the market, being one of these strategies the use of distributed manufacturing systems (Moon et al., 2002), with contrasted benefits in terms of higher product quality, lower production costs and fewer management risks (Wang, 1997; Chan et al., 2005; Kahn et al., 2013).

In distributed manufacturing systems there is an horizontal cooperation among entities when they have strategic relationships and join their individual strengths to achieve a common goal. By doing so, the complexity of manufacture is shared among different entities, resulting in conditions in which risks and costs become acceptable and market opportunities can be captured. Quite often single manufacturing centers are not able to produce products and increase product diversity within reasonable costs because of rigid organizational structures, deterministic approaches to take decisions, lack of technology and a competencies' hierarchical allocation (Sluga et al., 1998; Wang et al., 2006). As a result, single manufacturing centers are infrequent while distributed manufacturing systems are quite usual (Moon et al., 2002; Naderi and Ruiz, 2010). Constructing these collaborative manufacturing systems help industries to address market global challenges in an efficient way but their optimization is more complicated. The optimization of these systems has received a considerable attention from practitioners and the research community in recent years.

The distributed permutation flowshop scheduling problem (DPFSP) (Naderi and Ruiz, 2010) consists of a set of distributed manufacturing factories with flowshop configurations. The responsibility of the factories is to produce a product composed of various jobs. Each factory has to process a certain number of jobs, and all of them should be completed at a given deadline or before. Typically, the DPFSP involves two decisions: assigning each job to be manufactured to a factory, and determining a job sequence for each factory. The classical DPFSP assumes deterministic processing times to simplify the problem. However, real-world manufacturing systems are dynamic and often exposed to uncertainties and unforeseen events such as machine breakdown, changing due date, operator unavailability, materials out of stock, order rush, etc (Rodammer and White, 1988).

This chapter addresses a problem related to the DPFSP, which assumes that the components have already been assigned, and deals with job sequencing for each flowshop. Furthermore, the processing times of the components in each flowshop are random variables. The objective is to find a robust job sequence for each factory which starts to process at the latest possible time while completes all jobs by the deadline. Since stochastic processing times are considered, it will only be guaranteed that jobs are finished by then with a given probability. This probability depends on the probability of each factory ending on time. Thus, if a minimum probability is required, each PFSP can not be separately solved.

The problem can be also related to the PFSP-ST. The literature on this problem is not extensive, especially when compared with the PFSP (Lin et al., 2015; Fernandez-Viagas and Framinan, 2015b; Fernandez-Viagas and Framinan, 2015c; Hsu et al., 2015), but it is becoming more popular (Baker and Altheimer, 2012; Kianfar et al., 2012; Juan et al., 2014a). Since the PFSP is an $\mathcal{NP}$-hard problem when the number of machines are equal to or higher than 3 (Garey et al., 1976), the problem studied is also $\mathcal{NP}$-hard. As a consequence, it is sensible to focus on designing heuristic or metaheuristic approaches for obtaining good solutions in reasonable CPU times.

7.2 Literature review

A review on three problems sharing characteristics with the problem described is presented.

7.2.1 PFSP-ST

Baker and Trietsch (2011) design heuristics for addressing the 2-machine PFSP-ST, where the processing times are independent random variables following specific probability distributions. Later, Baker and Altheimer (2012) present a methodology for the m-machine version. In addition, several variations of the PFSP-ST have been analyzed. For instance, Allaoui et al. (2006) and Choi and Wang (2012) work on the stochastic hybrid FSP, aiming to minimize the expected makespan. The same problem has been tackled by Kianfar et al. (2012) with the goal of minimizing the average tardiness of jobs. A novel approach is applied in Zhou and Cui (2008) for tackling the multi-objective PFSP-ST, where both the flow time and delay time of jobs are minimized.

An interesting line is related to uncertainty. Basically, there are two categories: proactive (or robust) scheduling and reactive scheduling. For works falling in the first category, Roy (2010) propose constructing an original predictive schedule. The aim is to find schedules that do not require new schedules (or significant changes) when confronting disruptions. These works may consider probability distributions or sets of scenarios. Al Kattan and Maragoud (2008), Ghezail et al. (2010) and Liu et al. (2011) address the PFSP with uncertainty implementing proactive scheduling strategies. On the other hand, reactive scheduling consists in revising and re-optimizing schedules when unexpected events take place. A classical option is to obtain a predictive scheduling and then try to repair it according to the actual state of the system. A comprehensive review on rescheduling under disruptions is provided by Katragjini et al. (2013).

Some authors employ exact methods for addressing the PFSP-ST. A disadvantage of many of these methods is that they only work with a specific set of probability distributions and relatively small instances. Moreover, it may be difficult to adapt them for handling dependencies among processing times. Simulation techniques enable researchers to deal with these situations in a natural way. Baker and Altheimer (2012) propose a hybrid approach combining heuristics and simulation, and test three heuristic methods: two relying on the CDS heuristic (Campbell et al., 1970) and one on the NEH heuristic.

7.2.2 DPFSP

In this problem the jobs have not been assigned to each flowshop, so this assignment becomes part of the decision problem. This problem is also known as the distributed flowshop scheduling problem (DFSP) since Naderi and Ruiz (2010) resumed the topic for a distributed environment and makespan minimization. Nevertheless, this decision scheduling problem was first studied by David et al. (1996) based on a glass industry considering non-delay flowshops and batch production mode. Note that each factory is treated as line in this paper, but the mathematical scheduling problem inside is the same. Since then, it has been also studied under different names in the literature: parallel flowline (Vairaktarakis and Elhafsi, 2000) and parallel flowshops (Cao and Chen, 2003). Before Naderi and Ruiz (2010), the particular two-machine-flowshop layout in each factory or line has been solved using approximate algorithms by Zhang and Van De Velde (2012) and Al-Salem (2004). This particular problem turns to be a pure assignment problem due to the Johnson's rule (Johnson, 1954). For a general configuration of m machines, Naderi and Ruiz (2010) propose and compare several mixed integer linear programming models and constructive heuristics to solve the problem. Regarding iterated optimization algorithm, the problem has received an increasing attention for makespan minimization in the last years. Gao and Chen (2011) propose a GA using

local search phases based on interchange and insertion of jobs. A TS algorithm is proposed by Gao et al. (2013). An iterated greedy (IG) algorithm without local search phases is presented by Lin et al. (2013). A SS algorithm with a reference set made up of solutions and restarts mechanisms is proposed by Naderi and Ruiz (2014). Fernandez-Viagas and Framinan (2015a) present an IG algorithm with bounded local search phases employing properties of the problem to reduce the space of solutions. Recently, Ribas et al. (2017) propose several constructive heuristics and two simple iterated algorithms (IG and ILS) with variable neighbourhood searches but with zero-buffer flowshops (blocking constraint).

A particular case of the DFSP refers to the so-called distributed assembly flowshop scheduling problem, which combines the DFSP with assembly scheduling. In this problem, a distributed flowshop composed of f identical flowshops is followed by a single assembly operation. n jobs consisting each one of f components have to be assembled after each component has been manufactured in one of the flowshops. This decision problem includes job assignment plus the scheduling of jobs in the assembly line. The main references for this problem are Hatami et al. (2015) and Hatami et al. (2013). In the first reference, the authors consider the objective of makespan minimization, while in the second sequence-dependent setup times are assumed.

7.2.3 Assembly scheduling

This problem is also denoted n-stage assembly or assembly flowshop scheduling. There are m tandem lines that are arranged prior to a single assembly station. Using this layout, n different products (jobs) have to be manufactured, each one consisting of m components manufactured in the tandem lines. The processing time of each component in each line is different. Some authors distinguish among the *fixed* case (i.e. each component can be processed only in a given tandem line), and the *unfixed* case.

Different objectives are sought, such as makespan minimization (Sung and Juhn, 2009), total flowtime (Al-Anzi and Allahverdi, 2013; Sung and Kim, 2008), due date fulfilment (Al-Anzi and Allahverdi, 2007), or the combination of several indicators (Seidgar et al., 2014). Most references refer to the 2-stage case (production followed by assembly), so they assume that each tandem line consists of a single machine. The underlying hypothesis is that there is a single processing time for each component before the assembly process. Different exact and approximate methods have been proposed, and some variants of the original problem have been tackled by Sung and Juhn (2009), where two types of components –manufactured and imported– are considered, and by Liao et al. (2015), where assembly batches are assumed. Several other variants for three stages have been addressed (see e.g. Koulamas and Kyparisis, 2001 and Komaki et al., 2017), but in none of the different versions the processing times have been assumed to be stochastic.

7.3 The DPFSP-ST

There is a set F of f distributed manufacturing factories. The shop configuration of each factory is a PFSP, which is a particular case of the flowshop scheduling problem (FSP) (Johnson, 1954). In the FSP, there is a set M of m machines where each job of a set N of n jobs must be processed on each machine. Each job starts to process from the first machine to the last one. Therefore, the number of operations per job is equal to the number of machines. The j^{th} operation of job i is processed on machine j, and can start if the $j-1^{th}$ operation on machine $j-1$ has been completed and machine j is free. Processing times are supposed to be known in advance and deterministic. Other classical assumptions (Baker, 1974) are: *(i)* all operations and jobs are independent and available for processing at time 0; *(ii)* all machines are continuously available and there are no breakdowns; *(iii)* each machine can process at most one job at a time; *(iv)* each job can be processed in only one machine at a time; *(v)* once an operation of a given job on a given machine has started, it cannot be interrupted (i.e., no preemption is allowed until the processing has been completed); *(vi)* setup and removal times are sequence-independent and are included in the processing times or are negligible; and *(vii)* in-process storage is considered infinite. In the FSP, there are $(n!)^m$ possible solutions since the number of job permutations per machine is $n!$. The PFSP is a simpler version which assumes that all machines have the same job permutation and the job passing is not allowed. It has $n!$ possible solutions.

In this manufacturing layout, a product consisting of various components (jobs) has to be processed on the machines located at the factories. The processing time of each job i in each machine j, P_{ij}, is considered a random variable. The product is considered finished when all its jobs have been completed. It is required that all components are completed by a (deterministic) deadline $\tilde{d}$ with a probability not lesser than p.

Consequently, it is intended that the processing operations for job i at factory k should terminate by the deadline $\tilde{d}$. In a PFSP with a deadline, a specific job sequence has a makespan associated and the starting time can be set at the deadline minus the makespan. In contrast, the PFSP-ST is characterized by having potential different makespans under different conditions for a given job sequence. Therefore, in this setting, at least one of the three following approaches should be considered:

- To ignore the stochastic nature of the problem and replace the random variables by their representation (typically their mean). While it may provide solutions of poor quality, this is not necessarily the case (Framinan and Perez-Gonzalez, 2015). The reason is that a deterministic optimization algorithm is faster and, as a consequence, may visit more solutions during a limited amount of time. Thus, if the level of stochasticity is low, there is a chance that solutions found are robust enough to have a good performance in a stochastic environment. This approach is labelled as *makespan* (M) in the following.

- To minimize the expected makespan. This approach stresses the average behaviour of the layout. However, if the starting time is set at the deadline minus the expected makespan, there is no guarantee that all processing operations will be completed on time. This approach is labelled as *expected makespan* (EM).

- To ensure that the final product will be finished on time with a probability p. This option allows the decision-maker to include a restriction that sets the probability of finishing on time or, conversely, the risk of a delay. This approach is labelled as *percentile makespan* (PM).

  It is assumed that the factories are independent, so p can be computed as: $p = \prod_{k=1}^{f} p_k$, and by assuming an equal allocation of probabilities we have $p_k = \sqrt[f]{p}$. Therefore, the problem is equivalent to ensure that factory k will finish its jobs with a probability p_k. In order to do so, the p_k-th makespan percentile can be computed for each factory k given a sample of makespans, and its starting time can be set to the deadline minus the makespan percentile.

Figure **??** shows the concepts of starting time, expected makespan and makespan percentile. The choice between the last two approaches depends on the risk-aversion of the decision-maker. For example, if the decision-maker prefers to focus on the worst outputs (i.e., the largest makespans), it is better to minimize the makespan percentile requiring a high probability. On the other hand, if he prefers to analyze the average case, he should focus on minimizing the expected makespan.

7.3.1 Methodology

Three algorithms are presented: the ILS_M algorithm considers the deterministic version of the problem, while the SIM-ILS_{EM} and the SIM-ILS_{MP} algorithms minimize the expected makespan and the percentile makespan, respectively. For each solution returned by an algorithm, the (deterministic) makespan, the expected makespan and the makespan percentile are computed. The aim of working with different algorithms is to study and compare their behaviour. While simulation techniques are used in the SIM-ILS_{EM} and the SIM-ILS_{MP} algorithms, the ILS_M algorithm, which works with average processing times, skips that part. From here, SIM-ILS algorithm refers to the basic structure of the SIM-ILS_{EM} and the SIM-ILS_{MP} algorithms.

The SIM-ILS algorithm combines the ILS metaheuristic with MCS. The metaheuristic searches for promising solutions while MCS techniques are employed to assess their performance. The promising solutions are returned by the metaheuristic when solving a (deterministic) PFSP instance, which is created from the original PFSP-ST instance by replacing the random variables P_{ij} by constant values p_{ij} using the means, i.e., $p_{ij} = E[P_{ij}]$. Simulation is applied to a given solution to compute the expected makespan or makespan percentile. The algorithm works with a list of best stochastic solutions found and the best deterministic solution found. The best deterministic one

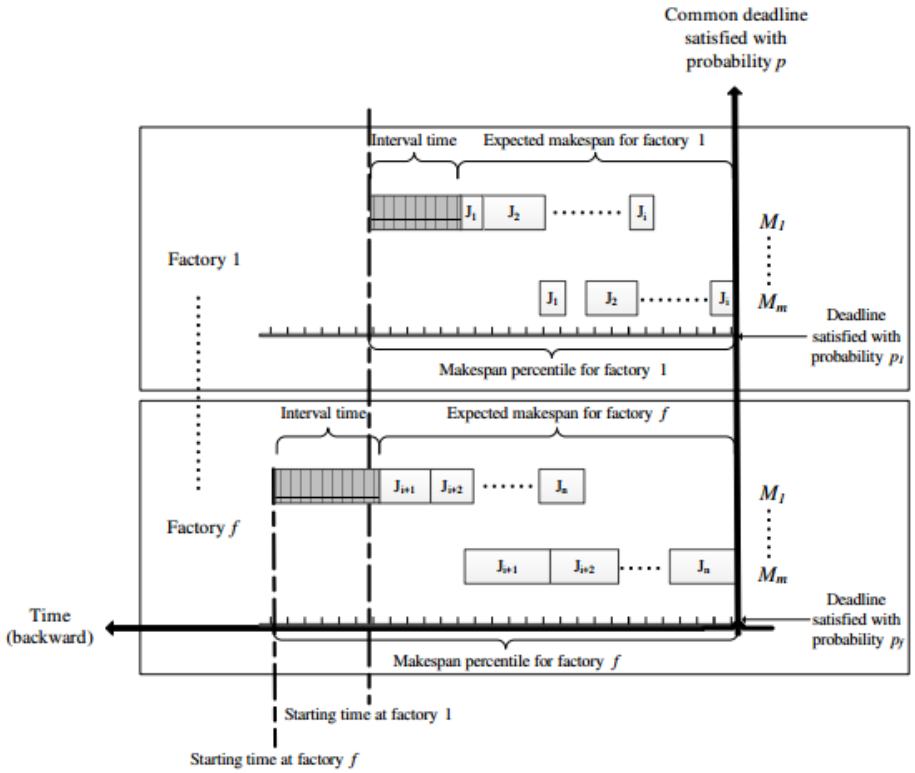


FIGURE 7.1: Starting time, expected makespan and makespan percentile in the DPFSP-ST.

is the job sequence with the smallest makespan referring to the PFSP instance. Depending on the objective considered, the best stochastic solutions found are the job sequences with the smallest expected makespans or makespan percentiles, referring to the PFSP-ST instance. The algorithm starts solving the PFSP. The obtained result is set as the best deterministic solution and the best stochastic solution. During the algorithm execution, the best stochastic solutions are saved in a list with length l. This list is sorted iteratively in increasing order of the considered objective function. Thus, the solution at the first position is considered as the best stochastic solution. The steps of the algorithm are detailed in Figure **??** and explained below.

Generation of the initial solution

A biased-randomized version of the classical NEH heuristic (Nawaz et al., 1983) described in Juan et al. (2014a) is proposed to generate initial solutions.

Solution improvement

An iterative improvement procedure using *shift-to-left* as first-improvement type pivoting rule (Ruiz and Stützle, 2007; Juan et al., 2014a) is applied in different parts of the algorithm to improve solutions. Each iteration of the procedure consists of three steps. In the first, a position s is randomly selected, without repetition, from the current job sequence. The selected positions are saved in a selection list. In the second step, the job placed in the position s is removed from the sequence and the *shift-to-left* movement is applied, i.e., the insertion of the job in each possible position at the left side of s is tested. The makespan of each option is calculated through the accelerations of Taillard (Taillard, 1990). Finally, the job is inserted in the position resulting in the sequence with the smallest makespan. The iteration of these steps are continued until all positions have been selected or a better solution is achieved. If there is an improvement, the algorithm is restarted with an empty selection list.

Simulation

The assessment of a solution using MCS techniques follows these steps: *(1)* a number of iterations *numsim* is considered to repeat the simulation process; *(2)* a job processing time is generated for each random variable according to the probability distribution associated, and the makespan is computed; *(3)* this process is repeated *numsim* times; and *(4)* a performance measure such as the expected makespan, $E[C_{max}]$, or the *pro* makespan percentile, $P[C_{max}]^{pro}$, is computed.

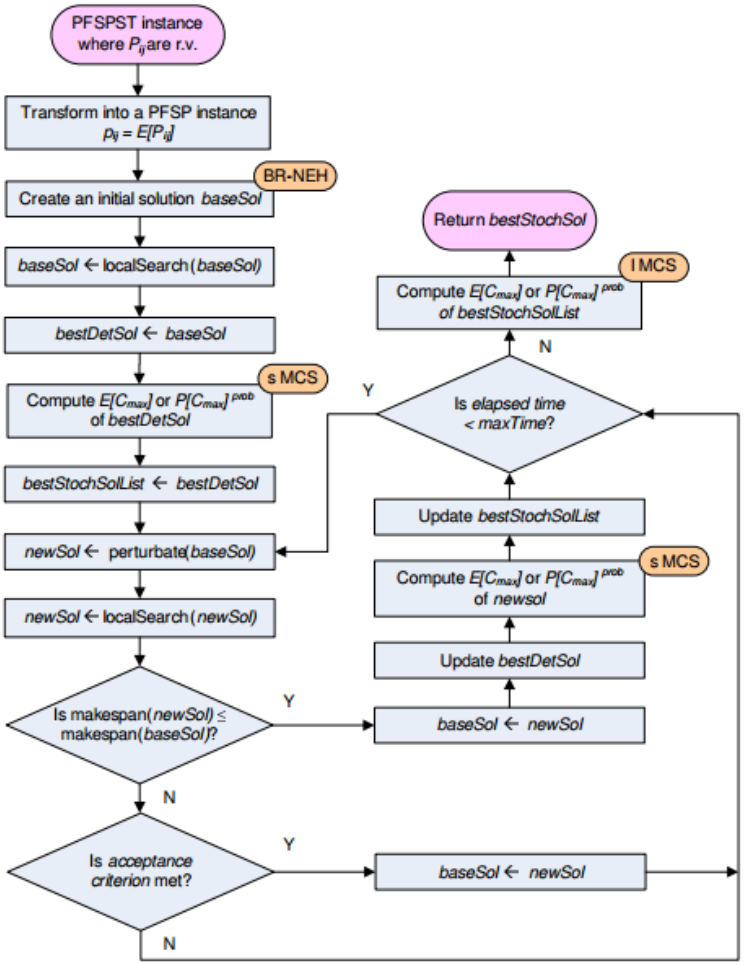


FIGURE 7.2: Flowchart of the proposed approach for the DPFSP-ST.

While the assessment of solutions during the search is done quickly (i.e., *numsim* is relatively small), a long simulation (*numsim* relatively big) is used at the end to provide accurate estimates related to the best deterministic and best stochastic solutions.

Iterated local search

A series of steps are performed iteratively during the search. Initially, a perturbation operator is applied to change the region of the current solution space and then, the new solution is improved using the local search. The simple and efficient *enhanced-swap* operator proposed by Juan et al. (2014e) is used to perturbate the solution. It follows three instructions: *(1)* two different positions are selected randomly from the current job sequence; *(2)* the jobs at these positions are interchanged; and *(3)* the *shift-to-left* movement is applied for both jobs.

In the second step, the algorithm decides whether the new solution is accepted. If it has a smaller makespan than the current base solution, then the latter is replaced by the new. In this case, the best deterministic solution is accordingly updated (i.e., replaced by the new solution if this has a smaller makespan). Additionally, a short simulation is applied to check whether the best stochastic solution list has also to be updated considering the objective function value. Finally, if the new solution does not provide a smaller makespan than the current base solution, an acceptance criterion is applied. These steps are repeated until the stopping criterion based on the elapsed CPU time is reached.

Acceptance criteria

The algorithm assigns an acceptance probability to the new solutions that are worse than the current base solution. This criterion prevents the algorithm from getting stuck in a local optima. It is used for the first time by Hatami et al. (2015). Given a new solution π_n with a worse makespan than the current base solution π_C, the acceptance criterion decides if it is accepted or not. Let $C_{Max}(\pi_c)$ and $C_{Max}(\pi_n)$ denote the makespans of each solution. The acceptance of π_n depends on

the probabilistic mechanism shown in Equation **??**, where *random* is a random number uniformly distributed between 0 and 1, and the relative percentage difference is: $RPD = \frac{C_{(\pi_n)} - C_{(\pi_c)}}{C_{(\pi_c)}} \times 100$.

$$random \leq e^{-RPD} \tag{7.1}$$

7.3.2 Computational experiments

The algorithms have been implemented as Java applications and tested on 27 instances. A standard personal computer, Intel QuadCore i5 CPU at 3.2 GHz and 4 GB RAM with Windows 7, has been used to execute all tests.

Set of instances and test

Since no benchmark instances exist for the problem analyzed, a new set is constructed based on Taillard instances (Taillard, 1993). Table **??** gathers the following characteristics for each instance: name, total number of the jobs (total n), number of machines (m) and number of factories (f). For a given factory, each instance contains a processing time p_{ij} for job i at machine j, which describes a random variable P_{ij} following a log-normal distribution with mean p_{ij} and variance σ_{ij}^2 set to $c \cdot p_{ij}$. In real-life applications, empirical distributions based on historical data could be used.

Table 7.1: Description of the generated instances for the DPFSP-ST.

	Total n								
	20			50			100		
f / m	5	10	20	5	10	20	5	10	20
2	Ins. 1	Ins. 4	Ins. 7	Ins. 10	Ins. 13	Ins. 16	Ins. 19	Ins. 22	Ins. 25
3	Ins. 2	Ins. 5	Ins. 8	Ins. 11	Ins. 14	Ins. 17	Ins. 20	Ins. 23	Ins. 26
4	Ins. 3	Ins. 6	Ins. 9	Ins. 12	Ins. 15	Ins. 18	Ins. 21	Ins. 24	Ins. 27

Three levels of processing time variability c (small, medium and high) are considered: 0.25, 1 and 1.5, respectively. Three different values of 80%, 90% and 95% are considered for the general probability p (used only for the SIM-ILS$_{MP}$ algorithm). The maximum computational time for solving the PFSP-ST of each factory is limited to $0.05 \cdot n \cdot m$, which seems a reasonable amount for real-life applications. Ten seeds are randomly generated and only the best result is stored. Regarding the number of iterations for assessing solutions, 600 and 1000 runs are employed during the algorithm and at the end, respectively. Note that the selection of these values are mainly driven by the computing time available. Thus, if more time is available, then these values can be incremented in order to obtain better and more accurate results.

Results

Results are displayed in Tables **??**-**??**, where each table represents a specific level of processing time variability: low, medium and high. Due to space limitations and the fact that results show similar trends for all three values of general probability, only those related to 90% are shown. The composition of the tables is as follows. The first column identifies the instance. The next five summarize the results of the ILS$_M$ algorithm. For each instance, they show the following information regarding the best solution found: $C_{max}(1)$, $E[C_{max}](2)$, $P[C_{max}]^{pro}(3)$, gap between the first two measures, computed as: $(E[C_{max}](2) - C_{max}(1))/C_{max}(1) \cdot 100$, and gap between the second and the third ones. While the first gap represents the 'extra' processing time, on average, for applying a solution assuming deterministic processing times, the second focuses on percentiles, showing the additional processing time required to finish the product with a probability of 90%. The next four columns provide the following results of the SIM-ILS$_{EM}$ algorithm: $E[C_{max}](4)$, $P[C_{max}]^{pro}(5)$, gap between the expected makespan of the best solutions found by the ILS$_M$ and the SIM-ILS$_{EM}$ algorithms, and the gap of percentiles among the same solutions. The third gap, which is expected to be null or negative, shows the benefit of using a simheuristic approach (i.e., taking into account the variability of the processing times) in terms of expected makespan. The fourth gap quantifies the difference of percentiles. Similarly, the next four columns refer to the best solution found by the SIM-ILS$_{MP}$ algorithm. In particular, they contain: $E[C_{max}](6)$, $P[C_{max}]^{pro}(7)$, and gaps of expected makespans and percentiles between the best solutions found by the SIM-ILS$_{EM}$

and SIM-ILS$_{MP}$ algorithms. These gaps allow us to quantify the processing time difference based on whether one measure or the other is minimized. Finally, the last column shows the mean computational time of the three solutions obtained. In addition, a row is added at the end of each table to gather the mean gaps and computational time.

Boxplots in Figure **??** show the distributions of gaps of $E[C_{max}]$ and $P[C_{max}]^{pro}$ regarding the best values considering the three algorithms and a probability of 90%. While the approach minimizing a given measure is expected to present a null value for the corresponding gap, this figure reveals the difference between choosing one approach or the other. Focusing on the instance 14, Figure **??** represents the 30 solutions found (resulting of 3 algorithms and 10 seeds). Each column is a measure, and colors and line formats are used to distinguish algorithms. As the previous figure, this analysis provides insights about a "potential" trade-off between the measures. Additionally, the figure gives information about the effect of using multiple seeds.

Figure **??** represents the relationship between probability required, variability level of the processing times and $P[C_{max}]^{pro}$ for the instance 14 using the SIM-ILS$_{MP}$. Finally, Figure **??** shows the effects of different instance characteristics on $P[C_{max}]^{pro}$ considering a medium level of variability and a probability of 90%. First, an analysis of variance was carried out to identify which factors and pairwise interactions had a statistically significant effect on the results. For each of these elements (single factors or pair of them), a figure is drawn which shows the mean value associated to each level of the factor or combination of levels for pair of factors. Given the randomness in the generation of instances, all factors are expected to have significant positive effects.

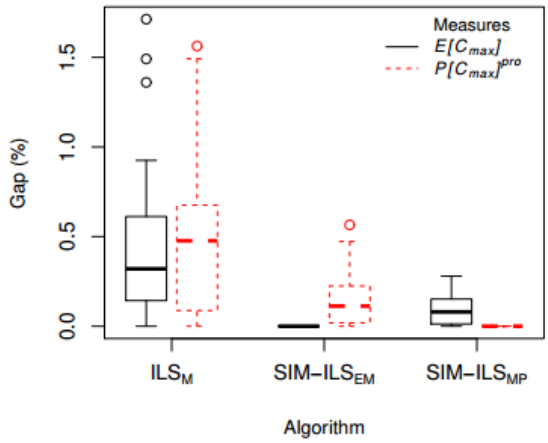


FIGURE 7.3: Boxplots of performance gaps for the DPFSP-ST instances considering a medium level of variability and $p = 90\%$.

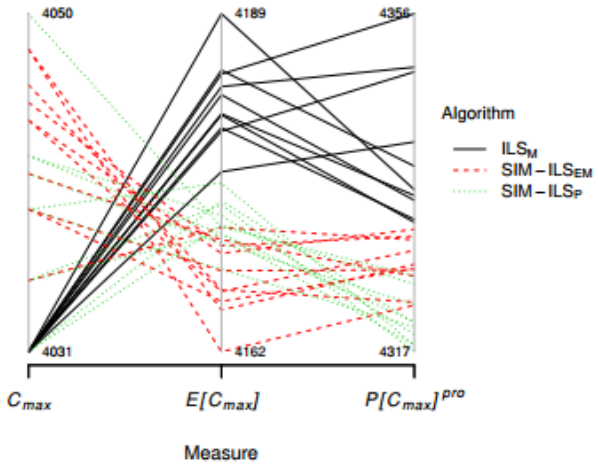


FIGURE 7.4: Parallel coordinates plot showing different measures for the DPFSP-ST instance '14', considering a medium level of variability and 10 seeds.

TABLE 7.2: Results considering low level of variability ($c = 0.25$) and general probability $p = 90\%$.

	ILS$_M$					SIM-ILS$_{EM}$				SIM-ILS$_{MP}$				
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]^{pro}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]^{pro}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]^{pro}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1516.00	1555.54	0.73	2.61	1512.33	1552.96	-0.24	-0.17	1514.19	1551.75	0.12	-0.08	5.00
2	1758	1772.30	1829.89	0.81	3.25	1762.47	1824.72	-0.55	-0.28	1765.96	1822.13	0.20	-0.14	3.99
3	1862	1869.94	1949.62	0.43	4.26	1864.56	1944.33	-0.29	-0.27	1865.41	1942.73	0.05	-0.08	4.50
4	2154	2169.84	2216.99	0.74	2.17	2167.72	2213.02	-0.10	-0.18	2169.41	2212.05	0.08	-0.04	9.99
5	2829	2854.76	2927.28	0.91	2.54	2852.35	2923.38	-0.08	-0.13	2854.23	2921.13	0.07	-0.08	10.00
6	3179	3194.56	3295.33	0.49	3.15	3189.44	3294.94	-0.16	-0.01	3191.61	3288.55	0.07	-0.19	7.99
7	3413	3451.80	3503.35	1.14	1.49	3440.75	3495.76	-0.32	-0.22	3442.72	3493.16	0.06	-0.07	19.99
8	4306	4333.25	4421.54	0.63	2.04	4332.04	4424.23	-0.03	0.06	4334.49	4418.29	0.06	-0.13	20.00
9	5733	5760.93	5904.21	0.49	2.49	5752.36	5890.05	-0.15	-0.24	5756.43	5884.47	0.07	-0.09	14.00
10	2960	2987.24	3040.81	0.92	1.79	2960.68	3020.68	-0.89	-0.66	2964.55	3019.89	0.13	-0.03	12.46
11	3250	3285.55	3352.58	1.09	2.04	3272.02	3349.40	-0.41	-0.09	3273.94	3346.25	0.06	-0.09	12.49
12	3378	3417.10	3505.31	1.16	2.58	3392.12	3501.08	-0.73	-0.12	3395.54	3490.67	0.10	-0.30	12.49
13	3572	3620.75	3669.11	1.36	1.34	3620.75	3669.11	0.00	0.00	3620.75	3669.11	0.00	0.00	25.01
14	4031	4093.01	4172.92	1.54	1.95	4083.66	4163.44	-0.23	-0.23	4084.43	4156.70	0.02	-0.16	25.00
15	4574	4620.28	4728.14	1.01	2.33	4605.65	4720.76	-0.32	-0.16	4609.00	4714.98	0.07	-0.12	24.99
16	5058	5132.22	5194.14	1.47	1.21	5131.72	5191.09	-0.01	-0.06	5131.72	5191.09	0.00	0.00	49.99
17	6039	6113.22	6205.43	1.23	1.51	6109.34	6208.90	-0.06	0.06	6113.31	6204.41	0.06	-0.07	49.97
18	7013	7090.64	7215.40	1.11	1.76	7081.97	7215.77	-0.12	0.01	7085.22	7204.73	0.05	-0.15	49.98
19	5797	5840.48	5913.77	0.75	1.25	5810.07	5891.05	-0.52	-0.38	5818.16	5889.75	0.14	-0.02	24.98
20	5819	5854.95	5967.56	0.62	1.92	5825.96	5939.00	-0.50	-0.48	5832.76	5935.59	0.12	-0.06	24.95
21	6065	6104.18	6236.87	0.65	2.17	6078.89	6220.23	-0.41	-0.27	6083.24	6217.93	0.07	-0.04	24.98
22	6235	6324.25	6392.90	1.43	1.09	6324.25	6392.90	0.00	0.00	6326.47	6388.82	0.04	-0.06	49.98
23	6414	6502.24	6598.61	1.38	1.48	6498.90	6592.36	-0.05	-0.09	6498.90	6592.36	0.00	0.00	50.01
24	7183	7289.80	7422.74	1.49	1.82	7283.76	7418.55	-0.08	-0.06	7283.75	7407.78	0.00	-0.15	50.00
25	7603	7697.50	7763.14	1.24	0.85	7697.50	7763.14	0.00	0.00	7697.50	7763.14	0.00	0.00	100.03
26	8598	8710.46	8823.77	1.31	1.30	8710.46	8823.77	0.00	0.00	8710.46	8823.77	0.00	0.00	100.01
27	9892	10038.66	10178.40	1.48	1.39	10029.92	10175.82	-0.09	-0.03	10029.92	10175.82	0.00	0.00	99.99
Mean				1.02	1.99			-0.24	-0.15			0.06	-0.08	32.69

Table 7.3: Results considering medium level of variability (c = 1) and general probability p = 90%.

	ILS$_M$					SIM-ILS$_{EM}$				SIM-ILS$_{MP}$				
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]^{pro}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]^{pro}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]^{pro}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1541.15	1617.25	2.40	4.94	1533.01	1614.37	-0.53	-0.18	1533.96	1606.78	0.06	-0.47	5.00
2	1758	1799.07	1927.61	2.34	7.14	1782.58	1901.69	-0.92	-1.34	1787.49	1897.95	0.28	-0.20	3.99
3	1862	1880.42	2046.20	0.99	8.82	1876.25	2041.74	-0.22	-0.22	1879.09	2030.26	0.15	-0.56	4.50
4	2154	2208.99	2298.54	2.55	4.05	2203.58	2295.21	-0.25	-0.14	2205.94	2287.00	0.11	-0.36	10.00
5	2829	2898.36	3035.07	2.45	4.72	2896.75	3033.22	-0.06	-0.06	2899.76	3032.76	0.10	-0.02	9.99
6	3179	3223.10	3424.39	1.39	6.25	3215.97	3414.35	-0.22	-0.29	3215.97	3414.35	0.00	0.00	8.00
7	3413	3517.97	3617.33	3.08	2.82	3493.77	3603.77	-0.69	-0.37	3495.26	3594.01	0.04	-0.27	19.99
8	4306	4387.69	4552.74	1.90	3.76	4380.10	4564.70	-0.17	0.26	4382.97	4552.50	0.07	-0.27	19.99
9	5733	5811.38	6094.35	1.37	4.87	5792.84	6067.04	-0.32	-0.45	5797.46	6055.08	0.08	-0.20	14.00
10	2960	3030.64	3136.25	2.39	3.48	2979.60	3094.51	-1.68	-1.33	2983.69	3090.15	0.14	-0.14	12.49
11	3250	3336.39	3502.73	2.66	4.99	3321.23	3475.18	-0.45	-0.79	3326.65	3466.45	0.16	-0.25	12.49
12	3378	3482.60	3673.45	3.10	5.48	3431.44	3638.61	-1.47	-0.95	3435.06	3627.96	0.11	-0.29	12.51
13	3573	3710.11	3810.50	3.84	2.71	3708.59	3810.36	-0.04	0.00	3708.59	3810.36	0.00	0.00	25.00
14	4031	4181.10	4332.06	3.72	3.61	4162.39	4322.68	-0.45	-0.22	4174.02	4317.41	0.28	-0.12	25.01
15	4574	4704.41	4911.53	2.85	4.40	4683.63	4901.01	-0.44	-0.21	4686.14	4896.47	0.05	-0.09	25.00
16	5058	5249.31	5356.33	3.78	2.04	5243.28	5349.75	-0.11	-0.12	5243.28	5349.75	0.00	0.00	50.00
17	6039	6241.34	6419.70	3.35	2.86	6226.07	6409.99	-0.24	-0.15	6229.50	6405.71	0.05	-0.07	49.98
18	7013	7205.94	7440.63	2.75	3.26	7205.49	7448.99	-0.01	0.11	7205.94	7440.63	0.01	-0.11	49.98
19	5797	5897.76	6042.62	1.74	2.46	5855.19	6010.80	-0.72	-0.53	5866.03	6001.97	0.19	-0.15	24.96
20	5819	5954.39	6150.09	2.33	3.29	5874.48	6086.12	-1.34	-1.04	5887.17	6081.12	0.22	-0.08	24.92
21	6065	6181.98	6426.37	1.93	3.95	6128.44	6403.68	-0.87	-0.35	6144.92	6395.87	0.27	-0.12	24.99
22	6233	6449.74	6575.00	3.48	1.94	6441.78	6569.46	-0.12	-0.08	6441.78	6569.46	0.00	0.00	50.00
23	6415	6663.16	6858.39	3.87	2.93	6633.10	6821.64	-0.45	-0.54	6642.01	6820.06	0.13	-0.02	49.99
24	7183	7464.61	7726.46	3.92	3.51	7428.87	7676.50	-0.48	-0.65	7440.15	7674.73	0.15	-0.02	49.98
25	7600	7863.12	7990.74	3.46	1.62	7863.12	7990.74	0.00	0.00	7863.12	7990.74	0.00	0.00	100.00
26	8592	8897.85	9103.21	3.56	2.31	8897.85	9103.21	0.00	0.00	8897.85	9103.21	0.00	0.00	99.98
27	9888	10267.27	10539.56	3.84	2.65	10250.60	10535.72	-0.16	-0.04	10252.56	10529.88	0.02	-0.06	99.97
Mean				2.78	3.88			-0.46	-0.36			0.10	-0.14	32.69

TABLE 7.4: Results considering high level of variability (c = 1.5) and general probability p = 90%.

	ILS$_M$					SIM-ILS$_{EM}$				SIM-ILS$_{MP}$				
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]^{pro}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]^{pro}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]^{pro}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1558.65	1652.71	3.56	6.03	1543.93	1645.57	-0.94	-0.43	1547.66	1634.96	0.24	-0.64	5.00
2	1758	1815.86	1971.38	3.29	8.56	1793.26	1936.85	-1.24	-1.75	1796.58	1932.27	0.19	-0.24	3.99
3	1862	1899.64	2088.00	2.02	9.92	1883.39	2087.17	-0.86	-0.04	1887.72	2070.03	0.23	-0.82	4.50
4	2154	2230.49	2338.89	3.55	4.86	2223.55	2331.53	-0.31	-0.31	2225.38	2325.86	0.08	-0.24	10.00
5	2829	2923.50	3100.37	3.34	6.05	2920.98	3095.04	-0.09	-0.17	2925.05	3089.05	0.14	-0.19	9.99
6	3179	3250.82	3500.13	2.26	7.67	3231.39	3488.72	-0.60	-0.33	3239.13	3474.39	0.24	-0.41	8.00
7	3413	3550.25	3693.79	4.02	4.04	3521.54	3653.49	-0.81	-1.09	3522.88	3647.18	0.04	-0.17	20.00
8	4306	4417.80	4645.59	2.60	5.16	4408.96	4633.95	-0.20	-0.25	4414.71	4618.62	0.13	-0.33	19.99
9	5733	5829.65	6148.97	1.69	5.48	5815.95	6148.41	-0.23	-0.01	5820.54	6141.05	0.08	-0.12	14.00
10	2960	3051.56	3186.60	3.09	4.43	2989.60	3128.79	-2.03	-1.81	2989.60	3128.79	0.00	0.00	12.46
11	3250	3369.09	3547.77	3.66	5.30	3348.04	3530.25	-0.62	-0.49	3354.05	3526.75	0.18	-0.10	12.50
12	3378	3510.02	3741.10	3.91	6.58	3455.00	3711.78	-1.57	-0.78	3459.46	3691.02	0.13	-0.56	12.49
13	3570	3745.06	3862.43	4.90	3.13	3745.06	3862.43	0.00	0.00	3745.06	3862.43	0.00	0.00	24.99
14	4031	4221.27	4401.91	4.72	4.28	4206.78	4396.52	-0.34	-0.12	4210.35	4395.90	0.08	-0.01	24.99
15	4574	4743.71	4997.22	3.71	5.34	4729.26	5001.05	-0.30	0.08	4737.43	4987.14	0.17	-0.28	24.99
16	5058	5312.82	5448.19	5.04	2.55	5304.92	5443.07	-0.15	-0.09	5304.92	5443.07	0.00	0.00	50.01
17	6039	6301.25	6533.04	4.34	3.68	6289.15	6517.20	-0.19	-0.24	6293.81	6507.38	0.07	-0.15	49.97
18	7013	7282.57	7597.07	3.84	4.32	7273.14	7594.89	-0.13	-0.03	7277.18	7565.94	0.06	-0.38	49.99
19	5797	5967.38	6140.62	2.94	2.90	5881.63	6066.46	-1.44	-1.21	5889.21	6054.73	0.13	-0.19	24.99
20	5819	5990.94	6236.24	2.95	4.09	5908.93	6161.28	-1.37	-1.20	5916.77	6155.52	0.13	-0.09	24.94
21	6065	6230.30	6563.94	2.73	5.36	6169.34	6505.67	-0.98	-0.89	6172.18	6485.97	0.05	-0.30	24.98
22	6237	6531.98	6675.40	4.73	2.20	6509.74	6672.00	-0.34	-0.05	6510.28	6662.21	0.01	-0.15	49.99
23	6411	6718.36	6953.37	4.79	3.50	6707.55	6922.70	-0.16	-0.44	6707.55	6922.70	0.00	0.00	50.00
24	7183	7519.80	7820.43	4.69	4.00	7499.62	7802.18	-0.27	-0.23	7499.62	7802.18	0.00	0.00	50.00
25	7603	7970.91	8150.20	4.84	2.25	7962.01	8126.37	-0.11	-0.29	7962.01	8126.37	0.00	0.00	100.00
26	8600	9011.27	9264.51	4.78	2.81	9011.27	9264.51	0.00	0.00	9011.27	9264.51	0.00	0.00	99.98
27	9892	10386.80	10731.39	5.00	3.32	10354.76	10707.00	-0.31	-0.23	10354.76	10707.00	0.00	0.00	99.99
Mean				3.74	4.73			-0.58	-0.46			0.09	-0.20	32.69

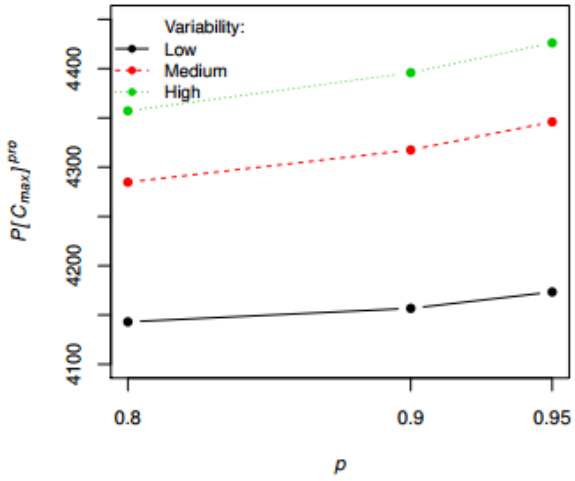


FIGURE 7.5: $P[SC_{max}]^{pro}$ as function of general probability and variability level for the DPFSP-ST instance '14' considering the SIM-ILS$_{MP}$ algorithm.

7.3.3 Analysis of results

Tables **??**-**??** provide detailed information on the performance of the algorithms. The following comments refer to the results of the ILS$_M$ algorithm. Mean gaps between C_{max} and $E[C_{max}]$ for small, medium and high levels of variability are 1.02%, 2.78%, and 3.74%, respectively. These values between $E[C_{max}]$ and $P[C_{max}]^{pro}$ are 1.99%, 3.88%, and 4.73%. They quantify the extra processing time required, on average, when variability is not considered, and the processing time needed to satisfy the deadline with a probability of 90%. For example, in the scenario of low variability, the processing time will be on average 1.02% higher than that assumed, and the processing time needed to finish with the specific probability will be 1.99% higher than the $E[C_{max}]$. Both gaps increase as the variability is incremented. Comparing the results of the ILS$_M$ and the SIM-ILS$_{EM}$ algorithms, the mean gaps of $E[C_{max}]$ (−0.24%, −0.46% and −0.58%) and $P[SC_{max}]^{pro}$ (−0.15%, −0.36% and −0.46%) quantify the benefits of using a simheuristic algorithm. Regarding the results of the SIM-ILS$_{EM}$ and SIM-ILS$_{MP}$ algorithms, the mean gaps of $E[C_{max}]$ (0.06%, 0.10% and 0.09%) and $P[C_{max}]^{pro}$ (−0.08%, −0.14% and −0.20%) at different level of variability, evidence the benefits of using one or the other approach. Thus, the main findings are: *(i)* ignoring the variability in processing times may have an important effect on the performance measures (even in scenarios with a low level of variability); *(ii)* the solutions found by the SIM-ILS$_{EM}$ and the SIM-ILS$_{MP}$ algorithms are relatively similar in terms of these measures but not equal; and *(iii)* the gaps tend to increase with the variability, i.e., minimizing the expected makespan is almost equivalent to consider the makespan percentile when the variability is low, but the difference increases as the variability is incremented. As a consequence, a decision-maker has to assess whether he prefers to minimize the expected makespan (i.e., processing finished at the deadline, on average) or the percentile (i.e., be sure that the processing will be finished at the deadline or before with a given probability), which may be seen as a more conservative or risk-aversion approach.

Figure **??** compares performance measures among the algorithms proposed. The distributions of the gaps are relatively symmetric, with few outliers on right tails. It is easy to see that the biggest gaps are related to the ILS$_M$ algorithm, while the gaps referring to $P[SC_{max}]^{pro}$ values are higher. Similarly, Figure **??** shows that there is a stronger correlation between the simheuristic-based algorithms in the sense that the profiles are similar. It is interesting to analyze the differences among the solutions obtained with multiple seeds. For instance, while solutions of the ILS$_M$ algorithm have the same or a similar C_{max}, these solutions may differ significantly in the other measures (i.e., there are solutions more robust than others). For the instance studied, the ranges of the last two measures are higher than that of the first.

Figure **??** represents a valuable tool for a decision-maker. It analyzes the relationship between the probability required to process a product at the deadline or before and the processing time needed. As the probability tends to 1 (i.e., no risk) the processing time tends to infinite. Instead of having a single solution, the decision-maker may choose the best option (given risk-aversion,

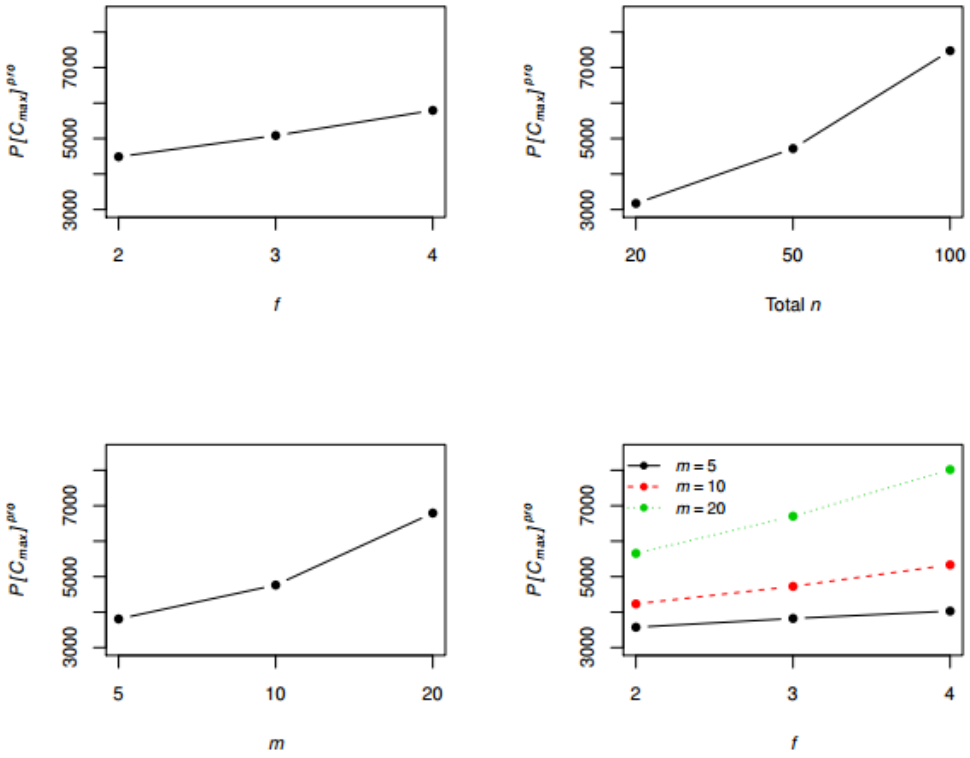


FIGURE 7.6: Effect of different DPFSP-ST instance characteristics on $P[C_{max}]^{pro}$ considering the SIM-ILS$_{MP}$ algorithm.

company policies/situation, etc.) among many. As expected, for a given p value, $P[C_{max}]^{pro}$ increases as the variability is incremented.

Figure **??** reveals that factors total n, m, f, and the interaction between f and m have statistically significant and positive effects (when considering the others elements) on $P[C_{max}]^{pro}$. The ranges related to total n and m are the highest. While the effects of f and total n seem lineal, the effect of m draws a convex function. Focusing on the interaction, it can be concluded that the effect of f is positive for any value of m, but $P[C_{max}]^{pro}$ increases as m is incremented.

7.4 Conclusions

The manufacturing industry is becoming increasingly complex and competitive. Companies need powerful optimization algorithms to design proper strategies that make them efficient in order to remain in the market. Although there is an extensive literature on classical scheduling problems, there is a lack of works on richer and more realistic problems. This chapter studies a novel problem called DPFSP-ST. It consists in the manufacturing of a product that requires several jobs that are performed in independent factories. The sub-problem of each factory can be modelled as a PFSP-ST. All factories are expected to finish at a given deadline or before. This problem describes several real-life applications where a company acquires intermediate products from others and assembles them to obtain a final product with a higher added value.

Three algorithms are proposed to deal with this problem which aim to minimize a different objective function: the makespan (ignoring the stochasticity), the expected makespan and the makespan percentile given a probability p. This percentile is the value below which a given proportion p of makespans fall when simulating scenarios, and can be interpreted as follow: if the starting time in a factory is set to the deadline minus this percentile, the processing of the product will be finished before or at the deadline with a probability p. While all algorithms rely on the ILS metaheuristic, the second and the third ones are simheuristic algorithms, i.e., integrate MCS techniques in order to deal with the stochasticity. Note that the second algorithm is intended to provide good results on average whereas the third one aims to guarantee that the manufacturing

will be finished by the deadline with a given probability. A set of computational experiments enable the comparison of the algorithms in terms of makespan, expected makespan and makespan percentile, and quantify these differences. It is proven that: *(i)* gaps among algorithms for each measure increase as the level of stochasticity is incremented; *(ii)* while there is a strong correlation between simheuristic algorithms (in the sense that solutions having the best performance in terms of expected makespan are also of good quality regarding makespan percentile, and the other way around), it is weaker between the first algorithm and any of the others; *(iii)* in some cases the differences between the second and the third algorithm may be significant, so a priority must be set by the decision-maker; *(iv)* the fact that the algorithms are so fast enable the running of the third one considering different probabilities, which provides a deeper insight of the relationship between probability (related to the risk-aversion, i.e., how sure decision-maker wants to be about finishing at a given deadline or before) and makespan percentile (i.e., how much time he needs to start before the deadline); *(v)* the effect of using different seeds is significant; and *(vi)* the makespan percentile linearly depends on the number of factories, jobs and machines, and the interaction between number of factories and machines.

Chapter 8

Applications in finance

This chapter reviews works using metaheuristics in portfolio optimization and risk management, studies the deterministic and stochastic portfolio optimization problem, and presents an application to stocks and individual commodity futures contracts. It proposes hybrid algorithms based on the ILS or the VNS metaheuristics, and MCS.
It is based on the following journal articles: Doering et al. (submitted[a]), Kizys et al. (submitted), Calvet et al. (submitted[b]), and Doering et al. (submitted[b]).
This work has been presented at the following conferences: Doering et al. (2016b), Calvet et al. (2016f), and Doering et al. (2016a).

8.1 Introduction

Investments play an essential role in improvements of welfare standards. This striving for improvement is represented through the formulation of optimization problems for most of the questions in financial economics. Traditionally, exact methods have been employed. The current internationalization and integration of financial markets and institutions has caused financial decision-making processes to become even more complex. Metaheuristics constitute an attractive alternative for problem solving in the financial sector (Gilli et al., 2011).

The second section reviews the literature on metaheuristic optimization applications for portfolio and risk management in a systematic way. The linkages between portfolio optimization and risk management are identified. It is expected that the revocation of the strict classification of financial COPs can lead to a methodological transfer of knowledge. In addition, the trends that have gradually become apparent in the literature are outlined.

The third section focuses on a single-period version of the constrained mean-variance POP. Three realistic constraints are considered. First, justified on the grounds of the investor's preference, the *pre-assignments* force some specific assets to be included in the portfolio. Second, the *quantity* constraint keeps the quantity of each selected asset within user-specified *floor* and *ceiling* values. The ceiling rules out excessive exposure to a specific asset. The floor is introduced in order to rule out the possibility of tiny (and therefore disproportionately costly) fractions of assets. Third, the *cardinality* constraint, which imposes a floor and a ceiling on the number of assets included in the portfolio, accounts for the fact that diversification benefits decrease when the portfolio features a huge number of assets. In the presence of these constraints, the problem becomes $\mathcal{NP}$-hard (Bienstock, 1996) and, thus, exact optimization methods quickly lose their efficiency as the number of considered assets grows. A matheuristic algorithm is devised for rich portfolio optimization (ARPO) that is based on the combination of an ILS metaheuristic, quadratic programming, and biased randomization strategies.

Contrary to the well-established real-life constraints, the growing body of literature assumes constant rates of returns and covariances. This empirically unsupported assumption poses a key limitation when real-life approaches are sought. The aim of the fourth section is to address this limitation. Indeed, since asset returns are random variables that obey certain probability density functions, and future returns are unpredictable, the minimum desired rate of return may not be attained with certainty. More concretely, the above simplifying assumptions are relaxed, and rates of returns and covariances are considered random variables. The resulting problem is known as the SPOP. Here a simheuristic algorithm to solve it is proposed based on the VNS metaheuristic. While the metaheuristic generates promising portfolios, simulation techniques are applied to: *(i)* estimate the expected risk of these portfolios under uncertainty conditions; *(ii)* complete a risk

analysis on each portfolio; and *(iii)* provide feedback to the metaheuristic in order to better guide the searching process.

Finally, the last section addresses the rich POP, considering individual futures contracts in addition to stocks. Recently, stock markets have become more integrated, resulting in higher positive correlation among individual stocks and thus diminishing successful diversification (You and Daigler, 2012). Because most research on metaheuristics applied to POPs relies on pre-established benchmarks, the outcomes of such a development on the quality of the established portfolios cannot be detected. Thus, it would be convenient to include a second asset class to exemplify possible diversification benefits. Individual commodity futures contracts are selected because they have been found to have low correlations with stocks (Jensen et al., 2002; Chong and Miffre, 2010). Their correlational properties have been found to be caused among others by an opposite reaction of futures to macroeconomic shocks (Silvennoinen and Thorp, 2013; Bansal et al., 2014).

8.2 Survey on metaheuristics in portfolio optimization and risk management

The increasing popularity of the application of metaheuristics to POPs and risk management problems (RMPs) is depicted in Figure **??**, based on Scopus-indexed publications. The search for POPs was conducted by examining the articles that explicitly consider portfolio optimization, index tracking or project selection in the abstract, title or keywords and make use of metaheuristics. For risk management problems, the search terms were bankruptcy, credit risk or stock or foreign exchange trading. In the case of portfolio optimization, it becomes obvious that the trend in publications is increasing. Continuing increases in computing power, the advancement of metaheuristic frameworks and parallelization strategies favour metaheuristics when dealing with $\mathcal{NP}$-hard financial COPs. On the contrary, risk management problems seem to have received much less attention. These proportions are broken down in Figure **??**, which shows that traditional portfolio optimization represents the majority of metaheuristic applications.

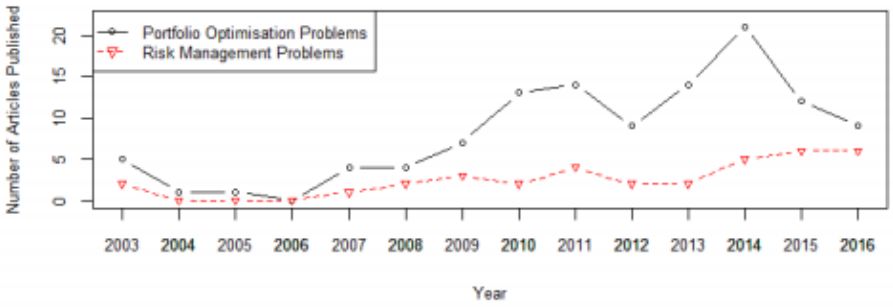


FIGURE 8.1: Scopus-indexed publications applying metaheuristics to POPs and RMPs for the period 2003 to 2016-1 (first semester).

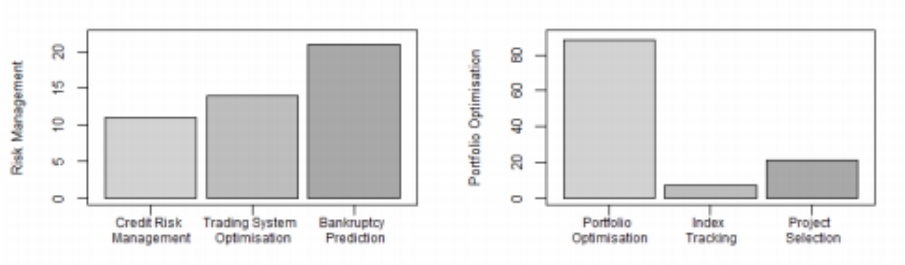


FIGURE 8.2: Number of publications on POPs and RMPs.

8.2.1 Portfolio optimization

Since Markowitz (1952) developed the portfolio optimization theory centred around the mean-variance approach, the academic community has been highly engaged in advancing the tools for portfolio optimization. The theory is based on two constituting assumptions, namely: *(i)* the financial investors being concerned with the expected returns; and *(ii)* the risk of their respective investment. It is thus the goal to minimize the level of risk expressed through the portfolio variance for a given expected return level, resulting in the so-called unconstrained efficient frontier, from which the portfolio choice is determined by the risk awareness of the investor. This established the POP, which is a strategy of: *(i)* selection of financial assets; and *(ii)* determination of the optimal weights allocated to those assets that results in a desired portfolio return and associated minimum level of risk. Based on the investor's involvement with the asset selection, two types of investment management strategies can be identified. On the one hand, active investment strategies aim at beating market returns. On the other hand, passive investment strategies aim at replicating a benchmark index.

Table **??** presents a summary of the metaheuristics applied to each of the problems reviewed: single-objective portfolio optimization, multi-objective portfolio optimization, index tracking, enhanced index tracking, and project portfolio selection. The number of articles found on each topic and metaheuristic is included inside each cell. The classical portfolio optimization is an active investment strategy, particularly when active re-balancing of the portfolio takes place in multi-period observations and, by its nature, investment appraisal requires the active selection of project portfolios. Index tracking is traditionally a passive strategy, while enhanced index tracking involves active management to some extent.

Table 8.1: Application of metaheuristics and hybridization to POPs.

Optimization problem	Single-solution search				Population-based search										Hybrid
	SA	TS	FD	SD	GA	FA	ACO	DE	EA	ABC	PSO	IWO	AIS	SS	
Single-objective portfolio optimization	2	3	1	1	2				1		3				3
Multi-objective portfolio optimization					2	2		2	1	1	4				2
Index tracking	1				2			2	3			1			3
Enhanced index tracking	1	1	1	1	2			1			1		2		1
Project selection		3			2		6				1			2	5

Traditional portfolio optimization

While the original Markowitz problem can be solved using quadratic programming, metaheuristics have increasingly been employed to cope with the fact that the problem becomes $\mathcal{NP}$-hard when more realistic constraints are introduced (Beasley, 2013). In effect, cardinality constraints, quantity constraints, and pre-assignment constraints have received overwhelming attention in the literature.

Single-objective portfolio optimization The classical POP can be considered a single-objective optimization problem with either one of the following model formulations: the investor minimizes the risk exposure subject to a minimum attainable expected return, or the investor maximizes the expected return for a given maximum level of risk. The first variant can be formulated as follows (Chang et al., 2000): A quadratic objective function is computed by aggregating over the covariances of the constituent asset returns and then minimized:

$$Min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{8.1}$$

subject to a minimum desired rate of return, the constraint that the weights have to add up to one, and the constraint that all asset weights must lie between zero and one, inclusive, thus eliminating short selling as a measure of preventing investors from excessive risk-taking by restricting them to the available budget. In formal terms:

$$\sum_{i=1}^{N} w_i \mu_i \geq R \tag{8.2}$$

$$0 \leq w_i \leq 1, \quad \forall i \in \{1, 2, \ldots, N\} \tag{8.3}$$

where N is the total number of available assets, μ_i is the expected return of an asset i, R is the minimum required return, w are the respective weights of the assets, and σ_{ij} is the covariance between two assets i and j.

Chang et al. (2000) solve it using three metaheuristic approaches (GA, SA, and TS) in order to generate a cardinality-constrained efficient frontier. They suggested pooling the results from the different approaches because no single heuristic was uniformly dominating. However, Soleimani et al. (2009) introduce sector capitalization and minimum transaction lots as further constraints and find that the GA they developed outperformed TS and SA. Following the suggestion of Chang et al. (2000), Woodside-Oriakhi et al. (2011) explore the pooling option. They found that, on average, SA contributes little to the performance of the process and that a pooled GA and TS algorithm is superior to single metaheuristic approaches at the expense of higher computational time.

As for the application of strict single metaheuristic methodologies, PSO has been found to be competitive with all three of the previously employed algorithms (GA, TS, and SA) for the cardinality-constrained portfolio selection problem and especially successful in low-risk portfolios (Cura, 2009). To evaluate the performance of PSO for even more realistic instances, Golmakani and Fazel (2011) further introduce minimum transaction lots, bounds on holdings, and sector capitalization in addition to cardinality constraints. These authors applied a combination of binary PSO and improved PSO (CBIPSO), and found that CBIPSO outperforms GA in that it provides better solutions in less computing time. As constraints become increasingly complex, the question of constraint-handling in determining feasible solutions arises. Reid and Malan (2015) investigate this research line and develop a portfolio repair constraint handling technique applied in a PSO portfolio optimization.

Di Tollo and Roli (2008) provide a review on the applications of metaheuristics and some of the proposed constraints explicitly highlighting the potential use of hybrid approaches. Likewise, such a hybrid method is proposed by Maringer and Kellerer (2003), who combine principles of SA and EA to optimize a cardinality-constrained portfolio. By combining exact mathematical programming and metaheuristics, Woodside-Oriakhi et al. (2011) further hybridize and create different matheuristics. This option is also investigated by Schaerf (2002) and Di Gaspero et al. (2011) who respectively combine TS and first descent (FD) and steepest descent (SD) local search metaheuristics with quadratic programming to optimize a portfolio while accounting for cardinality constraints, lower and upper boundaries for the quantity of an included asset, and pre-assignment constraints. Concerning optimality, Cesarone et al. (2013) develop an exact increasing set algorithm that, for small instances, solves the POP with quantity and cardinality constraints optimally and can be extended into a heuristic procedure to account for larger instances. It outperforms the metaheuristics employed by Di Gaspero et al. (2011) and Schaerf (2002).

Multi-objective portfolio optimization

Multi-objective optimization methods combine two objective measures into a single one that is to be optimized (Mishra et al., 2014) or, more often, find a set of Pareto solutions while balancing two or more objective functions simultaneously. With respect to single-objective optimization methods that require the *ex-ante* definition of an acceptable degree of profitability, multi-objective optimization requires no previous knowledge about the investor's degree of risk aversion and is thus a more general approach transferrable to different decision-makers. The approach of combining risk and return characteristics into a single objective function is taken by Zhu et al. (2011). They introduced the Sharpe ratio as a simultaneous measure.

According to Streichert et al. (2003), the multi-objective POP can be formulated as follows. It becomes necessary to minimize the portfolio risk expressed by the portfolio variance:

$$Min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} \tag{8.4}$$

while maximizing the return of the portfolio, i.e.:

$$Max \sum_{i=1}^{N} w_i \mu_i \tag{8.5}$$

subject to:

$$\sum_{i=1}^{N} w_i = 1 \tag{8.6}$$

$$0 \leq w_i \leq 1, \quad \forall i \in \{1, 2, \ldots, N\} \tag{8.7}$$

Alternatively, Equations **??** and **??** can be combined into a single one by incorporating objective weights as follows (Mishra et al., 2014):

$$Min\ \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} - (1 - \lambda) \sum_{i=1}^{N} w_i \mu_i \tag{8.8}$$

subject to the aforementioned constraints. In this case, the weights as determined by the parameter λ represent the risk aversion of the investor. By varying this parameter and running repeatedly, a Pareto efficient frontier can be established. Because of the high performance of PSO in solving the single-objective POP, enhanced PSO algorithms for solving the multi-objective POP have been proposed by Deng et al. (2012) and He and Huang (2012). Cardinality and bounding constraints are incorporated by Deng et al. (2012) who find that their algorithm mostly outperforms GA, SA, and TS algorithms as well as previous PSO approaches, especially in the case of low-risk portfolios. Similarly, He and Huang (2012) propose a modified PSO (MPSO) algorithm that outperforms regular PSO for their optimization sets. More recently, they also develop a PSO to deal with discontinuous modelling of the POP and find that it generally outperforms PSO and also performs better than MPSO in larger search spaces (He and Huang, 2014). Other population-based algorithms applied in optimizing cardinality-constrained portfolios include firefly algorithms (FA) (Tuba and Bacanin, 2014b) and artificial bee colony (ABC) algorithms (Tuba and Bacanin, 2014a). The authors hybridized FA and ABC by incorporating the FA search strategy into ABC to enhance exploitation and found that their data suggested the superiority of the methodology compared to GA, SA, TS, and PSO (Tuba and Bacanin, 2014a). Streichert et al. (2003) account for further constraints: buy-in thresholds (acquisition prices) and round lots (smallest volume of an asset that can be purchased). They employ two MOEAs: a GA and an EA enhanced through the integration of a local search that applies Lamarckism, thus allowing the individual improvements to be passed on to the offspring.

Nevertheless, apart from the neglect of realistic non-linear constraints, there is a second point of criticism to the original Markowitz model, namely its assumption of normal financial returns, which, in reality are characterised by a leptokurtic distribution (Krink and Paterlini, 2011), making it necessary to consider non-parametric risk measures. Such a measure is the value-at-risk, as employed by Babaei et al. (2015) who develop two multi-objective algorithms based on PSO to solve a cardinality- and quantity-constrained POP. Through splitting the whole swarm into sub-swarms that are then evolved distinctly, their methodology outperform similar benchmark metaheuristics. In order to optimize a non-parametric value-at-risk and to include further constraints, including lower and upper bounds for the weights of included assets, a threshold for asset weight changes, lower and upper bounds for the weights of one asset class and a turnover rate that determines the maximum asset allocation changes possible at once, Krink and Paterlini (2011) develop the differential evolution (DE) for multi-objective portfolio optimization algorithm. An extended version of a generalised DE metaheuristic is also employed in optimizing a highly constrained POP by Ayodele and Charles (2015). The included constraints consist of bounds on holdings, cardinality, minimum transaction lots, and expert opinion. An expert can form an opinion based on indicators beyond the scope of the analysed data and influence whether or not an asset should be included. Lwin et al. (2014) consider cardinality, quantity, pre-assignment and round lot constraints and develop a MOEA that is improved through a learning-guided solution generation strategy, which promotes efficient convergence. It has been shown that the developed algorithm outperforms four benchmark state-of-the-art MOEAs.

An extensive review of the application of EAs to the POP is provided by Metaxiotis and Liagkouras (2012). Likewise, for an extensive review on different POPs, including single- and multi-objective optimization, the reader is referred to Mansini et al. (2014).

Passive investment

Passive investment strategies have received less attention in the optimization literature. They are characterized by limited on-going buying and selling, as well as by ensuing limited maintenance. Based on the traditional capital market theory stating that market portfolios offer the greatest return per unit of risk, passive investment strategies have been shown to outperform actively managed funds and thus gained popularity (Alexander and Dimitriu, 2004).

Index tracking

The index tracking problem (ITP) is a portfolio management strategy, in which investors aim at mimicking a market or sector index. This is done by either replicating the index or by selecting a portfolio that follows the index behaviour as closely as possible without including all the stocks that make up the original index. In the case of perfect replication, there are transaction costs associated with updating the portfolio to continuously accurately depict the index. Therefore, the ITP is largely concerned with the latter, partial replication. There are thus two stages in index tracking, the common goal of which is to minimize the resulting tracking error. The first consists of selecting the assets to include in the portfolio and the second relates to determining the weights. It consists of a combinatorial and a continuous numerical problem, which both have to be addressed simultaneously (Krink et al., 2009). Once similar rich constraints as in portfolio optimization are introduced, it becomes extraordinarily difficult to solve with exact methods.

The problem can be addressed with the following formulation (Beasley et al., 2003). Minimize the tracking error:

$$Min\ E = \frac{[\sum_{t \in S} |r_t - R_t|^{\alpha}]^{(\frac{1}{\alpha})}}{T} \tag{8.9}$$

where $S = 1, 2, \ldots, T$ are the time periods considered, r_t is the tracking portfolio return, R_t is the return of the tracked index, and α is the penalization power that is applied to the difference between the realized return and the benchmark return. In the most basic formulation, the following constraints have to be considered:

$$\sum_{i=1}^{N} z_i = K \tag{8.10}$$

which ensures that any new tracking portfolio contains K stocks, as z_i takes on the value of one if a stock is included in the replication portfolio and zero otherwise. The weights have to be limited:

$$0 < w_i \leq 1,\ z_i = 1,\ \ \forall i \in \{1, 2, \ldots, N\} \tag{8.11}$$

The non-included stocks must naturally dispose of a weighting of zero:

$$w_i = 0,\ z_i = 0,\ \ \forall i \in \{1, 2, \ldots, N\} \tag{8.12}$$

Maringer and Oyewumi (2007) investigate partial replication and introduce cardinality constraints concerning upper and lower weight limits and integer constraints in the ITP employing a DE methodology. Their findings suggest that partial replication is indeed sufficient in replicating the benchmark index. This is due to the fact that only a decreasing marginal improvement is reached by increasing the cardinality.

Scozzari et al. (2013) develop a mixed integer quadratic programming formulation to solve the ITP including hard constraints set by the European Union on ceilings of asset inclusion weights as well as low turnover rates and resulting low transaction costs in small instances. Early research by Beasley et al. (2003) introduce a population-based evolutionary metaheuristics to solve the partial reproduction ITP with regard to stock indices including constraints on transaction costs (as well as a ceiling for the total inclusion of stocks). Derigs and Nickel (2004) develop a two-stage SA metaheuristic, in which they control for cardinality constraints and transaction costs through turnover volume restrictions.

For larger instances, especially in multi-period analysis, Scozzari et al. (2013) propose hybridizing metaheuristics with exact methods. This has been done by Krink et al. (2009) who apply a DE metaheuristic combined with a combinatorial search operator. Although their methodology

initially failed to find acceptable solutions, they show that extending DE with a search operator by selecting the assets with highest weights in the benchmark improves the results greatly in comparison with GA, SA, and PSO. Ruiz-Torrubiano and Suárez (2009) employ a GA hybridized with quadratic programming. More recently, Ni and Wang (2013) also tackle the ITP employing a hybridized GA with increased learning ability that is enabled through goal programming. The authors include cardinality and integer constraints, as well as proportion constraints for individual portfolio assets. While both methodologies yielded successful solutions, the models neglect transaction costs. The trade-off between transaction costs and tracking performance has been investigated by Chiam et al. (2013) who develop a multi-objective evolutionary index tracking platform that considers multiple periods and simultaneously optimizes tracking performance and transaction costs while considering round lots and non-negativity constraints as well as floor constraints as buy-in threshold to prevent unnecessary transaction costs and capital injections.

Affolter et al. (2016) find that due to the missing measure to define the distance between portfolios with respect to their assets and weights, invasive weed optimization (IWO) do not lead to satisfactory optimization results. Di Tollo and Maringer (2009) have created a framework for classifying the metaheuristics applied to ITP and present a review of the literature.

Enhanced index tracking

Beasley et al. (2003) define an objective function that accounts for a trade-off between the tracking error and excess returns above those of the benchmark index. Considering that investors might see a trade-off between the trading error and excess returns above the index has led to the enhanced index tracking problem (EITP), in which investors aim at beating the benchmark index. The EITP then becomes a multi-objective optimization problem, in which the tracking error is minimized while maximizing the degree of beating the benchmark index so that a solution dominates another if the excess return is higher given the same level of trading inaccuracy or if the trading accuracy for the same level of excess return exceeds that of the other solution. This can be formulated by including a second objective function that defines the excess return between r_t and R_t:

$$Max\ r^* = \sum_{t \in S} \frac{r_t - R_t}{T} \tag{8.13}$$

Canakgoz and Beasley (2009) solve the ITP as well as the EITP including transaction costs, an upper limit on the total number of stocks purchased, and a limit on the incurred transaction costs using exact methods (mixed-integer linear programming formulations). However, Li et al. (2011a) show they could mostly outperform the methodology employed by Canakgoz and Beasley (2009) by implementing an immunity-based optimization algorithm. Including further constraints, Li and Bao (2014) also employ an immunity-based multi-objective optimization algorithm with non-negativity and floor and ceiling buy-in thresholds. They conclude that the inclusion of optimization of the tracking process is valuable. A perfectly enhanced tracking portfolio would outperform the index by a low-frequency trend such as steady excess return while negative returns should be trendless and characterized by high frequency variation. Thus, the tracking process can be enhanced by considering different frequencies for tracking error and excess returns when the former is minimized and the latter maximized (Li and Bao, 2014). Optimization of the tracking process is expected to increase in importance for multi-period assessment; the authors, however, leave this for further research. The question of multi-periodicity has been investigated by Andriosopoulos et al. (2013) who address the EITP employing both DE and GA. They show that the so-constructed mimicking portfolios inhibit less risk compared to the underlying benchmark index, while replicating their performance. Nevertheless, they conclude that the GA version outperforms DE in terms of minimum tracking errors, as well as maximum mean excess returns. As they explicitly consider different time horizons for rebalancing the portfolio, these authors reinforce the idea that there exists a trade-off between transaction costs, which decrease with longer rebalancing periods, and Sharpe ratios (as a measure of the tracking performance and profitability), which is negatively impacted by decreased rebalancing frequency as investigated by Chiam et al. (2013) for the ITP.

An alternative approach is pursued by Guastaroba and Speranza (2012) who apply a kernel search framework to both the ITP and the EITP. They introduce the possibility that an investor already holds a portfolio as a further constraint to consider in addition to transaction costs. However, they treat the EITP as a single-objective optimization by outperforming the market index, while keeping the tracking error below a given threshold. Compared to a general-purpose solver,

the performance of the kernel search model is superior. Thomaidis (2011) consider an EITP problem with restrictions on the maximum of tradable assets, and employ fuzzy set theory to consider non-standard investment objectives, such as the probability of under-performing. The resulting cardinality-constrained problem is solved using nature-inspired optimization techniques: SA, GA, and PSO.

Lastly, while some authors declare active and passive portfolio management as mutually exclusive concepts, the close connection between index tracking and portfolio optimization could be illustrated by the approach taken by Di Tollo et al. (2014) who combine the two methods in a multi-criteria optimization problem. They employ a hybrid metaheuristic consisting of local search metaheuristics (FD, SD and TS) and quadratic programming to estimate the efficient frontier. Combining the concepts of risk and return with tracking error leads to a three-dimensional objective function and Pareto frontiers. Their methodology is found competitive in performance with other metaheuristics such as TS.

Project portfolio selection

Unlike banks and institutional investors, non-financial companies as well as governments are faced with a different type of portfolio choice. As a method to determine which proposals to pursue and the corresponding budget allocation, investment or project appraisal is related to portfolio optimization in its goal of maximizing a benefit figure. Usually, decisions cannot be altered or adjusted during the course of the projects, or at least not without incurring financial losses. This problem becomes $\mathcal{NP}$-hard due to its sheer complexity (Fernandez et al., 2015). It is a multi-period problem and the budget-allocating entity usually pursues several conflicting objectives, some of which can be of qualitative nature. For that matter, Doerner et al. (2004) propose a two-stage procedure. First, the Pareto frontier is constructed. Then, in the second phase, it is interactively explored by the decision-maker to account for personal preferences. A formal description of this problem, based on the one presented in Doerner et al. (2004), is included next. The benefit function $b_{lt}(x)$ that comprises the value of the l different benefit groups, such as generated funds, cash flows, patents or other beneficial outcomes is to be maximized over all time periods t for all included projects, i.e.:

$$b_{lt}(x) = \sum_{i=1}^{N} b_{ilt} x_i \tag{8.14}$$

where x_i is a binary variable that takes on the value of one for included projects and zero otherwise, subject to constraints concerning resource limitations R_{qt} that apply to all resource categories r_q, such as budget, capacity, or manpower, as well as minimum benefit requirements B_{lt}:

$$r_{qt}(x) \leq R_{qt}, \forall q \in \{1, \ldots, R\} \text{ and } \forall t \in \{1, \ldots, T\} \tag{8.15}$$

$$b_{lt}(x) \leq B_{lt}, \forall l \in \{1, \ldots, B\} \text{ and } \forall t \in \{1, \ldots, T\} \tag{8.16}$$

Because of the modelled similarities, the methodological approaches employed are inspired by the research on traditional portfolio optimization. Early work (Ghasemzadeh and Archer, 2000) conduct optimization after the construction of a weighted objective function and constraints concerning budget and man-hours in an integer linear programming approach. However, instances are very limited because the authors aspired a comparison between manually computed portfolios and those constructed employing their decision support system. For their metaheuristic approach Doerner et al. (2004) employ a Pareto ACO (P-ACO). As there are possible synergies between projects that should be evaluated, the authors make an attempt at incorporating these considerations into their methodology and point out that P-ACO specifically constructs project portfolios through pheromone vectors. This has two advantages. Firstly, infeasible solutions are avoided and secondly, project interactions can more naturally be considered in the construction of solutions. They further take into account floor and ceiling constraints for inclusion of projects from any given subset, as well as resource limitations and minimum benefit requirements for individual projects. Compared to Pareto SA and a non-dominated sorting GA (NSGA), P-ACO yields the most efficient results. This approach has been then further enhanced by Stummer and Sun (2005),

who compare the performance of a P-ACO procedure enhanced through adding a neighbourhood search routine, a TS procedure, and a variable neighbourhood procedure. Their findings suggest that the improved P-ACO model performs better than TS with many objective functions and a large set of efficient solutions. Furthermore, Doerner et al. (2006) conclude that including both a learning and a two step integer linear pre-processing procedure to initialize efficient project portfolios improves performance of the P-ACO algorithm.

More recently, research has also drawn on findings from other areas, such as scheduling: Gutjahr et al. (2008) and Gutjahr et al. (2010) also take employee competencies and the evolution of their knowledge scores over time through learning or depreciation into account. While the earlier work optimizes a weighted average objective function using ACO and GA metaheuristic procedures and finds the GA to be superior when the search space is not highly constrained, the authors develop a multi-objective optimization model, which simultaneously optimizes the objectives of maximum economic gains and aggregated competence increase in their later work. They also divide the problem into master and slave subproblems, the first of which is concerned with the project selection, while the slave problem optimizes the allocation of personnel to the projects over time. Although the slave problem can be solved using exact methods, the master problem is solved using the NSGA-II and P-ACO metaheuristics. NSGA-II outperforms P-ACO in synthetic instances, while P-ACO outperforms NSGA-II for real-life instances. Carazo et al. (2010) include scheduling as a continuative concept following the project selection. Their developed metaheuristics approach is based on SS for project portfolio selection (SS-PPS). As previous work, they also consider interdependences between different projects and can show that their model outperforms other heuristic approaches based on EA. Similar to Rabbani et al. (2010), who present a multi-objective PSO metaheuristic that is competitive with respect to SPEA II, Urli and Terrien (2010) formulate the project portfolio selection problem as a multi-objective non-linear integer program, which they solve using the SSPMO metaheuristic (Molina et al., 2007). In a first phase, they generate an initial set of efficient solutions through TS and then combine these via SS. While this approach solves small and medium instances in satisfactory computation time, the determination of all non-dominated project portfolios still remains difficult when considering large instances (100 projects or more).

Another issue that has only recently been addressed is project divisibility. While business projects are at least partially indivisible, research projects funded by governments can often also be executed with partial funding and it is thus a further question how much of the sought after funding is awarded. Cruz et al. (2014) use ACO in solving a stationary project portfolio optimization problem, in which partial support of the requested budget is allowed. They develop a non-outranked ACO approach, incorporating a fuzzy outranking preference model. Outranking is employed in an *a priori* preference system in order to model that decision-makers will have preferences towards different portfolios on the efficient frontier based on their personal goals. Fernandez et al. (2015) further enhance this approach by including integer linear programming methods to generate an initial population. They also include synergies in their optimization. It can be asserted that project synergies, project divisibility, the incorporation of multi-periodicity, and outranking are prominent real-life constraints that increase the complexity of the portfolio selection process.

8.2.2 Risk management

Risk management of companies refers to the evaluation of data concerning the institution's exposure to a certain source of risk and it is further concerned with statistics on trends that will influence that exposure in the future. While quantitative data is relevant and necessary for this, it must be complemented by qualitative information for informed decision-making (Chorafas, 2007). Risk management is addressed in terms of optimization through metaheuristics for credit risk assessment and the resulting bankruptcy prediction. García et al. (2015) provide a review of systems and applications to the optimization of trading rules in the financial markets. Table **??** presents the metaheuristic methodologies applied to the different subproblems of risk management.

TABLE 8.2: Application of metaheuristics and hybridization to risk management.

Optimization problem	Single-solution search		Population-based search										Hybrid
	SA	TS	GA	ACO	EA	ABC	PSO	SS	HBMO	FA	BA	HS	
Credit risk assessment		2	4	1			1	1	1				6
Bankruptcy prediction			4		1		2						6
Optimization in stock trading	2		4			1	2			1	1	1	7
Optimization in foreign exchange trading			3		1								4

Several conclusions can be drawn. Firstly, GA are the preferred metaheuristics in risk management. Furthermore, PSO has also received widespread attention. Contrary to that, more exotic algorithms, such as harmony search (HS), FA, or bat algorithms (BA). Secondly, it can be seen that bankruptcy prediction, as well as optimization of trading systems for foreign exchange markets, have received less attention and have been approached with fewer methodologies. They thus represent interesting future research lines. Thirdly, it becomes evident that hybridization among metaheuristics or other optimization methods is far more prevailing in risk management optimization than in portfolio optimization. Lastly, relatively recently developed metaheuristics, such as IWO and honey bees mating optimization (HBMO), have not been applied as comprehensively as well-established ones.

Credit risk assessment and optimization

Credit risk assessment is one of the most researched topics in the banking industry. There are many different approaches for financial institutions. However, during the last years, non-financial companies have also recognized the need to treat their trade credits to customers with the same caution and scrutiny. While the use of metaheuristics is still scarce, they are increasingly used as a pre-processing procedure in order to identify the most relevant predictors of credit risk in the analysis of large datasets. Marinakis et al. (2008) classify a set of companies into different classes of credit risk level. They propose and compare TS, GA, and ACO for solving the feature selection subset problem. The accuracy measures are determined by whether or not a subject has been classified in the right category (Table **??**).

TABLE 8.3: Definitions of the classified and the misclassified samples.

		Actual class	
		1	2
Estimated class	1	T_1	F_2
	2	F_1	T_2

The overall classification accuracy (OCA) can serve as optimization objective that is to be maximized:

$$OCA = \frac{T_1 + T_2}{T_1 + F_1 + T_2 + F_2} \cdot 100. \tag{8.17}$$

More recently, Marinaki et al. (2010) employ HBMO in determining the relevant features. Their metaheuristic reduces the number of used features by more than half and still yields superior results compared to PSO, ACO, GA, and TS. Oreski et al. (2012) employ NN hybridized with GA to enhance the classification accuracy of the NN classifiers by choosing optimal features. Oreski and Oreski (2014) further improve the results by employing a hybrid GA instead of GA. Their results suggest that they achieve a higher and less volatile accuracy with fewer features through a reduction of the search space and an incremental phase of the GA. Chi and Hsu (2012) employ GA in selecting relevant variables to combine a bank's internal behavioural scoring model with an external credit bureau scoring model and thus creating a dual scoring model that outperforms the individual model. A survey on the importance of employing the right fitness function in the GA is provided in Kozeny (2015).

Trends in credit risk assessment concern the hybridization of metaheuristics with other techniques for feature selection. Wang et al. (2010) develop a feature selection based on rough set and TS. Similarly, Wang et al. (2012) use a rough set and SS feature selection able to improve results in all three considered base sets, i.e. NN, J48 decision tree and LR. Lastly, Danenas and Garsva (2015) pursue the idea of optimizing the classifiers of a linear SVM using PSO. While their results

are comparable to the use of other classifiers (LR and radial basis function or RBS networks), the proposed methodology delivers less stable performance.

Bankruptcy prediction

Bankruptcy occurs when debtors are unable to repay outstanding debts. While bankruptcy prediction constitutes part of the credit risk evaluation process, it is vital for banks and companies to constantly monitor their credit risk exposure. Because of the two-classes framework (firms that go bankrupt and firms that do not), the basic optimization framework is similar as suggested for credit risk assessment. The difficulty and difference lies in the relatively longer aspired forecasting period and the difficulty in predicting the exact time of bankruptcy.

It is worth considering to differently value the two classes of mistakes that occur. While falsely classifying a subject as bankruptcy candidate merely leads to missed revenues, a false classification as healthy company usually leads to a failure on a payment and thus has greater consequences.

Early research conducted by Back et al. (1996) highlight the contribution of GA in predicting bankruptcy when hybridized with NN. Shin and Lee (2002) introduce the prediction of corporate bankruptcy using GA and historical financial data. Kim and Han (2003) further employ GA to extract decision rules based on qualitative expert decisions and find their methodology to be superior compared to NNs or inductive learning methods because the rules created by GA are more accurate and have larger coverage. An extensive survey can be found in Kumar and Ravi (2007) who review both statistical and computing methods. Their evaluation conclude that all statistical methods are outperformed by back propagation NNs. They further highlight the prediction accuracy of SVM. More recently, Kirkos (2015) presents the literature on artificial intelligence and machine learning techniques employed in bankruptcy prediction.

Min et al. (2006) improve the performance of SVM with regards to optimizing the feature subset and parameters simultaneously. They show that selecting an appropriate feature subject has implications for the kernel and that their integration improves bankruptcy prediction accuracy. Chen (2011) highlights that while intelligent techniques provide higher prediction accuracy for smaller datasets and are adversely affected by increasing datasets, statistical methods perform more accurately when the dataset is large. But the author also indicates that a hybrid between PSO and SVM may yield a good balance between short- and long-term prediction accuracy. This is consequently done by Lu et al. (2015) who combine switching PSO (SPSO) and SVM. The SPSO is employed in searching the optimal parameter values of RBF kernel of the SVM and the authors show that this hybridization yields superior results to GA-SVM and PSO, respectively. These findings are supported by Chen (2011) and Chen (2014) who also employ PSO-SVM and show high accuracy with a reduced number of parameters. Furthermore, Gaspar-Cunha et al. (2014) propose an evolutionary approach that simultaneously minimizes the number of features and maximizes the accuracy of the classifier in SVM.

Recently, ensemble learning has been applied to the bankruptcy prediction problem. Kim and Kang (2010) propose hybridizing an ensemble with NNs and show that it improves prediction accuracy compared to regular NNs. However, these attempts often suffer from high correlation among the individual classifiers, and thus Kim and Kang (2012) improve their methodology to include a GA-based coverage optimization to alleviate multicollinearity through classifier selection. More recently, Davalos et al. (2014) develop an accurate GA-based ensemble classifier model with heterogeneous instead of individual classifiers that is comprehensible due to its if-then-structure.

However, the financial ratios employed in the main research lines are unavailable for a large portion of companies. Small and medium-sized enterprises (SMEs) do not dispose of regular audited financial data or market prices and public ratings due to publicly traded equity or debt instruments and it is necessary to include available and relevant indicators for these individual firms. Thus, with special regards to SMEs, Gordini (2014) compares the prediction accuracy of GA, SVM, and LR. The author shows that the prediction of GA is superior.

Optimization of decision-support systems for trading

The development and optimization of automated trading systems has become of prevalent importance for broker investment banks and other institutional investors alike. A large portion of the literature addresses stock trading, while some researchers have concentrated on the foreign exchange markets.

Stock market trading

Derigs and Nickel (2003) develop a decision support system (DSS) for portfolio optimization and index tracking. They stress the importance of hard (government-imposed and compulsory) and soft (shaped by preferences of the investor) constraints. They implement a local search guided by SA in order to optimize the DSS with respect to floor and ceiling constraints and transaction costs. These authors show for the application to passive tracking of the DAX 30 that their system delivers solutions with minimal tracking errors in acceptable computing time. Focusing on real-time decisions, Chavarnakul and Enke (2009) propose a trading system for the stock market based on volume adjusted moving average that is hybridized with NN to decrease the time of receiving trading signals, fuzzy logic to deal with uncertainty, and GA techniques to optimize the trading signals to overall increase efficiency. Depending on the strength and direction of a given signal, the system assumes a buy or sell position. If the signal is not confident enough, a hold position is taken. The so established neuro-fuzzy based GA is shown to lead to fewer trades and thus reduce transaction costs, while profitability is increased.

Gorgulho et al. (2011) also propose a system to automatically manage a portfolio of assets and highlight the necessity of adapting the system to the state of the market. They employ GA and technical analysis rules. The system requires the user to input the available budget, the maximum of assets to be included at any time, whether or not short selling is considered and the allowed amount of transaction costs. Then, an initial portfolio is constructed employing a GA. Over the course of the investment, the system regularly updates the proposal based on technical trading rules based on closing positions and refilling empty positions. Teixeira and De Oliveira (2010) combine technical trading rules with nearest neighbour classification. Their analysis is based on historical data of stock closing prices and volume. Their system outperforms a buy and hold strategy in most cases. Because the parameters in these functions have to be determined, metaheuristics have been applied. Brasileiro et al. (2013) further refine the strategy by Teixeira and De Oliveira (2010) by searching for the best system parameters and number of lags with an ABC algorithm. Nunez-Letamendia (2007) shows that GA is robust and powerful when applied to optimizing technical trading rules. Similarly, Lin et al. (2011) apply a GA to improve trading rules for individual stocks, which are then combined with echo state networks to provide suggestions for trading. Their results show an outperformance of the buy and hold strategy. In a more recent work, Wang et al. (2014a) employ a time-variant PSO (TVPSO) to determine the optimal parameter values of a complex trading system: performance-based reward strategy (PRS). PRS combines moving average and trading range breakout trading rules. Considering transaction costs and excluding short selling, the system is able to achieve high profits and the application of the TVPSO significantly optimizes the trading system.

Hybridizations of metaheuristics and NN have recently shown to provide accurate forecasts of stock market prices. While both provide better results than a passive buy and hold strategy, HS based models have been shown to outperform GA based models with regards to forecasting errors (Göçken et al., 2016). Very recently, the hybridization of data mining techniques with metaheuristics has created clustering metaheuristics able to predict patterns in the general movement of stock markets, such as periods of lower and higher return (Prasanna and Ezhilmaran, 2015).

Foreign exchange market trading

Foreign exchange market trading can either concern hedging foreign exchange risk or speculation. Only the latter has the objective of making a profit by exploiting market inefficiencies. Myszkowski and Bicz (2010) establish a trading system based on decision trees that consider technical trading indicators. EA then generates trading strategies. While these are able to achieve a profit, the system is still too fragile to be used in automated trading. Mendes et al. (2012) propose employing GA to optimizing trading rules and although their developed trading system performs well in terms of computational time because of the small population size employed, it fails to perform well in terms of profitability when faced with transaction costs. Zhang and Ren (2010) develop a high-frequency trading system based on the optimization of technical indicators through GA that is able to produce annualized profits. In addition to intraday prediction, Evans et al. (2013) implement a trading system based on NNs, whose topology is optimized using GA. In comparison with Zhang and Ren (2010), they are able to considerably improve annualized net profits.

8.2.3 Linkage between portfolio optimization and risk management

Despite the fact they have been addressed as two independent types of problems in most of the literature, this section highlights the relationship between portfolio selection and risk management. In the first place, POPs directly consider a risk measure (such as portfolio variance, portfolio semivariance, value at risk, alpha and beta, among others) and, therefore, they can also be seen as risk management problems. For example, the specification of adequate risk measures that accurately depict return distributions is a well-established area of research. It is concerned with one-dimensional risk measures of individual assets and multi-dimensional measure to account for interaction of asset portfolios (Rachev et al., 2010).

In the second place, most risk management models related to optimization problems can be seen as rich variants of POPs. For instance, stock market trading is in the essence of the problems concerned with constructing an initial portfolio and updating it over time to reflect current macro- and microeconomic developments. Likewise, while credit risk and bankruptcy risk are only estimated in the overwhelming majority of the risk management literature, it is the ultimate goal to build low-risk portfolios by including preferably those assets with a lower credit risk and excluding other assets expected to go bankrupt. Using a MOEA, Moreno-Paredes et al. (2013) explicitly acknowledge the linkage between credit risk management and portfolio optimization by treating the loan decision among a set of customers as a credit portfolio optimization problem. More generally, implicit in a POP is pooling assets with imperfectly correlated returns that leads to a diversification of idiosyncratic sources of risk and a reduction in the overall risk of portfolio investment.

Figure **??** depicts the relationship between risk management and POPs reviewed here. The extension of the ovals representing risk management problems and POPs respectively signifies the extension of possible solving approaches beyond traditional optimization techniques. The risk management problems of bankruptcy and credit risk prediction are located directly on the border to portfolio optimization, as the predictions are generally employed in a following portfolio selection process. Foreign exchange trading, unlike stock trading, is considered a sole risk management problem. While stock trading consists of the establishment and maintaining of a portfolio, speculative profits in foreign exchange trading are generated through simultaneous buying and selling and not the establishment of a portfolio. Concerning the subproblems of portfolio optimization, the ITP and EITP do not directly consider risk measures. Unlike that, the POP is directly concerned with risk minimization and thus closely related to risk management problems.

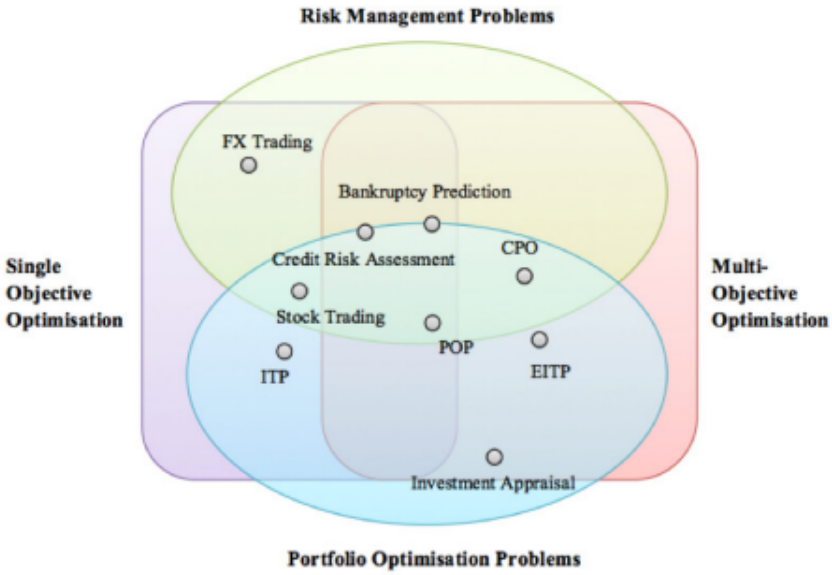


FIGURE 8.3: A unified classification of portfolio optimization and risk management.

8.2.4 Emerging trends

From the previous discussion, one clear trend is the transfer of methodological knowledge from portfolio optimization to risk management optimization. Another trend is related to the increasing complexity of the problems being addressed, which makes even more evident the need for faster (or parallelizable) metaheuristic approaches. These 'fast metaheuristics' will be needed as the models introduce further constraints to account for real-life circumstances, and as the real-time factor that is required in most of the decision-making processes will become even more relevant. Strongly related to this point, distributed and parallel computing techniques can be employed to accelerate the 'wall clock times' (Juan et al., 2013b). Furthermore, the fact that two or more objectives have to be considered simultaneously to account for the complexity has shown to increase the employment of multi-objective optimization techniques.

Another clear trend is the predominance in the use of population-based metaheuristics over single-point metaheuristics. It is our opinion that no family of metaheuristics are shown to be superior in performance (regarding both quality of solutions as well as computing times) to another. At least, we have not found any scientific evidence that supports that claim. Thus, a lot of research can be done yet regarding the use of single-point metaheuristics. Related to this, it is possible to observe a trend to develop hybrid algorithms, which combine different types of metaheuristics as well as metaheuristics with machine learning and statistical techniques. While hybridization can be an effective strategy to solve complex problems, it might also add some degree of additional complexity to the solving algorithms. This, in turn, might make them more difficult to be clearly understood, correctly implemented, and applied in practical scenarios. Adding complexity to algorithms –e.g., additional parameters that require fine tuning– also makes it difficult to reproduce their experimental results. For those reasons, only in cases in which significant improvements in performance are obtained, is the hybridization of algorithms justified.

With regards to data, recent research has shown a trend to employ different risk measures to more accurately depict the characteristics of the underlying data. This is also a particularly interesting research area as further stakeholders of financial optimizations (e.g. SMEs) do not provide traditional optimization inputs and thus alternative measurements of risk are promising. Further concerning measuring, while hybridizations of simulation and optimization have recently been developed and gained popularity in the application to SCOPs in different areas, the above finance-related problems have not yet been extensively addressed by simheuristics, even though financial data is characterised by stochastic macro- as well as firm-level uncertainty. It can thus be expected that this research line is promising, with respect to both, the design of new problems and the application of simheuristics to established COPs that previously neglected stochastic uncertainty.

8.3 The POP

The single-objective POP has been already introduced in the previous section. Here, a richer version is addressed which includes realistic constraints. A binary variable z_i $\forall i \in \{1, 2, \ldots, N\}$ is created to reveal if an asset i is selected ($z_i = 1$ if $w_i > 0$) or not ($z_i = 0$ otherwise). The number of assets in the portfolio, $\sum_{i=1}^{N} z_i$, is bounded by user-defined values, k_{min} and k_{max} (cardinality constraints). Moreover, the user can pre-select certain assets to be included in the portfolio, i.e.: $\forall i \in \{1, 2, \ldots, N\}$, $p_i = 1$ if the asset i is pre-assigned (i.e., $w_i > 0$) and $p_i = 0$ otherwise (pre-assignment constraints). Finally, for each asset i, its associated quantity in the portfolio, w_i, is bounded by user-defined values, ε_i and δ_i (quantity constraints).

The mathematical model is:

$$Min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}, \tag{8.18}$$

$$\sum_{i=1}^{N} w_i \mu_i \geq R \tag{8.19}$$

$$\sum_{i=1}^{N} w_i = 1 \tag{8.20}$$

$$0 \leq w_i \leq 1, \forall i \in \{1, 2, \ldots, N\} \tag{8.21}$$

$$k_{min} \leq \sum_{i=1}^{N} z_i \leq k_{max} \tag{8.22}$$

$$\varepsilon_i z_i \leq w_i \leq \delta_i z_i, \forall i \in \{1, 2, \ldots, N\} \tag{8.23}$$

$$0 \leq \varepsilon_i \leq \delta_i \leq 1, \forall i \in \{1, 2, \ldots, N\} \tag{8.24}$$

$$p_i \leq z_i, \forall i \in \{1, 2, \ldots, N\} \tag{8.25}$$

$$z_i \leq M w_i, \forall i \in \{1, 2, \ldots, N\} \tag{8.26}$$

$$z_i \in \{0, 1\}, \forall i \in \{1, 2, \ldots, N\} \tag{8.27}$$

Equations (**??**) – (**??**) outline the unconstrained optimization problem and determine the unconstrained efficient frontier (UEF). Specifically, Equation (**??**) provides the lower bound for the investor's required return. Equation (**??**) ensures that portfolio weights add up to unity. The purpose of Equation (**??**) is to regulate leveraged positions. By solving the constrained optimization problem given by Equations (**??**) – (**??**) the constrained efficient frontier (CEF) is obtained. Equation (**??**) formulates cardinality constraints. Equation (**??**) defines quantity constraints. A minimum and maximum quantities of wealth invested in asset i is given by ε_i and δ_i. Both parameters ε_i and δ_i range from 0 to 1 (Equation (**??**)). Given a vector of N binary decision variables Z (Equation (**??**)), and a binary vector P of pre-assignments, whenever asset i is pre-assigned, it has to be included in the portfolio (Equation (**??**)). In Equation (**??**), M is a large positive value such that $Mw_i \geq 1$ for all $w_i \geq 0$. Thus, if the quantity in the portfolio of asset i, w_i, is equal to 0, it means that this asset is not included in the portfolio (i.e., $z_i = 0$).

According to the problem description, the output of the algorithm will be an assets-investment plan, $W = (w_1, w_2, \ldots, w_N)$, satisfying all the aforementioned constraints and with the lowest possible risk.

8.3.1 Methodology

The ARPO matheuristic combines three main components: *(i)* an ILS framework; *(ii)* the use of a biased randomization process that guides the generation of promising solutions (perturbation stage); and *(iii)* the use of a quadratic programming solver that, given a current portfolio, optimizes the levels of investment of each asset (local search). Pseudo-code **??** shows the main procedure of the algorithm. Apart from the inputs defining the instance, also the maximum computing time allowed, *maxTime*, and an additional parameter, *beta*, are passed to the procedure –the use of this additional parameter will be discussed later.

The procedure starts by generating a 'dummy' initial solution. It is constructed by including the assets with the highest return levels so that it provides the highest possible expected return while satisfying all the remaining constraints. This way, if the expected return provided by this solution does not reach the minimum return threshold imposed by the investor, then the problem will be infeasible since no other solution will do it. Notice that it is also likely to obtain a high risk associated with this initial solution.

At this point, a quick local search is applied to the initial solution, which uses quadratic programming in order to optimize the investment level assigned to each asset. The improved solution will be considered both as the current 'base' solution and the 'best-so-far' solution. Now, the ARPO procedure resumes by starting an iterative improvement process. It comprises three stages: *(i)* the *perturbation* stage, which applies strong changes to the base solution in order to increase exploration of the space of solutions; *(ii)* the *local search* stage, which tries to perform a quick improvement of the base solution by using a quadratic programming and a cache memory; and *(iii)* the *acceptation* stage, which makes use of a credit-based system in order to allow accepting,

Algorithm 13 Main procedure of the ARPO algorithm.

```
    procedure ARPO(inputs, minReturn, maxTime, beta)
 1: initSol ← genInitSol(inputs)                    ▷ generate sol with highest possible return rate
 2: if {getReturn(initSol) < minReturn} then
 3:     return unfeasible                                                    ▷ unfeasible problem
 4: end if
 5: genFriendshipLists(inputs)                  ▷ generate a sorted list of "friends" for each asset
 6: baseSol ← QPOptimize(initSol, minReturn)      ▷ optimize levels for each asset in portf.
 7: baseSol ← cleanSol(baseSol)                   ▷ delete from portf. assets with level = 0
 8: bestSol ← baseSol                                                    ▷ initialize bestSol
 9: elapsedTime ← 0
10: credit ← 0                                          ▷ used in the acceptance criterion
11: while {elapsedTime < maxTime} do                                     ▷ iterated local search
12:     newSol ← perturbateSol(baseSol, inputs, beta)    ▷ destruction-construction stages
13:     if {getMaxReturnAsset(newSol) < minReturn} then          ▷ fix solution if unfeasible
14:         newSol ← repairSol(newSol, inputs)
15:     end if
16:     if {newSol is in cache} then                                ▷ already optimized levels
17:         newSol ← loadFromCache(newSol)          ▷ use optimized levels saved in cache
18:     else          ▷ apply a local search based on quadratic programming optimization
19:         newSol ← QPOptimize(newSol, minReturn)   ▷ optimize levels f.e. asset in portf.
20:         newSol ← cleanSol(newSol)                ▷ delete from portf. assets with level = 0
21:         saveInCache(newSol)
22:     end if
23:     delta ← getRisk(newSol) - getRisk(baseSol)                ▷ newSol improves baseSol
24:     if {delta < 0} then
25:         credit ← -delta
26:         baseSol ← newSol
27:         if {getRisk(newSol) < getRisk(bestSol)} then          ▷ newSol improves bestSol
28:             bestSol ← newSol
29:         end if
30:     else{delta > 0 and delta ≤ credit}                              ▷ acceptance criterion
31:         credit ← 0
32:         baseSol ← newSol
33:     end if
34:     update elapsedTime
35: end while
36: return bestSol
    end procedure
```

under certain restrictive conditions, a new base solution even when it offers a slightly higher risk than the current base solution.

As regards as the *perturbation* stage (Pseudo-code **??**), this follows a destruction - reconstruction process. First, this process takes as an input the base solution. Second, the base solution is partially destroyed by deleting a randomly selected number of assets. Third, the destroyed solution is re-constructed (completed) by adding new assets to the portfolio.

Algorithm 14 Perturbation procedure to generate promising solutions.

```
    procedure perturbateSol(baseSol, inputs, beta)
 1: newSol ← copySol(baseSol)
                    ▷ 1. Remove a random number of randomly selected assets (destruction stage)
 2: nAssetsInSol ← getNAssetsInSol(newSol)
 3: if {nAssetsInSol > 1} then
 4:     nAssetsToRemove ← genRandomNumber(1, nAssetsInSol - 1)
 5:     for {i = 1 to nAssetsToRemove} do
 6:         asset ← selectRandomAsset(newSol)
 7:         newSol ← removeAsset(asset, newSol)
 8:     end for
 9: end if
                    ▷ 2. Randomly select one asset in current portf. to add several of its "friends"
10: asset ← getRandomAsset(newSol)
          ▷ 3. Use biased rand. to add friendly assets until reaching kMax (re-construction stage)
11: while {size(newSol) < getKMax(inputs)} do
12:     listOfFriendlyAssets ← getFriendlyList(asset)          ▷ Sorted list of friendly assets
13:     do          ▷ Randomly select a position using a Geometric(beta) prob. distribution
14:         position ← biasedRandom(size(listOfFriendlyAssets), beta)
15:         newAsset ← getAsset(listOfFriendlyAssets, position)
16:     while {newAsset in newSol}          ▷ Repeat until newAsset not in current portf.
17:         newSol ← addAsset(newAsset, newSol)
18:         asset ← newAsset
19: end while
20: return newSol
    end procedure
```

During this re-construction process, the selection of each new asset added to the portfolio is done following a 'friendship' criterion, i.e.: although the selection of the new asset is random, this new asset will be most likely selected among those assets that are highly compatible –i.e., showing a low covariance value– with the last asset added to the portfolio. This special behavior is attained throughout the use of a biased randomization selection process, which makes use of a geometric distribution of parameter *beta* ($0 < beta < 1$).

Finally, there might be times in which the newly generated solution does not fulfil the minimum return requirement. In those cases, a 'repair' stage is used to swap a randomly selected asset in the current portfolio by a high-return asset not currently in the portfolio (Pseudo-code **??**).

Algorithm 15 Repair procedure to make newly generated solutions feasible.

```
    procedure repairSolution(sol, inputs)
 1: unusedAssets ← getAssetsNotInSol(sol, inputs)          ▷ Consider assets not in portf.
 2: unusedAssets ← shuffle(unusedAssets)          ▷ Random sorting of the unused assets list
 3: assetA ← getRandomAsset(sol)          ▷ Select a random assetA in current portf.
 4: sol ← deleteAsset(assetA, sol)          ▷ Delete assetA from current portf.
 5: for {each asset assetB in unusedAssets} do          ▷ Search unused assetB with high return
 6:     if {getReturn(assetB) ≥ minReturn} then
 7:         sol ← addAsset(assetB, sol)          ▷ Add assetB to current port.
 8:         return sol
 9:     end if
10: end for
11: end procedure
```

8.3.2 Computational experiments

The ARPO algorithm has been implemented as a Java application. Two sets of stock market data are used to test it. The first set is retrieved from the repository ORlib: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html. These instances were proposed by Chang et al. (2000) and were studied by Schaerf (2002), Armañanzas and Lozano (2005), Moral-Escudero et al. (2006), Fernández and Gómez (2007), and Di Gaspero et al. (2011). The data set comprises constituents of five stock market indices, Hang Seng (Hong Kong), DAX 100 (Germany), FTSE 100 (United Kingdom), S&P 100 (United States) and NIKKEI 225 (Japan). These indices were extracted from DataStream and are measured at weekly frequency spanning the period from March 1992 to September 1997.

The portfolio frontier is divided into 100 equidistant points on the vertical axis that represents the user-defined rate of expected portfolio return. Although the algorithm has been designed for the constrained case, it is initially tested on the unconstrained mean-variance optimization problem. The test results show that it is able to return solutions that are overlapping with the UEF published at the OR Library, which contributes to validate the approach.

Next, the algorithm is executed on a constrained mean-variance frontier (the algorithm is executed 30 times and both the best and average results are recorded). The maximum time of execution for each instance is 20 seconds. The constraints are those imposed by the previous authors. The constraints involve the following conditions: $\varepsilon_i = 0.01$, $\delta_i = 1$, $k_{min} = 1$, $k_{max} = 10$, $\forall i \in \{1, 2, \ldots, N\}$. As in the aforementioned studies, pre-assignment constraints are not considered. Despite other authors claim that their approaches can solve the constrained problem with all the aforementioned constraints, this fact is not clearly showed, since the parameter values they use do not seem to impose a real challenge for their algorithms in terms of tight constraints.

8.3.3 Analysis of results

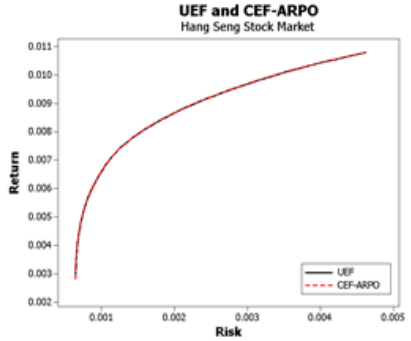
Table **??** shows the values of average percentage loss (APL) and associated computational times. In terms of the minimum APL, the ARPO algorithm outperforms on Instances 2 – 5 the hybrid solvers proposed by Di Gaspero et al. (2011), which comprise combinations of first descent and steepest descent with quadratic programming (FD+QP and SD+QP, respectively). In terms of computational time, ARPO shows a superior performance relative to that of the solver's SD+QP and is comparable or better than the solver's FP+QP performance. Although the minimum APL provided by ARPO is slightly superior to the hybrid solver combining a GA and QP in Moral-Escudero et al. (2006), on the remaining instances the minimum APL accomplished by ARPO is lower. Furthermore, the computational times are considerably lower than those reported by the TS in Schaerf (2002), and by GA+QP in Moral-Escudero et al. (2006).

The UEF (as provided in the ORlib) and CEF (as provided by ARPO) for the five stock market indices are compared in Panels A – E of Figure **??**. Panel A depicts the CEF for the Hang Seng stock market. A visual inspection suggests that the CEF is hardly distinguishable from the UEF. However, as the rate of expected return increases, the CEF tends to diverge relatively less from the UEF. In particular, at the higher end of the CEF that features rewarding but risky portfolios, the expected rate of return can be attained with fewer assets. Panel B depicts the CEF for the DAX 100 stock market. The CEF diverges from the UEF at the lower end of expected return. As the rate of expected return increases, the CEF becomes indistinguishable from the UEF. Panel C depicts the CEF for the FTSE 100 stock market. It indicates that –similarly to the DAX 100 stock index– the CEF departs from the UEF at the lower end of expected return. Panel D depicts the CEF for the S&P 500 stock market. It indicates that –as with the DAX 100 and FTSE 100 stock indices– the CEF departs from the UEF at the lower end of expected return. As the rate of expected return increases, the CEF becomes visually indistinguishable from the UEF. Finally, Panel E depicts the CEF for the NIKKEI stock market. It indicates that the relation between the CEF and the UEF follows a pattern similar to the Hang Seng stock index.

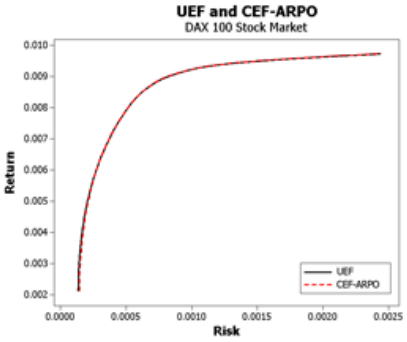
To evaluate differences between the UEF and the CEF-ARPO, the portfolio weights for FTSE 100 stock indices are provided (Figure **??**), where the required rate of return is 0.0041572635, which is an approximately central value within the overall of returns. The UEF considers all assets with weights ranging from 0 to 1 inclusively. The CEF is constrained to the minimum of 1 and the maximum of 10 assets, with portfolio weights ranging from 0.01 to 1. The minimum values of the

Table 8.4: Summary of results.

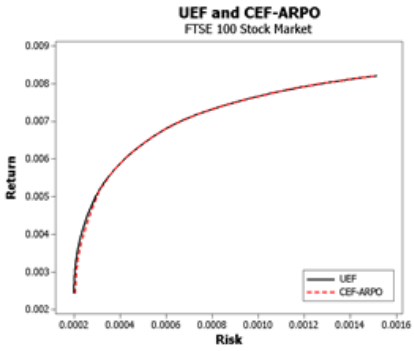
Instance	Di Gaspero et al. (2011)				Moral-Escudero et al. (2006)		Schaerf (2002)			ARPO
	FD + QP		SD + QP		GA + QP		TS		LS + QP	
	APL	T(s)	APL	T(s)	APL	T(s)	APL	T(s)	APL	T(Std) (s)
HS	0.00366	1.5	0.00321	3.1	0.00321	415.1	0.00409	251	0.00399	2.0 (0.87)
DAX 100	2.66104	9.6	2.53139	14.1	2.53180	552.7	2.53617	531	2.45403	1.0 (1.0)
FTSE 100	2.00146	10.1	1.92146	16.1	1.92150	886.3	1.92597	583	1.88340	13.7 (9.28)
S&P 100	4.77157	11.2	4.69371	18.8	4.69507	1163.7	4.69507	713	4.65095	15.5 (8.42)
NIKKEI 225	0.24176	25.3	0.20219	45.9	0.20198	1465.8	0.20198	1603	0.20189	5.0 (17.20)



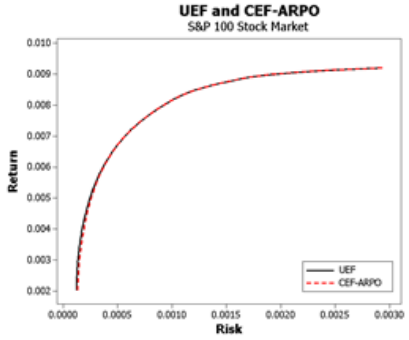
Panel A – Hang Seng Stock Market



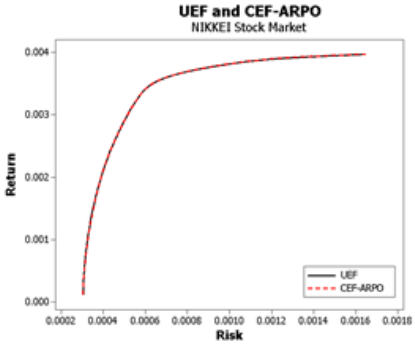
Panel B – DAX 100 Stock Market



Panel C – FTSE 100 Stock Market



Panel D – S&P 100 Stock Market



Panel E – NIKKEI Stock Market

FIGURE 8.4: UEF and CEF-ARPO.

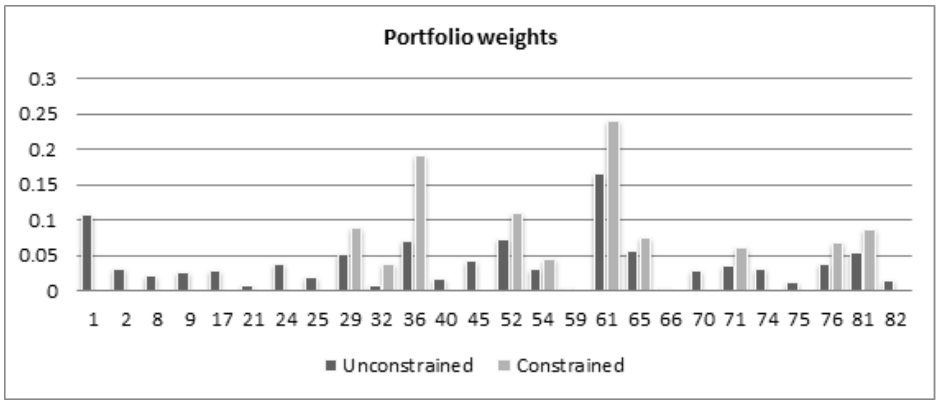


FIGURE 8.5: Portfolio weights for the FTSE 100 stock indices.

portfolio variance are 2.3872556507357437E-4 (the UEF) and 2.5098945345432527E-4 (CEF-ARPO). The percentage loss is 5.137%. The UEF selects 26 assets, whereas the CEF portfolio selects 10 assets, which are a subset of the assets selected in the UEF portfolio.

8.4 The SPOP

The difference between the POP and the SPOP lies in the modelling of asset returns and covariances. While they are represented by expected values in the first case, the second considers realistic stochastic uncertainty and thus treats these as random variables. This results in a modified return constraint where a return no lower than R must be achieved with a probability of, at least, P_0.

The mathematical formulation for the SPOP requires two modifications:

- Covariances (C_{ij}) in the objective function are considered to be random variables following a given probability distribution (e.g., the one that best fits the historical data available):

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j C_{ij} \geq R \tag{8.28}$$

- Equation **??** is replaced by the following probabilistic constraint:

$$P(\sum_{i=1}^{N} R_i w_i \geq R) \geq P_0 \tag{8.29}$$

where R_i refers to the asset return modeled as a random variable.

8.4.1 Methodology

The VNS metaheuristic is proposed as a base framework. The methodology includes biased randomization and employs the open-source quadratic programming solver ojAlgo (http://ojalgo.org) to determine the weights allocated to a given set of assets. Additionally, a cache memory is used in order to avoid calling the solver repeatedly for a specific set of assets.

The flowchart diagram of the approach is depicted in Figure **??** and described next:

1. Consider a SPOP instance defined by N assets. Each asset i has an associated return rate R_i, which is a random variable following a probability distribution. Each pair of assets i, j is characterised by a covariance C_{ij}, which is also random and depends on the correlation P_{ij} and the standard deviations S_i and S_j according to the following equation: $P_{ij} = \frac{C_{ij}}{S_i S_j}$

2. Transform the original stochastic problem into a POP instance by means of replacing the random variables by their expected values μ_i and σ_{ij}.

3. Construct an initial solution (*initSol*) by selecting the k_{min} assets with the highest returns, after including the s assets pre-selected by the investor, and calling the solver. Afterwards, simulation techniques are considered to compute the probability of satisfying the return constraint in the stochastic environment described by the original instance. In particular, a short number of scenarios (sim_{short}) is used to simulate returns. The solution is stored and one moves on to the fourth step, provided the constraint is satisfied. If this is not the case, a feasible solution is searched using a randomised and iterative procedure. First, the pre-selected assets compose a portfolio. In the next step, the non-preselected assets are ordered according to their expected return, and a random number, between $k_{min} - s$ and $k_{max} - s$, are selected using biased randomization, relying on a geometric distribution with a parameter β (Juan et al., 2011b). All weights are set to the minimum value initially, and then, each weight is set to the maximum value possible (taking into account for an asset a_i the following elements: ε_i, δ_i, and the fraction that remains to be allocated, i.e., $1 - \sum_{i=1}^{n} x_i$) in the order previously established. If an initial solution can be constructed through this, one moves to step 4. It is worthwhile to remark that the focus is on finding an initial feasible solution considering the stochastic environment and not the one with the lowest risk. The time spent searching for a feasible solution is limited by T_{init}, and the algorithm execution stops if no feasible solution is obtained.

4. A list *bestSols* is created for storing the l best found solutions in terms of expected risk. Then, *initSol* is copied into *currentSol* and k is set to one. Following this, the expected risk of *currentSol* is computed by using MCS, and the solution is included in the created *bestSols* list.

5. An iterative procedure is started and steps 6 and 7 are executed during a given amount of time (T_{loop}).

6. A new solution (*newSol*) is created by shaking the current one. This procedure consists of randomly erasing a number of non pre-selected assets in the solution and randomly introducing new assets until reaching k_{max}. The number of assets erased is determined by k. Moreover, a local search is applied to the resulting solution. It aims to improve the solution by replacing the asset with the lowest weight with another one from the list of non-selected assets.

7. *newSol* is compared against *currentSol*. If the former is better in terms of risk associated with the deterministic version of the problem, *newSol* is considered to be a promising portfolio setting and the return constraint for the stochastic environment is checked for it. In case of being satisfied, the expected risk is computed for the stochastic version of the problem. If the expected risk of *newSol* is better than that of *currentSol*, then *newSol* replaces *currentSol*, k is set to one and *bestSols* is updated. If it is not satisfied, the solution is discarded. If *newSol* is not better, k is increased in one unit if $k < K$ or set to one otherwise.

8. Once the iterative procedure ends, the algorithm returns *bestSols*. For each solution, a sample of risk measurements is obtained by simulating a large number of scenarios (sim_{large}). A risk analysis is performed where solutions are compared using the distributions of risk. In order to simplify the analysis, it is based on the expected values and the variances of the distributions, and the reliabilities (or probabilities of satisfying the return constraint). Accordingly, the Pareto dominant solutions (i.e., those that are not dominated by another portfolio for one measure while the other measures are at least equally good) are reported to the decision-maker.

8.4.2 Computational experiments

The algorithm has been implemented as a Java application. It is executed ten times using different seeds; only the best results are stored. Stock market data from the repository ORlib is used. This benchmark instance is deterministic. It has been adapted by replacing the deterministic returns and covariances by random variables. More specifically, the following complementary scenarios have been considered:

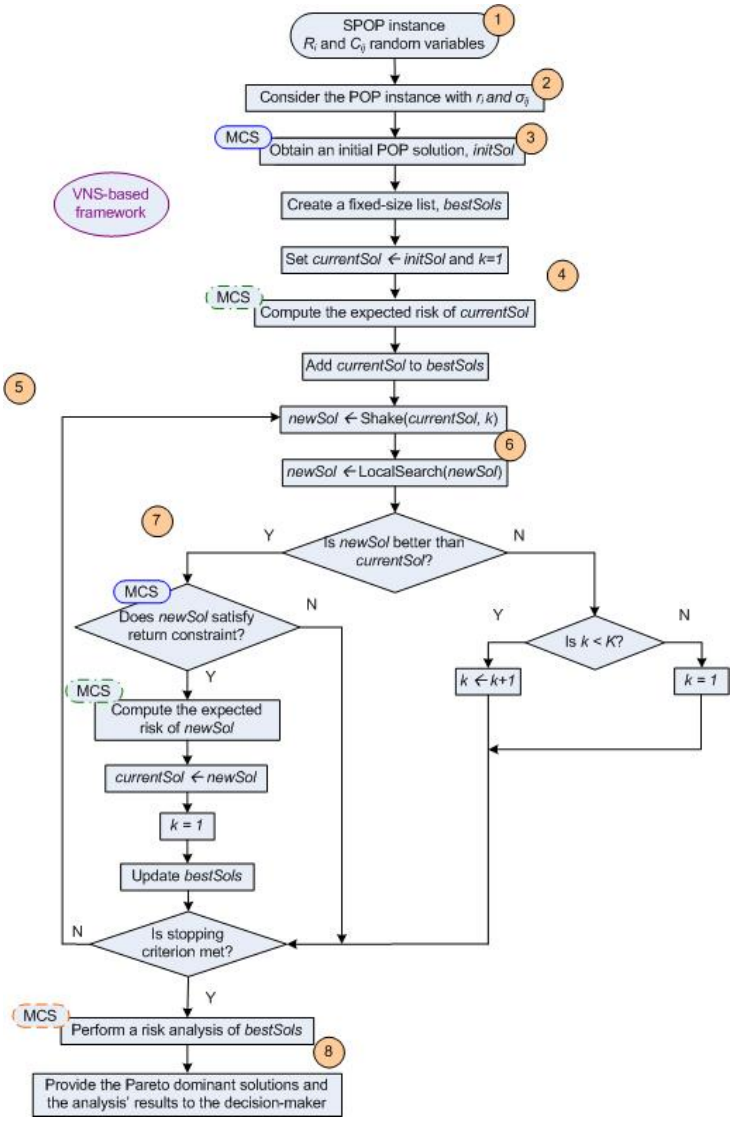


FIGURE 8.6: Flowchart of the proposed approach for the SPOP.

- S_i (Standard deviation) follows a $LN(\mu_S, \sigma_S)$, where LN represents a log-normal distribution, and μ_S and σ_S are the mean and the standard deviation of the variable natural logarithm, respectively. They may be determined by the value of the mean and the standard deviation of the variable that are set to σ_i and $c\sigma_i$, being c an input.

- P_{ij} (Correlation) follows a $TN(\mu_P, \sigma_P, l, u)$, referring TN to truncated normal distribution, where the parameters are the mean, the standard deviation, and the lower and upper limit, respectively. μ_P is set to the original correlation ρ_{ij}, while σ_P is an input. By the definition of correlation, l and u are set to -1 and 1, respectively. A special case is when $i = j$, then l and u are equal to 1 (i.e., $P_{ij} = 1$).

- R_i follows a $N(\mu_R, \sigma_R)$, where μ_R and σ_R are the mean and the standard deviation of the variable, respectively, which may be determined by the value of the mean and the standard deviation of the variable that are set to r_i and S_i, respectively.

Three values for c $(0.01, 0.025, 0.08)$ and σ_P($\sqrt{0.00002}$, $\sqrt{0.0002}$, $\sqrt{0.002}$) have been tested in order to explore different levels of stochasticity. The former values have been selected after performing some quick tests to explore the "reasonable" range for each parameter. Two computational experiments have been carried out. The first considers stochastic covariances (first two scenarios). The second builds on the first one, introducing stochastic returns (all three scenarios).

The parameter fine-tuning has been performed by doing fast experimental tests. The recommended number of neighbours (K) is 3 (Hansen et al., 2010). A movement in each neighbour involves changing 25%, 35%, and 45% of the assets, respectively. Regarding the number of solutions stored to analyse at the end (l), a total of 10 are considered. As suggested in Juan et al. (2011b), β is randomly selected from a uniform distribution with parameters 0.05 and 0.25. Finally, sim_{short} and sim_{large} are set to 2500 and 12500, respectively, and T_{init} and T_{loop} are set to 5 and 15, respectively.

8.4.3 Analysis of results

Table **??** summarises the results of the first experiment. The first experiment compares two types of solutions: (*i*) the best found solution to the deterministic version of the problem (BDS); and (*ii*) the best found solution to the stochastic version (BSS). For each variability levels (low, medium, and high) a different stochastic scenario is defined. Different risk measures (costs) associated with the BDS portfolio configuration are considered: the risk measure obtained when employing the BDS in a deterministic scenario, and the expected risk value obtained when using it in each stochastic scenario. The former could be considered as a lower bound for the BSS, while the latter could be considered as an upper bound. The first column reveals the required return. The next four columns depict the BDS. The following three columns contain the expected risk associated with the BSS for each of the stochastic environments. Also, the average computational time is provided. The last five columns gather some gaps: (*i*) the gap between the risk and the expected risk for a low level of stochasticity of the BDS; (*ii*) the gap between the risk of the BDS and the expected risk of the BSS (also for a low level of stochasticity); (*iii*) the gap between the expected risks for the BDS and the BSS considering the environment of a low risk, which quantifies the benefit of using the simheuristic approach instead of assuming constant values; and (*iv*) the previous gap considering the other two environments. Additionally, the average of each ratio has been added at the bottom of the table. Figure **??** illustrates boxplots of the gaps regarding expected risk between the BDSs and BSSs for the different environments. The expected risk of the BDS is subtracted from that of the BSS so that negative gaps indicate an improvement in the expected risk of the solution. Mean values are represented by diamonds.

Based on these outputs, it may be concluded that the algorithm is able to obtain a reasonably good BSS in 0.73 seconds on the average. The gaps between the risk of the BDS and the expected risk of the BDS (when used as a portfolio configuration for the stochastic environment) and the BSS are quite high even for the low-variability scenario (11.82% and 9.92% on the average). As expected, the measure of the BSS is closer to the lower-bound (the risk) than the one of the BDS. Regarding the benefits of using the simheuristic approach in comparison with assuming constant values in terms of expected risk, the mean gaps found for each environment are: -1.68%, -3.09%, and -7.10%. It is important to remark that the gaps are never positive. Thus, the BSS shows a lower expected portfolio variance than the BDS when the latter is used to solve the stochastic

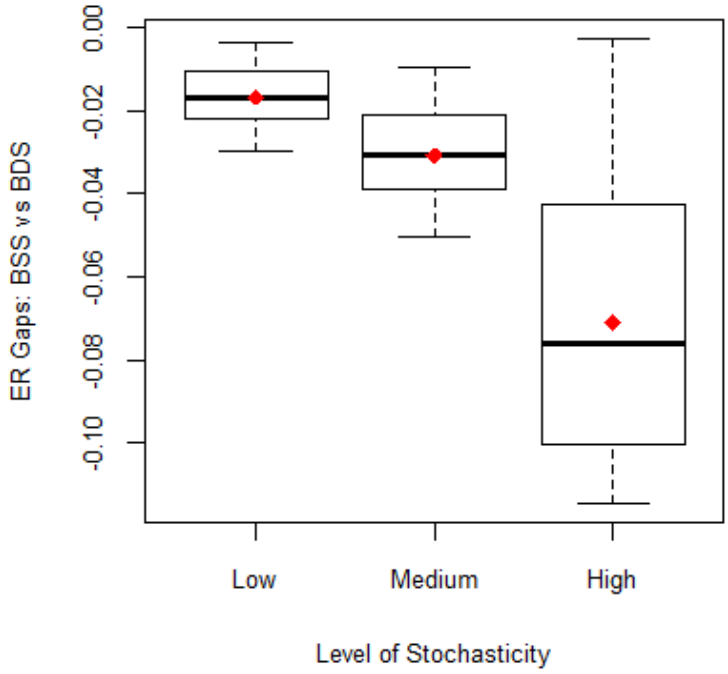


FIGURE 8.7: Risk gaps between the best deterministic and stochastic solutions for different levels of stochasticity (environments).

version of the problem. Furthermore, the performance of the BDS deteriorates when covariances become more uncertain.

Results from the second experiment are displayed in Table **??**. As in the previous table, the first column identifies the required return. Columns 2, 3, and 4 detail the expected risk of the BSSs under the lower, medium, and high levels of uncertainty, given a probability of 50% for attaining the required rate of return. Columns 5, 6, and 7 report the gaps between the expected values of risk of the BSSs under the lower, medium, and high levels of uncertainty, when the probabilities of attaining the required return are 50% and 47%, respectively (since the benchmarks used are extensions of classical ones for the deterministic version, only some probability values make sense). Finally, the average computational time is provided.

It can be concluded that the gap between the expected risk of the BSS requiring a probability of 47% and the one with a probability of 50% is relatively small, although it can be relevant in some cases. The average values for the different environments are: 0.31%, 0.39%, and 1.84%. Thus, the gap increases as the level of stochasticity gets higher.

8.5 Example with stocks and individual commodity futures contracts

Diversification is best achieved through combining assets with low or negative correlation into an investment portfolio. Due to the increased correlation among individual stocks, diversification possibilities of stock portfolios have become limited. Commodities in general and metals in particular on the contrary have shown to yield low correlation with stocks (Jaffe, 1989; Chua et al., 1990; Hillier et al., 2006; Daskalaki and Skiadopoulos, 2011), especially because macroeconomic shocks tend to impact stock and commodity prices in different directions (Silvennoinen and Thorp, 2013). Particularly concerning inflation, the reaction of commodities and stocks might differ fundamentally. Indeed, while unexpected inflation leads to an increase in the prices of commodities, stocks have generally been found to be an inflation-protected asset class (Hardouvelis, 1987; McQueen and Roley, 1993) or, in case of fluctuation, have even yielded falling prices (Fama, 1981; Amihud, 1996; Bansal et al., 2014). However, investment in physical commodities is characterized by high costs (storage) and additional uncertainty (perishable nature, seasonal cycles of the goods) so that commodity futures are a natural alternative, providing the same diversification

TABLE 8.5: Hang Seng Stock Market (Hong Kong) with stochastic covariances.

Deterministic Solution					Stochastic Solutions			Time (s.)	Gaps (%)				
Required Return	Risk (1)	E.R. - Low (2)	E.R. - Med. (3)	E. R. - High (4)	E.R. - Low (5)	E.R. - Med (6)	E.R. - High (7)		(2-1)	(5-1)	(5-2)	(6-3)	(7-4)
0.002861137	0.0006424	0.0007055	0.0007975	0.001114	0.0006948	0.000777	0.0010713	0.211	0.0982	0.0815	-0.0152	-0.0257	-0.0383
0.002941981	0.0006429	0.0007062	0.0008052	0.001195	0.0006952	0.0007772	0.0010701	0.409	0.0985	0.0813	-0.0156	-0.0348	-0.1045
0.003022827	0.0006437	0.0007015	0.0007888	0.0011057	0.0006961	0.0007782	0.0010706	0.627	0.0897	0.0814	-0.0076	-0.0135	-0.0318
0.003103671	0.0006444	0.0007108	0.0008089	0.0011647	0.0006976	0.0007799	0.0010727	0.456	0.1031	0.0826	-0.0186	-0.0358	-0.079
0.003184516	0.0006455	0.0007054	0.0007978	0.00113	0.0006996	0.0007824	0.0010764	0.369	0.0929	0.0838	-0.0083	-0.0192	-0.0474
0.003265361	0.0006468	0.0007057	0.0007951	0.0011208	0.0007011	0.000785	0.0010821	2.03	0.0912	0.084	-0.0065	-0.0127	-0.0346
0.003346206	0.0006483	0.0007113	0.0008062	0.0011714	0.0007021	0.0007857	0.0010806	2.461	0.0972	0.083	-0.0129	-0.0255	-0.0775
0.003427051	0.00065	0.0007143	0.0008095	0.0011649	0.0007035	0.0007869	0.0010809	1.541	0.0989	0.0824	-0.015	-0.0279	-0.0721
0.003507896	0.0006517	0.0007161	0.0008037	0.0010852	0.0007103	0.0007887	0.0010822	5.966	0.0989	0.09	-0.0081	-0.0187	-0.0028
0.00358874	0.0006536	0.0007103	0.000799	0.0011332	0.0007076	0.0007912	0.0010848	0.581	0.0867	0.0826	-0.0038	-0.0098	-0.0427
0.010137479	0.0035774	0.0040554	0.0047712	0.0074653	0.0039666	0.0045855	0.0067181	0.001	0.1336	0.1088	-0.0219	-0.0389	-0.1001
0.010218315	0.0036908	0.004178	0.0049221	0.0076114	0.0040965	0.0047415	0.0069622	0.002	0.132	0.1099	-0.0195	-0.0367	-0.0853
0.010299151	0.0038091	0.0043348	0.0051217	0.0079629	0.0042323	0.0049052	0.0072194	0.001	0.138	0.1111	-0.0236	-0.0423	-0.0934
0.010379986	0.0039324	0.004503	0.0053405	0.0083909	0.0043741	0.0050764	0.0074899	0.001	0.1451	0.1123	-0.0286	-0.0494	-0.1074
0.010460822	0.0040605	0.0046604	0.0055344	0.0087777	0.0045219	0.0052551	0.0077734	0.001	0.1477	0.1136	-0.0297	-0.0505	-0.1144
0.010541657	0.0041937	0.004762	0.0055848	0.0083847	0.0046757	0.0054415	0.0080701	0.001	0.1355	0.1149	-0.0181	-0.0257	-0.0375
0.010622493	0.0043317	0.0048879	0.005757	0.0089222	0.0048354	0.0056353	0.00838	0.001	0.1284	0.1163	-0.0107	-0.0211	-0.0608
0.010703329	0.0044747	0.0051485	0.0061357	0.0096893	0.0050011	0.0058368	0.008703	0.001	0.1506	0.1176	-0.0286	-0.0487	-0.1018
0.010784164	0.0046226	0.0052806	0.006288	0.010186	0.0051728	0.0060458	0.0090391	0.001	0.1423	0.119	-0.0204	-0.0385	-0.1126
0.010865	0.0047755	0.0055134	0.0065833	0.010458	0.005381	0.0063068	0.0096622	0	0.1545	0.1268	-0.024	-0.042	-0.0761
							Average	0.733	0.1182	0.0992	-0.0168	-0.0309	-0.071

TABLE 8.6: Hang Seng Stock Market (Hong Kong) with stochastic covariances and correlations.

Required Return	ER (50%)			ER Gaps [%] (50-47%)			Time (s)
	Low	Medium	High	Low	Medium	High	
0.002861137	0.0007006	0.0007872	0.0011358	0.00%	0.00%	4.38%	1.244
0.002941981	0.0007006	0.0007873	0.001135	0.00%	0.00%	4.44%	2.749
0.003022827	0.0007019	0.0007882	0.0011272	0.00%	0.00%	3.66%	2.647
0.003103671	0.0007027	0.00079	0.0011392	0.00%	0.00%	3.24%	1.7
0.003184516	0.0007068	0.0007925	0.0011075	0.33%	0.00%	1.23%	2.715
0.003265361	0.0007093	0.0007954	0.0011243	0.66%	0.00%	2.15%	1.438
0.003346206	0.0007085	0.000796	0.001137	0.09%	0.19%	2.67%	0.633
0.003427051	0.0007085	0.0007983	0.0011017	-0.11%	0.14%	0.00%	0.764
0.003507896	0.0007138	0.000799	0.0011144	0.54%	0.02%	0.05%	1.365
0.00358874	0.0007161	0.0008014	0.0011068	0.39%	0.00%	0.00%	1.726
0.010137479	0.0040004	0.0046595	0.0071471	0.00%	0.00%	1.05%	1.631
0.010218315	0.0041312	0.0048569	0.0074155	0.00%	0.82%	1.15%	2.8
0.010299151	0.004268	0.0049834	0.0076973	0.00%	0.00%	1.23%	1.421
0.010379986	0.0044359	0.0052031	0.0079926	0.57%	0.90%	1.31%	2.937
0.010460822	0.0045867	0.0053878	0.0083013	0.59%	0.93%	1.37%	4.313
0.010541657	0.0047435	0.0055803	0.0086235	0.61%	0.97%	1.42%	2.31
0.010622493	0.0049063	0.0057805	0.0089592	0.63%	1.00%	1.46%	3.76
0.010703329	0.0050752	0.0059884	0.0093083	0.65%	1.02%	1.49%	1.836
0.010784164	0.0052501	0.006204	0.0096709	0.67%	1.05%	1.52%	0.773
0.010865	0.0054126	0.0063772	0.0098998	0.37%	0.70%	3.06%	1.729
			Average	0.30%	0.39%	1.84%	

benefits without the implied disadvantages to an investor. Bansal et al. (2014) calculate the efficient frontier for an investment portfolio made up of indices of commodity futures and stocks and find it to lie above that for a traditional stock and bond portfolio. This diversification takes place independent of the state of the stock market: Crude oil futures contracts were shown to lead to successful diversification in both upward and downward trending stock markets (Geman and Kharoubi, 2008). Commodities emerge as a significant diversifier of both equity returns and volatility (Brooks and Prokopczuk, 2013). Investment in commodities is also demonstrated to significantly improve investor's expected utility. In this regard, Garrett and Taylor (2001) find that expected-utility-maximizing investors, depending on the degree of risk aversion, should invest 30 to 68% of their wealth in commodities. Unlike in Geman and Kharoubi (2008), this finding is event-dependent and period-specific. In contrast to the above mentioned studies that were conducted from the standpoint of a US-based investor, Belousova and Dorfleitner (2012) show that a euro investor can also accrue diversification benefits from commodity investments. In particular, the authors emphasize that industrial metals, agricultural commodities and livestock contribute to the reduction of investment risk, while precious metals and energy are associated with both lower portfolio risk and higher return. Investments in commodities become especially rewarding when the general financial climate becomes negative (Chow et al., 1999). In the quest for hedging and 'save haven' vehicles against losses in the sovereign bond market, Agyei-Ampomah et al. (2014) highlight the superiority of industrial metals (aluminium, copper, lead, nickel, tin and zinc) over precious metals (gold, silver, platinum and palladium). Antonakakis and Kizys (2015) underline the information contents of gold, silver and platinum in improving forecast accuracy of returns and volatilities of palladium, crude oil and the EUR/CHF and GBP/USD exchange rates. Prominent among commodities is gold – commonly regarded as a 'safe haven' asset – that provides wealth protection by hedging investments in the stock and foreign exchange markets, even during extreme price movements during periods of turmoil (Pukthuanthong and Roll, 2011; Ciner et al., 2013; Reboredo, 2013). The above results have previously been confirmed by You and Daigler (2012). However, they employ individual stocks and futures contracts rather than stocks and commodity indices. This increases the complexity of the optimization problem. In order to cope with this, they resort to a portfolio optimization software package, which is limited to 120 observations. To circumvent this, a metaheuristic algorithm is applied here that is not only capable of dealing with an extensive number of observations, but also with further constraints.

However, an extensive analysis on the diversification benefits of commodity futures by Cheung

and Miu (2010) raises concerns about the universal validity of the above findings, indicating that individual assessments become necessary. Furthermore, the ex-post performance of stock and commodity futures portfolios was found to be inferior to that of a portfolio made up of traditional assets by Daskalaki and Skiadopoulos (2011), thus making this another important question in the evaluation. It is thus also evaluated whether the applied methodology is able to identify stable asset weights based on ex-post performance and if so, whether the performance of the portfolio including commodity futures outperforms the traditional stock portfolio.

8.5.1 Problem and data description

There is a set of potential assets to choose from. On the one hand, there is a set of n stocks $S = s_1, s_2, ..., s_n$ and on the other hand, a set of m individual commodity futures contracts $F = f_1, f_2, ..., f_m$ is included, resulting in a total number of potential assets $A = a_1, a_2, ..., a_{m+n}$ equal to $m + n$. For all assets, the expected return based on historical data of a specified time period $E[R_i]$ is calculated. The inclusion of assets with negative expected returns is allowed for two reasons. The first is a technical one: As investors choose from a potential pool of assets whose returns are notably influenced by the historical time horizon chosen for analysis, it prevents introducing a bias. Furthermore, the introduction of futures contracts with slightly negative returns can still cause the portfolio to outperform that composed of only stocks. As a measure of riskiness of the portfolio, its variance is calculated. The methodology employed is the one described before for the POP.

As the approach is concerned with the comparison of a stocks-alone and a stocks-and-futures portfolio, individual daily historical closing price data for the Dow Jones 30 constituents on the one hand and daily settlement prices for the 21 most actively traded commodity futures prices in the United States covering the period from February 18, 2014 to April 1, 2016 are obtained, resulting in 535 observations for each time series. Due to expiration of fixed-maturity futures contracts, the continuous series are created by data providers by rolling over the futures contracts of different maturities. Table **??** presents the average daily returns and the corresponding standard deviations for each of the included stocks and commodity futures contracts. The average correlations within the two asset classes, as well the mean overall correlation, are presented in Table **??**. It becomes obvious that the correlation between the potential stocks is significantly higher than that within the class of commodity future contracts. Furthermore, the mean correlation between stocks and futures is the lowest overall for the data sample.

8.5.2 Analysis of results

In the following the results for two experiments are analyzed first with respect to risk analysis of the ex-ante portfolios and then with respect to the ex-post performance, or stability, of the found solutions. Ex-ante portfolios are those portfolios with constituent assets and weights determined by the matheuristic based on the data gathered for the sample period. Ex-post portfolios refer to the application of the ex-ante portfolio asset weights to the data following the sample period at time $t + 1$. It thus refers to a hypothetical investment at time t into the best found portfolios that is then evaluated one month later at time $t + 1$.

Figure **??** and **??** present two exemplary solutions. It is to be noted that assets 1 through 30 represent stocks and assets 31 through 51 represent commodity futures contracts. For low-level minimum returns in Figure **??**, the first stock portfolio contains portions of asset 7, 8, 9, 10, and 11 (all stocks), while the stocks and futures portfolio contains assets 11, 31, 39, 40, and 42 (one stock and four futures contracts). For high-level minimum returns, the solutions are much more similar with respect to the asset composition. The first exemplary stock solution in Figure **??** is composed of assets 14, 20, 21, and 30 (all stocks), while the stocks and futures portfolio contains asset 39 instead of asset 21. The exemplary solutions showcase that the solutions for higher minimum returns overlap further and are more similar in terms of selected assets constituents and weights than for low levels of return. This illustrates the finding that the allocation of commodity futures increases with increasing risk aversion of the investor, yielding that they represent a valuable alternative as diversification means especially for lower-risk portfolios.

Risk analysis

Table **??** summarizes the results of the two experiments. It compares the results obtained for a particular minimum return. At first sight, the previously mentioned complexity of the problem

TABLE 8.7: Descriptive statistics of stocks and futures.

Assets	Average return	Standard deviation
Stocks	0.000307%	0.012996
Apple	0.076993%	0.015544
Microsoft	0.084877%	0.015512
Exxon Mobil Corporation	-0.014890%	0.013191
Johnson & Johnson	0.035448%	0.009978
General Electric Company	0.047681%	0.012239
JPMorgan Chase & Co.	0.015270%	0.014037
The Procter & Gamble Company	0.013599%	0.009085
Verizon Communications Inc.	0.032659%	0.009721
Wal-Mart Stores Inc.	-0.010853%	0.011372
Pfizer Inc.	-0.004829%	0.011537
The Coca-Cola Company	0.038688%	0.009100
Chevron Corporation	-0.021950%	0.015983
Visa Inc.	0.069322%	0.014216
The Home Depot, Inc.	0.110242%	0.012426
The Walt Disney Company	0.050015%	0.012805
Merck & Co. Inc.	0.002151%	0.012748
International Business Machines Corporation	-0.026583%	0.012757
Intel Corporation	0.062516%	0.015464
Cisco Systems, Inc.	0.054523%	0.013921
UnitedHealth Group Incorporated	0.116388%	0.014094
McDonald's Corp.	0.058294%	0.010550
3M Company	0.050225%	0.010810
NIKE, Inc.	0.103030%	0.014542
The Boeing Company	0.005434%	0.014169
United Technologies Corporation	-0.017746%	0.011471
The Goldman Sachs Group, Inc.	0.005036%	0.013801
American Express Company	-0.061105%	0.013448
E. I. du Pont de Nemours and Company	0.009985%	0.015360
Caterpillar Inc.	-0.030999%	0.015346
The Travelers Companies, Inc.	0.067270%	0.009718
Commodity futures	-0.000496%	0.000338
Brentcrudeoil	-0.120960%	0.025457
Copper	-0.043944%	0.012876
Crudeoil	-0.125399%	0.025594
Cocoa	0.017888%	0.012117
Coffee	-0.055508%	0.023227
Corn	-0.031191%	0.014505
Cotton#2	-0.058913%	0.013930
Feedercattle	-0.032845%	0.010810
Gold	-0.004478%	0.009594
Heatingoil	-0.120557%	0.023106
KCWheat	-0.079898%	0.016955
Leanhog	-0.054406%	0.023260
Livecattle	-0.035734%	0.011952
Naturalgas	-0.116111%	0.022118
Orangejuice	-0.003081%	0.020640
Silver	-0.017541%	0.016391
Soybean	-0.047312%	0.015006
Soybeanmeal	-0.030699%	0.020249
Soybeanoil	-0.042044%	0.013330
Sugar#11	0.015679%	0.022557
Wheat	-0.054146%	0.017646

TABLE 8.8: Average correlations between asset classes.

Within stocks	Within futures	Stocks and futures	Overall
0.4385	0.0765	0.0075	0.1756

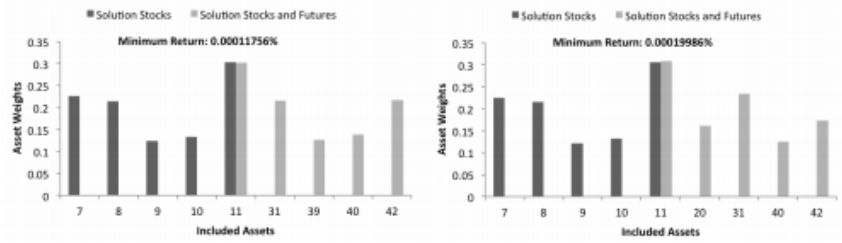


FIGURE 8.8: POP solutions for minimum returns on the lower spectrum.

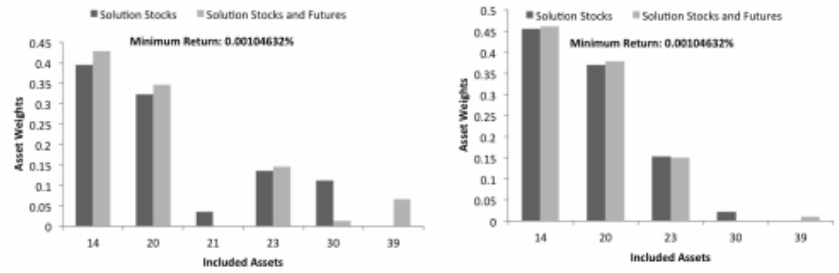


FIGURE 8.9: POP solutions for minimum returns on the higher spectrum.

becomes obvious when the instance times are considered: They significantly increase for the composite portfolios that are selected from an asset pool of 51 potential constituents as opposed to the basic formulation that only considers a pool of 30 stocks. More importantly, the associated risk is presented. As expected, it continuously increases with increasing returns demanded by the investor. The risks between two best found solutions based on different asset pools are then compared. A positive risk gap indicates that the risk was minimized with respect to the solution found for a stock-alone portfolio and thus successfully diversified. This was the case for 94 out of the 99 return instances, while the remainder showed a gap equal to zero.

Generally, the gap decreased with increasing minimum returns. Two conclusions may be drawn. On the one hand, there is a threshold return, beyond which additional returns require a more significant increase in associated risk because this return is generally only achieved by fewer assets, reducing diversification benefits. For this set of data, it is found at 0.00114%. From this threshold on, the minimum required return could solely be achieved by certain assets, thus reducing the pool of potential assets and leading to portfolios of fewer included assets than the maximum number of five dictated by the cardinality constraint. This leads to portfolios composed of only stocks and thus also to a zero gap. On the other hand, it becomes obvious that the gap is much larger for low-return portfolios, from which one can conclude that risk-averse investors profit to a larger extent from futures diversification.

Ex-post stability analysis

Concerning the stability of the resulting portfolios, an ex-post application of the portfolio weights has been conducted. Two of these are exemplarily presented below; the first corresponds to the lowest daily minimum return and the second corresponds to the highest minimum return level at which the portfolios still differed. A one-month ahead analysis is considered and the resulting returns of the portfolio are compared against the corresponding minimum return on a monthly basis. The ex-post analysis consists of the hypothetical investment into the best found solution at the corresponding asset weights at the end of the sample period. Then, after a holding period of one month, the returns on the investment are calculated based on the actual price movements of the constituent assets.

Table **??** presents the metrics of the ex-post analysis for the first of two best found portfolios presented in Figure **??** and **??**, respectively. Solution 1 was found for an extremely risk-averse investor, while solution 2 represents a risk-loving investor's investment recommendation. The actual return is presented below the minimum return for both the stocks and the combined stocks and futures portfolio. Exemplified by the two solutions, the ex-post performance differs greatly depending on the required minimum return and the asset pool.

TABLE 8.9: POP results for a selected subset of minimum returns.

	Stock-alone portfolio		Stock-and-futures portfolio		Gap	Gap [%]
Minimum return	**Risk (1)**	**Time [s]**	**Risk (2)**	**Time [s]**	**(1) - (2)**	**(1) - (2)**
0.0000117564	0.0000531088	0.873	0.0000218480	10.202	0.0000312608	58.86180821%
0.0000822945	0.0000531088	0.042	0.0000230232	0.567	0.0000300856	56.64899226%
0.0001528327	0.0000531088	0.031	0.0000238688	3.141	0.0000292400	55.05678908%
0.0002233709	0.0000533257	0.183	0.0000265600	0.279	0.0000267657	50.19287135%
0.0002939091	0.0000533474	0.058	0.0000271226	12.655	0.0000262247	49.15853444%
0.0003644473	0.0000540647	0.014	0.0000311877	2.541	0.0000228770	42.31411623%
0.0004349855	0.0000546027	0.287	0.0000343818	16.648	0.0000202209	37.03278409%
0.0005055236	0.0000558261	0.225	0.0000375692	8.41	0.0000182569	32.70316214%
0.0005760618	0.0000573681	0.099	0.0000428111	19.99	0.0000145571	25.37472916%
0.0006466000	0.0000601832	1.1	0.0000473472	11.772	0.0000128360	21.32821120%
0.0007171382	0.0000642364	0.254	0.0000543535	17.792	0.0000098829	15.38520216%
0.0007876764	0.0000694180	0.4	0.0000619521	6.255	0.0000074659	10.75499150%
0.0008582145	0.0000766848	3.21	0.0000705611	16.555	0.0000061238	7.98554603%
0.0009287527	0.0000855374	0.091	0.0000812718	13.253	0.0000042656	4.98682448%
0.0009992909	0.0000965467	1.369	0.0000936730	5.567	0.0000028737	2.97648703%
0.0010698291	0.0001099596	1.089	0.0001090107	1.553	0.0000009488	0.86295330%
0.0011403673	0.0001373443	0.001	0.0001373443	0.001	0.0000000000	0.00000000%

TABLE 8.10: Ex-post performance of two exemplary solutions.

	Solution 1 – Low return	Solution 2 – High return
Required daily return	0.0000118%	0.0011051%
Actual return stocks portfolio	-0.0009301%	0.0002369%
Actual return stocks and futures portfolio	0.0051194%	0.0003352%

Figure **??** presents the risk-return characteristics of all ex-post portfolios. At first sight, it can be constated that, solely taking into consideration the positive portion of the return axis, the two curves resemble the shape of a Markowitz efficient frontier curve. It further becomes obvious that stock-and-futures portfolios overall achieved positive average ex-post daily returns, while a large portion of stock-alone portfolios presents the potential investor with negative returns. Because these negative returns are the result of downside risk exposure of the accompanying portfolios with high variance, it is intuitive that this part of the plot does not possess a positive slope. Concluding, it becomes evident that adding futures to the portfolios significantly improved the portfolio's behavior with respect to traditional financial theory in that increased returns can be achieved by assuming a more risk-exposed investment position. Moreover, the superiority of the stock and futures portfolios in ex-post performance is reinforced when considering both investment dimensions, risk and returns. Figure **??** shows that the ex-post portfolios of stocks and commodity futures can be a more effective vehicle of diversification than the portfolios of stocks only. Indeed, the ex-post portfolio variance is smaller for the former than for the latter, as shown by the minimum variance portfolio. Moreover, for a given value of the portfolio variance, average returns are larger for the portfolio combining both stocks and commodity futures. Furthermore, including commodity futures caters not only to risk-averse but also to risk-taking investors, since a broader range of values for both portfolio variance and return can be obtained.

8.6 Conclusions

Finance constitutes a highly dynamic and stochastic field playing an essential role in economy and social welfare. In this context, decision-makers frequently face COPs such as the POP, in which an investor aims to select a few risky assets, and decide the proportion of the budget to invest in each one in order to achieve a minimum return while minimizing a portfolio's risk measure. A richer version of the POP is defined by a number of additional constraints such as: cardinality, quantity and pre-selection constraints. This problem is usually tackled by using expected values for returns and covariances. A more realistic scenario is covered by the SPOP, where the aforementioned inputs are modelled as random variables. In addition to provide a review on metaheuristics applied to rich portfolio optimization and risk management, this chapter has presented solving methodologies based on metaheuristics and simulation. Aiming to facilitate the maximum diversification,

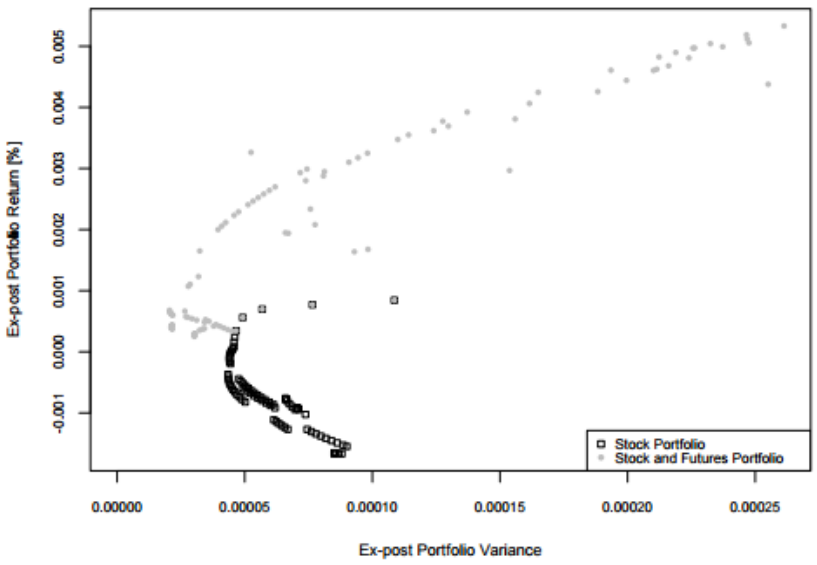


FIGURE 8.10: Frontiers of ex-ante optimal portfolios in ex-post analysis.

a study is performed to quantify the benefit of introducing commodity futures to a portfolio of stocks.

The main conclusions are:

- The number of related publications has been increasing during the last decade, especially in the case of POPs. Population-based metaheuristics, and in particular GA and PSO, have been the predominant solving methodologies. Regarding single-solution metaheuristics, TS and SA have been extensively applied too. There is not a 'single winner' approach, meaning that different metaheuristic implementations have provided results of comparable quality to different problems.
- There is a clear trend in promoting the development of hybrid algorithms, either by combining different metaheuristics or by combining metaheuristics with statistical or machine learning techniques. However, there is a lack of works considering stochastic versions of the optimization problems.
- Most POPs include some kind of risk management and, in the other direction, most RMPs considering optimization issues can be modelled as enriched variants of POPs.
- The methodologies presented are able to solve real-sized instances in small amounts of time.
- Even in an environment with a relatively low level of variability, a stochasticity-aware approach may provide much better results than a metaheuristic approach generating solutions for the (deterministic) POP.
- Futures contracts provide successful investment diversification. Particularly, risk-averse investors can drastically reduce their expected risk exposure by diversifying into stock-and-futures portfolios. Likewise, these portfolios of risk-averse investors yield more stable actual returns in the short term.

Chapter 9

Applications in computing

This chapter studies two important issues related to metaheuristics: the parameter fine-tuning and parallel computing. It presents a classification of works on parameter fine-tuning, and proposes a simple, general and automated methodology. In addition, a set of computational experiments on different COPs are carried out in order to analyze the effect of the number of agents and time on the performance of classical heuristics.
It is based on the following journal article: Calvet et al. (2016b).
This work has been presented at the following conferences: Calvet et al. (2015c) and Ruiz et al. (2015).

9.1 Introduction

Although the performance of metaheuristics is known to depend on its parameter values, the scientific community has not formally addressed the parameter setting problem (PSP) until the end of the last century. According to Eiben et al. (1999), during the first decades of metaheuristics research, many scientists based their choices on tuning the parameters "by hand", i.e., experimenting with different values and selecting the ones that provide the best outputs, or "by analogy", applying settings that have been proven successful for similar problems. More recently, the need for a systematic approach towards setting of metaheuristic parameters has been increasingly outlined in the literature (Hooker, 1995; Johnson, 2002). Subsequently, researchers employ a scientific approach to tackle the PSP more frequently. It is important to highlight that the selection of a systematic methodology leads to a gain of efficiency, as in general, less time is required to fine-tune the parameters while the performance of the metaheuristic is the same if not improved. However, there is no methodology commonly accepted by the scientific community and there is also a lack of publications that compare, in an exhaustive and objective manner, the main approaches and the techniques used so far. Moreover, some of the proposed methodologies are not easily reproducible or are highly metaheuristic and problem dependent. These are some of the reasons why, in spite of the amount of parameter fine-tuning works, many practitioners go on tuning by hand or designing algorithms without parameters (or with a very low number of them), even in the case when more parameterized algorithms could lead to better performances. This chapter aims to contribute to the literature by proposing a general and automated statistical learning based procedure to tackle the PSP. It is accompanied by some methodological guidelines to validate the results. In order to test the methodology and illustrate its application, the approach is employed to fine-tune a hybrid algorithm implemented to solve the MDVRP.

The use of distributed and parallel computing systems (DPCS), which allows the aggregation of multiple autonomous computing resources interacting to achieve a common goal (Coulouris et al., 2005), may also have a significant effect on the performance of metaheuristics. This chapter describes and tests an efficient, flexible, and browser-based framework to facilitate access to computational resources (Berry, 2009) and, ultimately, solve COPs in 'real time' (a few seconds). This framework enables the employment of new versions of web browsers (such as Google Chrome, Firefox, and Internet Explorer) as nodes in a cluster. The only required step is to visit a website. The embedded JavaScript code into this website enables the communication with the job dispatcher service. It may be considered a more scalable paradigm than traditional grid computing, since the connection of people is boosted by the fact that no third party software installation is required. Due to the relevance of COPs for SMEs and some academic works proposing the

implementation of DPCS for addressing them (Talbi, 2006; Talbi, 2009), the working and the potential benefits of the proposed approach are illustrated here by solving the CVRP and the PFSP using different numbers of nodes running a simple metaheuristics with a given seed and limit of computing time.

9.2 Parameter fine-tuning

Ries et al. (2012) define the PSP as the search for a set of parameter values θ^* in the parameter space Θ such that $\forall\theta \in \Theta : \theta^* \succeq \theta$ (where $\succeq$ denotes a relation of preference), for a given metaheuristic m in the metaheuristic space M, and a given instance x or group of them X in the instance space I. In practice, the amount of time available for experimenting T may be a restriction. In this case, the solution is approximate ($\hat{\theta}$). With regards to the difficulty of this problem, Montero et al. (2014) state that: (*i*) it is time consuming; (*ii*) the best set of parameter values depends on the problem at hand; and (*iii*) the parameters can be interrelated.

During the last decades, a large number of methodologies have been put forward to solve it. These proposals can be classified in two groups (Birattari and Kacprzyk, 2009): parameter control strategies (PCS), and parameter tuning strategies (PTS). This classification is extended by instance-specific parameter tuning strategies (IPTS), which includes features of the aforementioned groups.

9.2.1 Literature review

This section provides a brief description of each approach and some of the most cited works. The interested reader is refered to more specific publications such as Eiben et al. (1999), De Jong (2007) and Battiti and Brunato (2010) for an expanded review of PCS, Birattari and Kacprzyk (2009) in the case of PTS, and Ries (2009) for IPTS.

Parameter control strategies

These methodologies aim for a dynamic fine-tuning of the parameters by controlling and adapting their values while solving a problem instance. They follow two basic steps: firstly, an initial set of parameter values is chosen; secondly, an adaptation mechanism is integrated which changes relevant parameter values. Most of these strategies apply adaptive parameter control, which means that their adaptation mechanism is based on the assessment of particular information that is stored during the iterative process of a metaheuristic. This information is usually related to the goodness of intermediate solutions. The main drawbacks of this approach are the potentially high computational effort required and the lack of acquired understanding about good parameter values each time an instance is solved.

Eiben et al. (1999) address the PSP in EAs. Three categories were defined to classify the PCS. The first one, deterministic parameter control, alters the value of a parameter by some deterministic rule, which is usually time based. The second category, adaptive parameter control, does employ feedback to determine the direction and/or magnitude of a parameter change. This is the most used kind of control. The third, self-adaptive parameter control (Smith, 2008), encodes the parameters to be adapted into the chromosomes of an EA. De Jong (2007) describes the main motivations to use dynamic parameter setting strategies in EAs: first, as the running proceeds, information about the fitness landscape is generated, which may be used to improve the performance; also, changing the parameters is needed as an EA "evolves from a more diffuse global search process to a more focused converging local search process". Table **??** gathers a few representative works following this approach.

Parameter tuning strategies

This approach relies on the concept of robustness (Viana et al., 2005). A robust algorithm provides good results for a given set of instances of a problem using a fixed set of parameter values. The basic procedure involves finding a set of parameter values providing satisfactory results for a set of instances, usually using statistical and/or optimization techniques. Some authors analyse only a representative subset of instances and apply the set of parameter values found to solve all the instances. This approach also includes the case of solving one instance. Table **??** shows

TABLE 9.1: Representative works employing PCS.

Work	Main techniques	Metaheuristic	Optimization problem
Battiti and Tecchiolli (1994) and Battiti and Brunato (2005)	Reactive scheme	TS	QAP and maximum clique problem
Zennaki and Ech-Cherif (2010)	SVMs	TS	TSP
Lessmann et al. (2011)	Regression models	PSO	Water supply network planning problem

some works relying on this approach. Many authors focus on minimizing the number of runs, presenting simple models without interactions (e.g., Coy et al., 2001; Pongcharoen et al., 2007; Xu et al., 1998). DOE and regression analysis are the most employed techniques.

TABLE 9.2: Representative works implementing PTS.

Work	Main techniques	Metaheuristic	Optimization problem
Xu et al. (1998)	Tree growing and pruning method based on statistical tests	TS	Steiner tree-star problem
Bartz-Beielstein et al. (2004)	DOE, classification and regression trees, and design and analysis of computer experiments	PSO and Nelder-Mead simplex algorithm	Elevator group controller problem
Birattari and Kacprzyk (2009) and Birattari et al. (2010)	Racing algorithm (Maron and Moore, 1993) and the Friedman's two-way analysis of variance by ranks (Conover, 1999)	ILS and ACO	QAP and TSP
Adenso-Diaz and Laguna (2006)	DOE and local search	Neighbourhood structure, TS, SA, TS, heuristic based on the SA and the TS, and TS	Steiner problem, part-machine grouping problem, part-machine grouping problem, single-machine scheduling, proportionate flowshops, and bandwidth packing
Pongcharoen et al. (2007)	DOE	GA	TSP
Ridge and Kudenko (2007)	DOE and desirability functions	ACO	TSP
Gunawan et al. (2013)	DOE, response surface methodology and ParamILS (Hutter et al., 2009)	SA	Industry spares inventory optimization problem

Instance-specific parameter tuning strategies

As in the case of PCS, IPTS aim for an instance-specific tailoring of the parameters. At the same time, these strategies use a fixed set of parameter values, as the PTS, avoiding the need of modifying the metaheuristic algorithm and reducing the potential computing effort required to adapt parameter values during the algorithmic run. In order to implement these strategies the relation between the parameter values and the performance of the metaheuristic has to be analysed, taking into account instance features. The next step consists in developing a mechanism able to use the features of a new instance to recommend a set of parameter values. The key element is the selection of instance features easy and fast to compute, and good at discriminating instances on the shape of their fitness landscapes, which explore the relationship between the objective function values and the parameters. This learning may take a non-negligible amount of time, but it is assumed that this approach requires less computing time than the PCS approach does. Some contributions are included in Table **??**. The number of works is low since it is relatively new.

Approaches comparison

All approaches have different advantages. The dynamic adaptation of the parameter values that characterizes PCS usually provides better results. However, the computing effort tends to be

TABLE 9.3: Representative works implementing IPTS.

Work	Main techniques	Metaheuristic	Optimization problem
Ries (2009)	DOE and fuzzy logic	Guided local search and GA	TSP
Pavón et al. (2009)	CBR and Bayesian networks	GA	Root identification problem
Dobslaw (2010)	DOE and NNs	PSO	TSP

higher. On the other hand, the PTS approach is the easiest and fastest to use, once a set of parameter values is selected. Although the code of the algorithm is not changed, finding an adequate set may be also time-consuming. The last group of strategies represents a compromise solution: it takes less computing time than the PCS approach, but requires implementing a learning mechanism, for which statistical learning skills are needed. Therefore, there is no approach that stands out from the others. Probably, the most adequate depends on the specific problem to tackle, the instances to solve, the available time and the skills of the researcher. Despite this fact, some general guidelines can be formulated. PTS can be considered as the best option when working with robust algorithms. Regarding IPTS, they are more complex than PTS but provide better results when the algorithm is not robust. In case of prioritizing the algorithm performance, PCS usually constitute the most recommendable approach.

9.2.2 Methodology

A methodology that follows the PTS approach is proposed. As described before, this approach is not computationally intensive, and the inference from a representative sample of benchmark instances to the whole set usually provides good results, specifically if the analysed algorithm is robust. Another reason for focusing on PTS is that there is no methodology based on this approach and widely employed, but at the same time, there are plenty of techniques that can be used. The methodology is based on clustering and DOE. The remainder of this section presents a statistical learning based methodology to obtain a list of sets of parameter values, and a more global procedure to validate and assess its goodness.

General methodology

It is assumed that the experimenter has described a problem and chosen the metaheuristic to tackle it.

- The first step involves choosing a subset of the instances. Their fitness landscapes will be analysed in order to obtain sets of parameter values that provide good results for them. The subset has to be representative as these sets of parameter values will be used to solve the whole set of instances. An approach to select a representative subset is, firstly, to determine the instance features that have a major influence on which set of parameter values is the most adequate, and then, choose the instances in such a way that the feature values of the subset are representative of those of the entire set of instances. For example, if there is a parameter for which its optimum value is known to depend on the instance size, a representative subset of the instances will present the same proportion of instances of a given size that the whole set does. A possible simplification for feature selection consists of choosing those that are commonly used to discriminate instances of a specific problem. For instance, Ries et al. (2012) study the size, the distance metric, a ratio to describe the shape of the area within which a set of cities is distributed and a measure of clustering for the TSP.

  In contrast, a problem-independent approach is proposed here. Initially, for a given number of randomly generated sets of parameter values, each instance is solved several times using different seeds for the random number generator of the algorithm. Alternatively, the sets could also be generated using more advanced statistical techniques such as DOE. The medians of the objective function values found with the same parameter values but different seeds are considered. It is essential to remark the importance that a seed may have in the performance of an algorithm (Juan et al., 2015c; Czarn et al., 2004). Afterwards, feature scaling is applied to the values obtained for each instance. Then, this data is used to cluster

instances and select a representative one from each cluster. These instances form the subset to analyse.

For each instance of the subset, the steps ranging from the second to the fourth are implemented as follows.

- The second step requires selecting the range over which each parameter can be set. Some experience or knowledge about the problem and the metaheuristic may be highly valuable. The ranges should be large enough to cover at least one set of parameter values that can provide a sufficiently good solution with a high probability. On the other hand, a smaller range would allow the experimenter to describe more accurately, with the same resources, the relationship between the parameter values and the objective function value. If there is no a priori information about which are the best regions of the parameter space, a suitable procedure is to perform a rough and fast landscape analysis.
- The third step consists of designing an experiment. A central composite design is studied. Each parameter is considered a factor and the extreme values of its range define the levels. According to this design, the algorithm is executed also several times for each combination of factor values, each one with a different seed.
- In the fourth step, a procedure is developed to search the neighbourhood of the best set of parameter values found. Specifically, another central composite design centred on this set is applied.

Finally, the upshot is a list of recommended sets of parameter values, one per cluster; in particular, those that reported the best results on the last step. The procedure is shown in Figure **??**. An extended proceeding (Figure **??**) is described below in order to validate the list of sets of parameter values obtained and analyse the results provided by it.

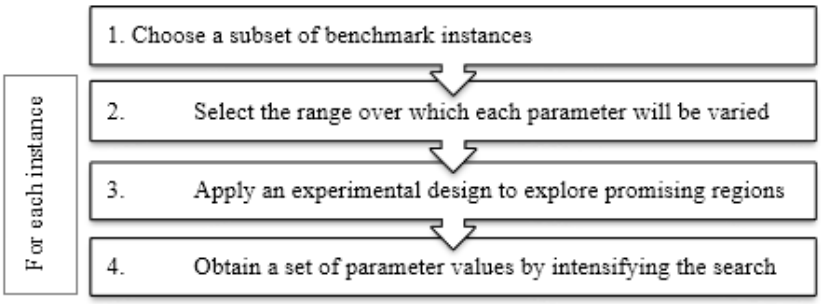


FIGURE 9.1: Outline of the procedure for parameter fine-tuning.

Before all else, a list of sets of parameter values, $\hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K)$ where K is the number of clusters, is chosen as has been explained in the precedent subsection. Later on, each instance of the subset used to select $\hat{\theta}$ is solved with the corresponding set of $\hat{\theta}$, and with different sets, $\bar{\theta}_j$ ($j = 1, 2, \ldots, J$) (equally spaced, randomly selected or relatively close to the set of $\hat{\theta}$ according to some distance measure). To assess the performance of a set of $\hat{\theta}$ in a specific instance regarding the other sets, the associated solutions are compared. Given a decision level parameter r ($1 \leq r \leq J + 1$), if the rank of the objective function value provided by the proposed set is equal or lower than r, then it is considered a good set for that instance. Once all the instances of the subset are examined, it can be reckoned the proportion of them in which the corresponding set has been classified as good. $\hat{\theta}$ is validated by comparing this proportion with a predefined parameter p ($0 < p < 1$); if the proportion is higher, then the experimenter has enough evidence of the quality of $\hat{\theta}$ to go on to test it with other instances in the next step.

If $\hat{\theta}$ is not validated, the process has to be readjusted and restarted. This readjustment may be done in several ways, some options are: checking the robustness and the adequacy of the clustering, adapting the ranges, dedicating more resources to the search, etc. The best strategy is problem-dependent. As a consequence, the choice should rely on the opinion of the experimenter, who will have acquired valuable information from the outputs observed.

Once the list of sets of parameter values has been labelled as valid, it is applied for solving the other instances (each one with the set proposed for the representative instance of the cluster where

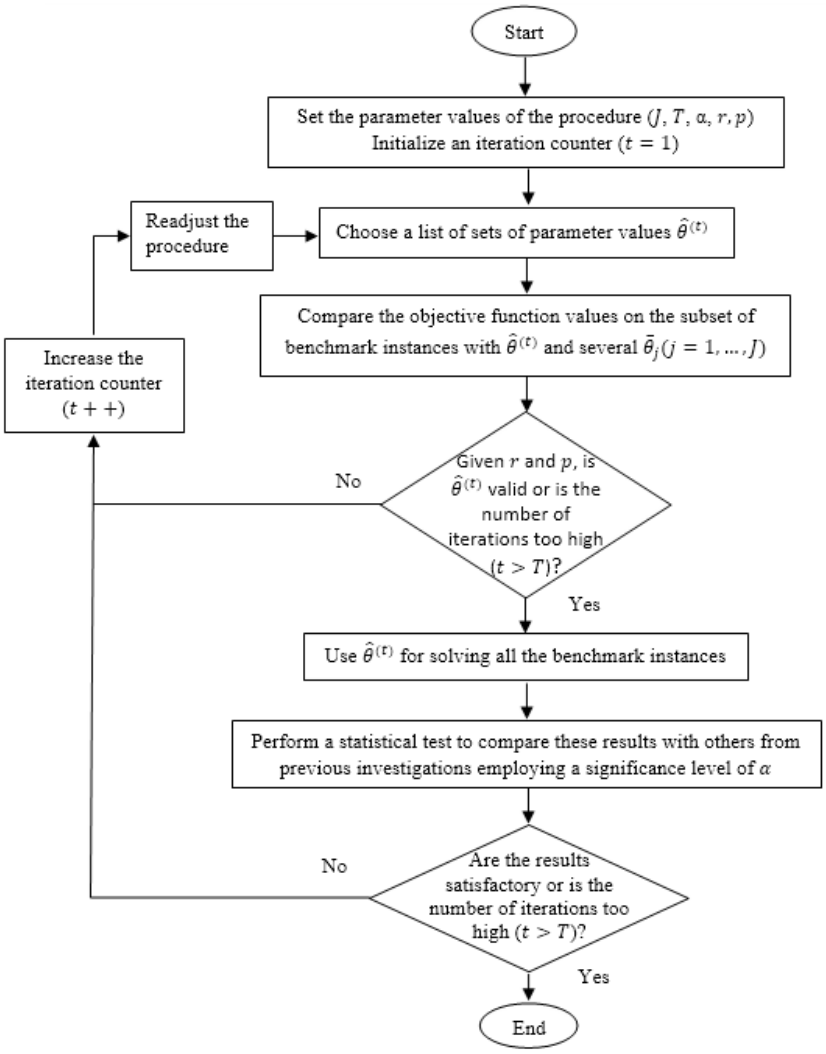


FIGURE 9.2: Flowchart representing the proposed methodology.

it has been assigned). To examine the effectiveness of the procedure, it is desirable to compare the solutions with others reported in the literature for the same instances, by performing the t-test for paired samples if data is normal, or the Wilcoxon signed rank test otherwise. If the means (or the mean ranks if data is not normal) do not differ significantly, it may be classified as a satisfactory outcome as it will mean that the proposed methodology, automated and general, has been proven to be competitive. If the results are unsatisfactory, the procedure should be modified and reinitiated.

It is useful to consider that, since the available resources are usually limited, the possible readjustments should be also limited (T represents this limit). Consequently, the process may end without a satisfactory list of sets of parameter values. In this case, the list which provides on average the best solutions will be accepted.

9.2.3 Computational experiments

The methodology has been implemented to fine-tune the parameters of the hybrid algorithm described in Juan et al. (2015c), which combines biased randomization and the ILS metaheuristic to address the MDVRP. This algorithm has three main parameters: bM, bR and $p*$, which take values between 0 and 1.

The first step is the selection of a representative subset of instances. Initially, 10 randomly generated sets of parameter values, 7 seeds and the 33 benchmark instances solved in the aforementioned paper were selected. Therefore, information from 2310 runs was stored. Data from different seeds was aggregated by computing the median; then feature scaling was applied. The instances that were considered easy-to-solve, those that presented no variation in the results, were

TABLE 9.4: Clustering of the benchmark instances.

Medoids	Clusters
p01	p01
p07	p04, p07, p11, p18, pr02, pr05, pr09
p09	p03, p09, pr04, pr10
p17	p17
p19	p19
p22	p22
p23	p20, p23
pr06	p05, p06, p08, p10, p15, pr01, pr03, pr06, pr07, pr08

separated. Afterwards, a clustering using the k-medoids algorithm (Theodoridis and Koutroumbas, 2009) was performed. The range of values considered for setting the value of k was 2-12. The final value was selected employing the average silhouette criteria (Rousseeuw, 1987). The composition of the clusters and the representative instances can be observed in Table **??**.

Once the subset of instances was formed, the second step, setting the ranges of the parameters, was carried out. After a statistical analysis, it was concluded that just two parameters, bM and bR, did significantly affect the performance of the algorithm. Therefore, only those two parameters were studied. Five equally spaced values ranging from 0 to 1 were analysed for each parameter. Each instance was solved seven times (considering different seeds) for each possible combination of parameter values. The objective function values were aggregated as before. Then, the values for other possible combinations were estimated by linear interpolation.

The ranges were set to cover the smallest rectangular area of the parameter space that included the lowest objective function values. In particular, the values labelled as the lowest were those meeting the following condition:

$$\textit{Objective solution} \leq \textit{minimum value} + \beta \cdot (\textit{maximum value} - \textit{minimum value})$$

The value of β was set at a different value for each instance. More precisely, it was the minimum value that encompassed, at least, 5% of the search space. Figure **??** shows the contour plot and the area in which the search was intensified for each instance.

The next step was applying a design for each instance of the subset. It was performed to better analyse the relation between the metaheuristic performance and the parameter values. A face-centred central composite (FCC) design was selected, as in most of the cases the space parameter could not be expanded (since all parameters could only take values between 0 and 1). Figure **??** displays the scheme for instance p01. The objective function values for the same instance are represented in Figure **??**.

Then, the neighbourhood of each set that provided the best solution for an instance was explored applying another FCC design, centred on that set and covering half of the area analysed with the previous design. The sets that finally presented the best performance were stored. They are outlined in Table **??**. Random values were assigned to the instances that did not present variations in the results when changing the parameter values.

TABLE 9.5: Proposed list of sets of parameter values.

Medoids	Clusters	bM	bR
p01	p01	0.513	0.501
p07	p04, p07, p11, p18, pr02, pr05, pr09	0.001	0.372
p09	p03, p09, pr04, pr10	0.283	0.283
	p17	random	random
p19	p19	0.443	0.378
p22	p22	0.001	0.231
p23	p20, p23	0.449	0.250
pr06	p05, p06, p08, p10, p15, pr01, pr03, pr06, pr07, pr08	0.500	0.231
	p02, p12, p13, p14, p16, p21	random	random

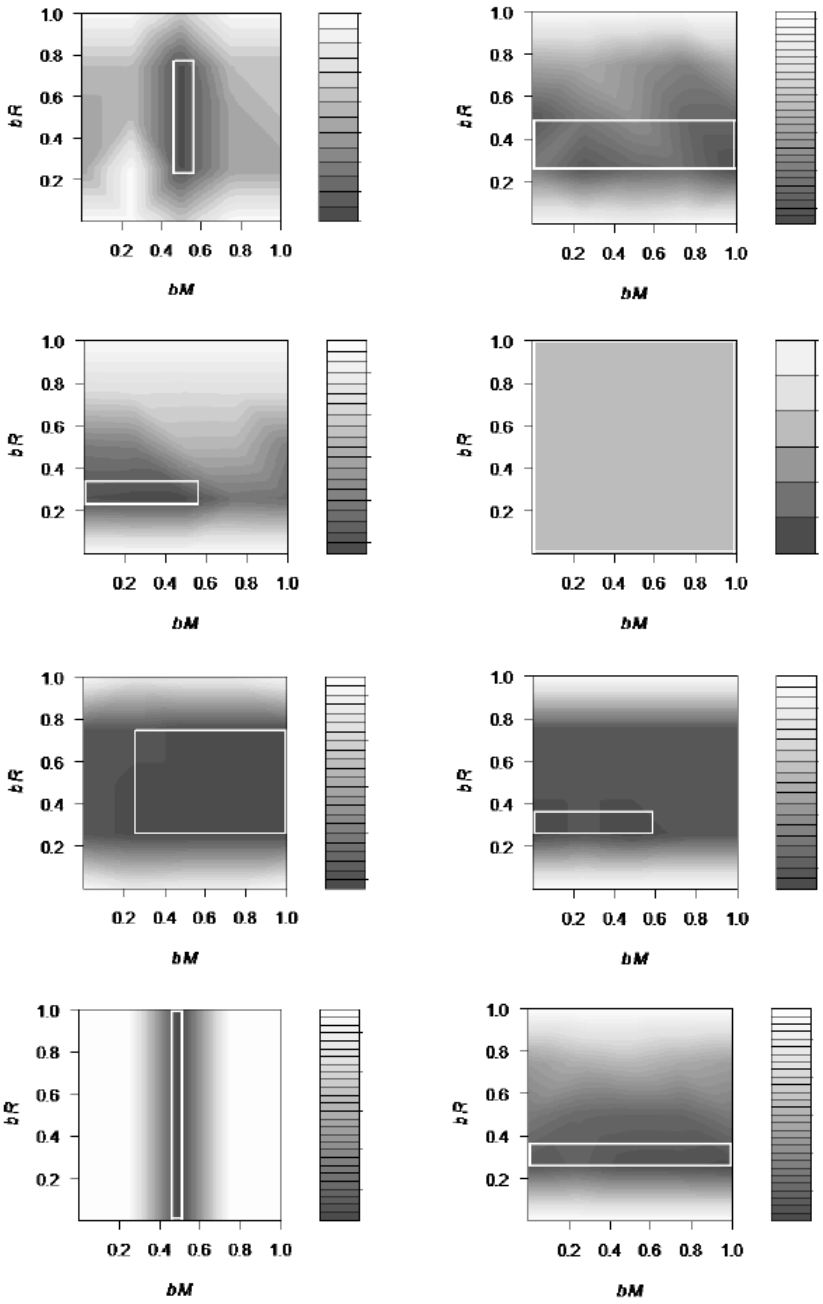


FIGURE 9.3: Contour plots of the medoids sorted from left to right, and top to bottom.

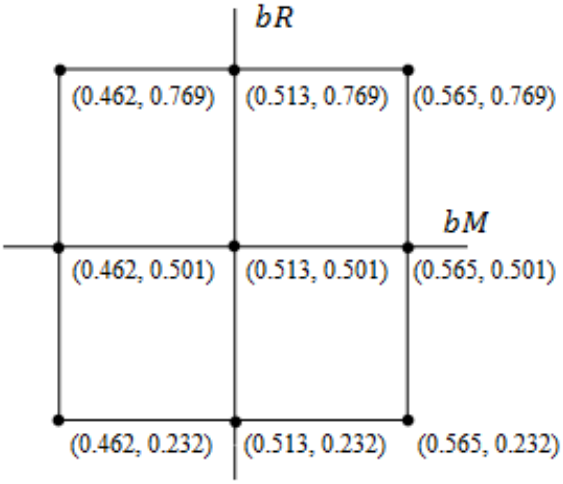


FIGURE 9.4: Scheme of the FCC design applied to the instance 'p01'.

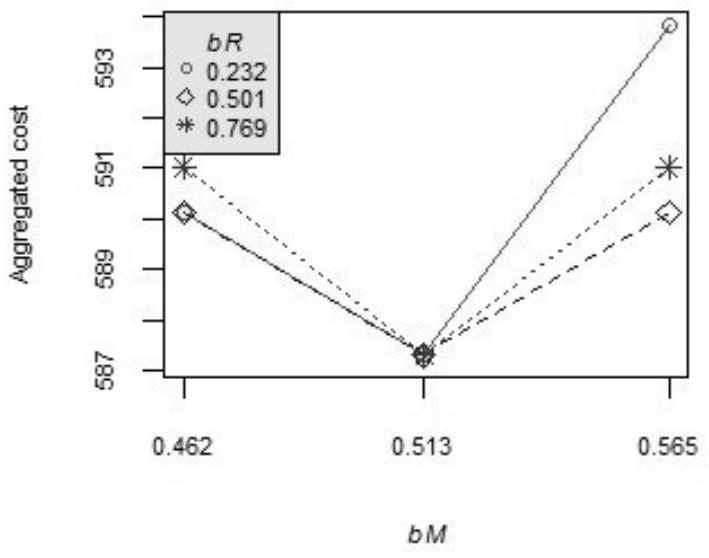


FIGURE 9.5: Solutions for the instance 'p01'.

9.2.4 Analysis of results

The following parameters were chosen to validate the list of sets: $J = 10$, $T = 3$, $\alpha = 0.05$, $r = 6$, $p = 0.7$. The number of sets randomly generated was fixed considering the trade-off between the reliability of our comparisons and the computational time required. The number of iterations was set considering only the time available. The significance level is the one most commonly used in the literature. The value of the fourth parameter is the mean rank that could be expected due to randomness with 11 solutions (1 set proposed and 10 randomly generated). The last parameter was calibrated to force the algorithm to provide good results at most of the instances. The algorithm was run 7 times with different seeds for each combination of parameter values, the medians and the minimum values were stored. The ranks of the results obtained are detailed in Table **??**. Ties receive a rank equal to the average of the ranks they span, shown inside the parentheses.

TABLE 9.6: Ranks of the results provided by our list and by 10 random sets.

Medoids	Rank (medians)	Rank (minimum values)
p01	1	3.5 (1-6)
p07	5	3.5 (1-6)
p09	2	2
p17	2 (1-3)	1
p19	6.5 (2-11)	10.5 (10-11)
p22	11	11
p23	1.5 (1-2)	1
pr06	5	1.5 (1-2)
Valid instances	0.75	0.75

According to our methodology, the list of sets can be considered valid as it presents a rank equal to or below 6 in 75% of the analysed instances, both considering medians and minimum values. In order to test our results, the algorithm was executed with the parameter values suggested in Juan et al. (2015c). Both series of results are comparable as were obtained using the same computer and stopping criteria based on the number of iterations. Table **??** presents the parameter values used in the aforementioned paper. Instead of setting fixed values, the authors introduced randomness by employing uniform distributions. The lower and upper bounds were selected after some tests.

TABLE 9.7: Sets of parameter values for comparison.

bM	bR	p*
Uniform (0.5, 0.8)	Uniform (0.1, 0.2)	Uniform (0.1, 0.5)

Table **??** shows the results obtained solving all instances with the proposed list of sets (our results, OR), and with the set proposed in Juan et al. (2015c) (JR).

The comparison of the solutions shows that our procedure achieves better results in most of the instances. Table **??** presents the average and the standard deviation of the differences, and the p-values of the test to compare the mean ranks of the results. It is a non-parametric test as the null hypothesis of the Shapiro-Wilk test, a test of normality, was rejected in all cases. The means are negatives, indicating that our methodology provides better solutions. The p-values reveal that the differences of the mean ranks are not statistically significant. Even though, the magnitude of the mean difference can be considered relevant in the context of the MDVRP.

9.3 Parallel computing

Desktop computers have become affordable machines that most people use every day for both work and leisure. Despite their current capacity, numerous institutions and individuals require more computational resources to execute intensive problem-solving processes. In these cases, DPCS constitute a useful approach. Multi-processors and/or multi-computers paradigms may be employed. A multi-computer schema presents a set of physical machines linked via network connections. These machines can be coupled geographically or in a more distributed environment (as in cloud computing). The main parallel paradigm is message passing, in which tasks and

TABLE 9.8: Instances experimental results.

Inst.	OR medians (1)	OR, minimum values (2)	JR, medians (3)	JR, minimum values (4)	% Gap (1) - (3)	% Gap (2) - (4)
p01	585.000	576.866	593.829	576.866	-1.509	0.000
p02	480.261	476.660	480.261	476.660	0.000	0.000
p03	644.464	641.186	649.229	641.186	-0.739	0.000
p04	1022.085	1019.570	1024.473	1024.062	-0.234	-0.441
p05	760.341	756.281	764.325	754.882	-0.524	0.185
p06	882.827	879.072	880.418	879.763	0.273	-0.079
p07	899.709	897.974	906.395	897.974	-0.743	0.000
p08	4440.534	4434.552	4438.407	4426.747	0.048	0.176
p09	3920.743	3906.561	3923.248	3900.274	-0.064	0.161
p10	3706.763	3667.344	3705.012	3687.054	0.047	-0.537
p11	3598.972	3584.691	3592.891	3585.690	0.169	-0.028
p12	1318.955	1318.955	1318.955	1318.955	0.000	0.000
p13	1318.955	1318.955	1318.955	1318.955	0.000	0.000
p14	1360.115	1360.115	1360.115	1360.115	0.000	0.000
p15	2573.393	2556.846	2573.393	2557.528	0.000	-0.027
p16	2605.565	2585.373	2605.565	2600.099	0.000	-0.570
p17	2720.231	2714.663	2725.799	2725.799	-0.205	-0.410
p18	3831.996	3806.783	3835.388	3806.783	-0.089	0.000
p19	3883.686	3883.686	3883.686	3881.427	0.000	0.058
p20	4080.348	4074.779	4091.482	4091.482	-0.273	-0.410
p21	5706.530	5692.789	5701.902	5692.789	0.081	0.000
p22	5808.738	5806.370	5806.480	5786.288	0.039	0.346
p23	6134.441	6128.873	6145.576	6123.306	-0.182	0.091
pr01	861.319	861.318	861.319	861.318	0.000	0.000
pr02	1330.495	1310.679	1331.543	1314.364	-0.079	-0.281
pr03	1813.634	1813.634	1814.452	1813.634	-0.045	0.000
pr04	2084.843	2077.582	2089.785	2079.832	-0.237	-0.108
pr05	2379.075	2359.947	2379.797	2368.525	-0.030	-0.363
pr06	2709.792	2693.680	2713.593	2696.504	-0.140	-0.105
pr07	1109.235	1109.235	1109.235	1109.235	0.000	0.000
pr08	1680.896	1674.930	1678.872	1674.594	0.120	0.020
pr09	2148.216	2147.192	2153.317	2142.650	-0.237	0.212
pr10	3016.255	3008.129	3028.606	3014.874	-0.409	-0.224

TABLE 9.9: Means and standard deviations of the differences and statistical tests.

		Mean of the differences	Standard deviation of the differences	P-value of the comparison of mean ranks
All instances	Medians	-0.149	0.330	0.954
	Minimum values	-0.070	0.219	0.980
All instances except the studied subset and those not analysed	Medians	-0.117	0.247	0.942
	Minimum values	-0.100	0.217	0.942

processes of different machines interchange data packets by sending and receiving messages to communicate.

SMEs are responsible for a significant part of the wealth generated in all developed economies. Often, they do neither possess advanced technical knowledge nor modern computational resources. However, a number of them could benefit from having more resources, for example to speed up intensive-computation processes or to obtain a higher performance. In order to access them, DPCS offer two alternatives: *(i)* to pay for using resources from an external provider; and *(ii)* to employ underutilized computer resources owned by the SME. This idea of aggregating idle or unused resources characterizes also volunteer computing systems. The main difference between both paradigms is that while the latter is usually associated to dynamic (any user can freely enter and leave) and heterogeneous environments, an SME knows the characteristics and the availability of its machines. Obviously, their scalability is also more limited. The alternative of using SME's underutilized resources presents several advantages. Firstly, SMEs do not have to send private information to servers of an external enterprise. Secondly, it is a cheaper solution since the SME does already have the resources. Finally, the energy consumption is reduced by seizing these resources, which could be still consuming otherwise (Cabrera, 2014). These desktop grids systems may be formed by personal computers with more computing capabilities than the required (standard computers in which employees mainly use word processors and spreadsheets, for instance)

or that are not used during some specific days or hours. Moreover, resources from several SMEs may be gathered to build a larger computing system. They can rely on a directory-of-resources service that keeps updated information of available computing resources. Once a user requires executing a process, he sends a query to this directory to select the resources and organize the tasks to perform. Once these tasks have been completed, the result is sent back to the user.

9.3.1 Methodology

The platform designed aims at facilitating the aggregation of a high number of computational resources by seizing underutilized or idle resources. It is based on software already installed in most computers, web browsers. Using a modern version of some of the commonest (Google Chrome, Firefox, or Internet Explorer), it may integrate a computer into the computing network. The only action required is to visit a website with an embedded JavaScript code that enables the communication in real-time with a job dispatcher service. Each one of the jobs includes a piece of data and the computing task to perform. Additional steps such as downloading, installing, or setting up additional software are not required, which makes this option a very attractive one for most SMEs. Because the ease to add new resources, this approach can be considered highly scalable. Thus, the described platform constitutes a flexible, simple, and scalable approach with multiple applications in SMEs.

The platform architecture is the typical of a master-slave cluster. The system has been designed to free the master from computationally expensive tasks. For the experiments described later, a single master has been sufficient to handle all the workload. In a production environment, the system could easily scale to thousands of slaves or even further when considering other architectures like a multi-master environment, etc. In our case, the master was placed on a dedicated server located on a cloud provider (Softlayer). The slaves were located over 2 different locations: The UOC's Lab and the Incubio's offices. The execution process goes as follows. First, the end-user submits the task to be executed to the master. This task consists mainly in a set of Map and Reduce functions written in JavaScript, as well as their input dataset. The master is responsible of creating a list of jobs. Each job is composed of a chunk from the dataset and the source of code that has to be executed over each piece of data. The master delivers and ensures that jobs are evenly distributed. After each job is completed, the master receives the results and stores them in a file or a database depending on the execution flow given by the user. The master keeps track of the jobs that have been assigned and processed. Different measures handle unfinished jobs, errors or exceptions that could appear unexpectedly by either rescheduling the jobs or stopping the execution and reporting the error.

Most approximate methods for solving COPs are probabilistic, which means that their solution depends on the seed used for a pseudo-random number generator. It has been proved that the execution time that an algorithm needs to report high-quality solutions can be reduced depending on this seed (Juan et al., 2014d). According to Talbi (2006) and Talbi (2009), DPCS are commonly employed to solve COPs. The typical approach in the related literature applies a master-slave scheme, in which a master or coordinator processor sends tasks to a set of slave processors in order to execute an intensive-computing process. Each slave is responsible for solving the same problem instance considering a different scenario, each one formed by a set of parameters and/or a seed. Once a slave has completed its task, it sends the solution to the master that stores it. In the simplest version, there is no communication between slaves. Following this approach, multiple instances of the algorithm are executed at the same time, each with a different seed. As shown in Figure **??**, each of these instances can be considered a cloned agent that is searching the solution space.

9.3.2 Computational experiments

In order to illustrate the benefits of the presented approach, two relevant COPs have been addressed: the CVRP and the PFSP. The randomized version of the CWS heuristic (Juan et al., 2014d) has been chosen for the CVRP. The Kelly instances are used to test our approach (Golden et al., 2008). The ILS-ESP algorithm (Juan et al., 2014a) has been employed to address the PFSP. It relies on a biased-randomized version of the NEH heuristic. The Taillard's benchmark instances (Taillard, 1993) are employed. They are grouped in 12 sets of 10, which are characterized by the following pairs of numbers of jobs and machines: 20x5, 20x10, 20x20, 50x5, 50x10, 50x20,

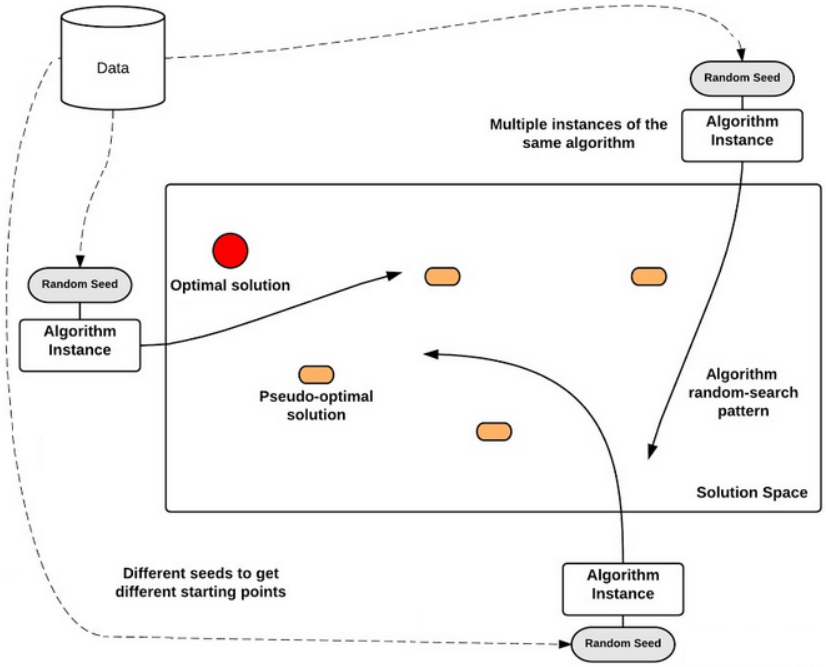


FIGURE 9.6: A multi-agent approach for solving COPs.

100x5, 100x10, 100x20, 200x10, 200x20, and 500x20. The instance resolution has been performed considering a specific combination of the parameters 'limit of time' (1, 5, 10, 15, 20, and 30 seconds) and 'number of agents' running in parallel (1, 4, 8, 16, 32, and 64).

All the experiments have been carried out using 64 slaves and 1 master. The master specifications are 3.5GHz Intel Xeon-IvyBridge with 8GB of RAM. The slaves are a heterogeneous set of desktop computers not having more than 8GB of RAM and up to 8 cores each. The machines were connected to the parallel computing environment using one of the following browsers: Microsoft Internet Explorer, Google Chrome or Mozilla Firefox, all of them with JavaScript enabled. The slaves were connected over a usual shared internet connection. For this reason, latencies or high speed connections were considered negligible.

9.3.3 Analysis of results

Figure **??** shows the results obtained after running the algorithm for solving the Kelly instances during 20 seconds of clock time per instance. Considering all instances, the first boxplot shows the gaps between the BKS and the solution generated by the CWS heuristic. The remaining boxplots show the gaps between the BKS and different executions of the randomized algorithm, each one using a different number of agents running in parallel. The number of agents tested were: 64, 128, and 256. It should be noticed that, for the 20 seconds considered, the distributed approach allows to reduce the gap down to almost 5% even for a reasonably low number of agents.

Regarding the PFSP instances, Table **??** summarizes the results of our computational experiments using a maximum time of 5 seconds. Each row refers to a different set of instances. Each column shows the gap between the BKS and our solution for different numbers of agents (1, 4, 8, 16, 32, and 64). Notice that the gaps shrink as the number of agents working in parallel is increased.

Figure **??** summarizes similar results for different values of the maximum clock time. It can be observed that, as time increases or as the number of agents increases, the average gap (for the entire set of benchmark instances) decreases. A detailed case is illustrated in Figure **??**, which displays the scatterplot of costs versus limit of time and number of agents for a given instance.

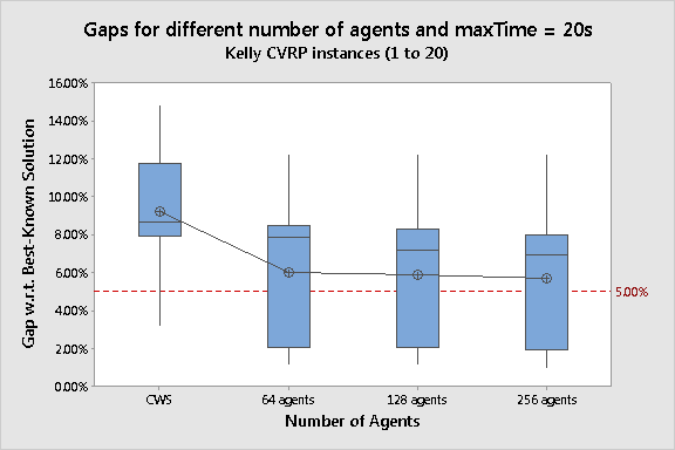


FIGURE 9.7: Results for the CVRP using the Kelly instances.

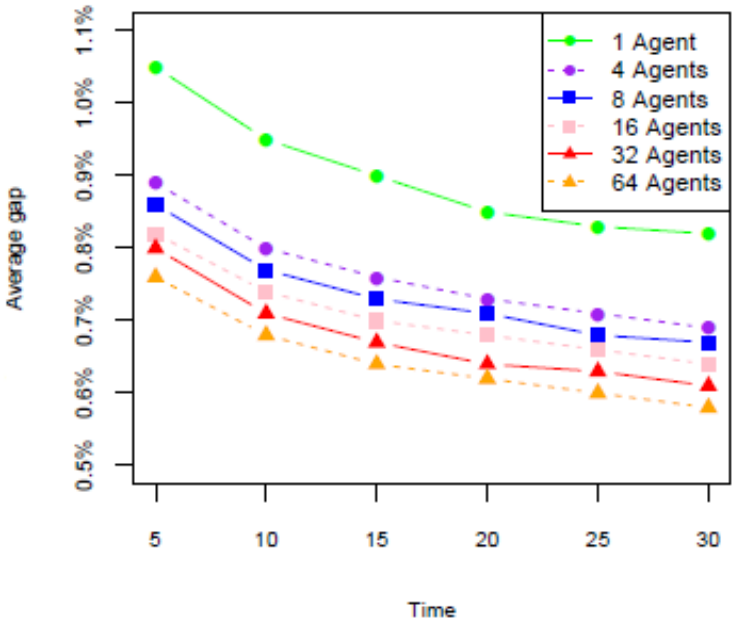


FIGURE 9.8: Average gaps for different numbers of agents and limits of time.

TABLE 9.10: Results for the PFSP considering the Taillard instances. Gaps for different number of agents and a maximum time of 5 seconds.

Taillard set	BKS - 1A	BKS - 4A	BKS - 8A	BKS -16A	BKS -32A	BKS - 64A
20x5	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
20x10	0.08%	0.08%	0.04%	0.00%	0.00%	0.00%
20x20	0.06%	0.02%	0.01%	0.00%	0.00%	0.00%
50x5	0.01%	0.01%	0.01%	0.01%	0.00%	0.00%
50x10	0.84%	0.74%	0.72%	0.65%	0.61%	0.54%
50x20	3.21%	2.85%	2.76%	2.68%	2.65%	2.58%
100x5	0.05%	0.02%	0.00%	0.00%	0.00%	0.00%
100x10	0.53%	0.33%	0.25%	0.22%	0.21%	0.18%
100x20	3.14%	2.67%	2.66%	2.63%	2.56%	2.46%
200x10	0.40%	0.26%	0.24%	0.23%	0.20%	0.14%
200x20	2.36%	2.20%	2.17%	2.15%	2.06%	2.00%
500x20	1.88%	1.53%	1.42%	1.32%	1.32%	1.26%
Averages	1.05%	0.89%	0.86%	0.82%	0.80%	0.76%

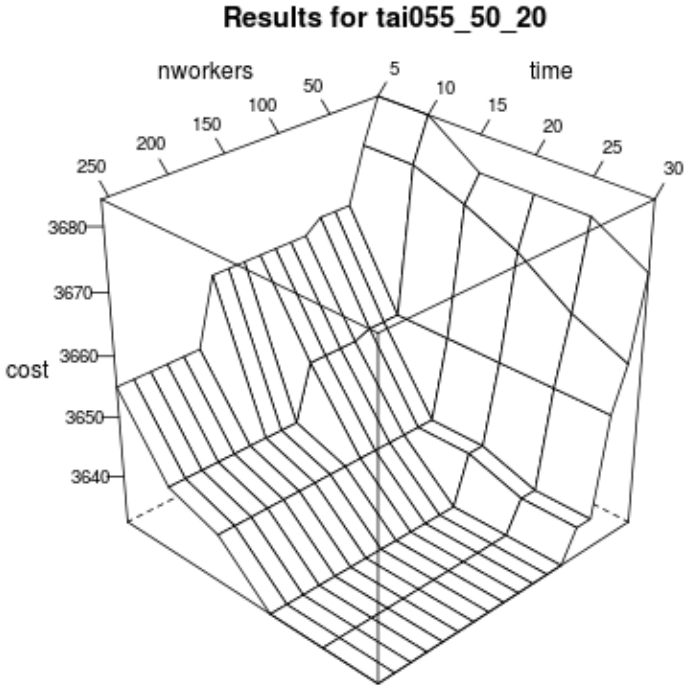


FIGURE 9.9: Objective solutions for different numbers of agents and limits of time.

9.4 Conclusions

The performance of metaheuristics is significantly affected by the parameter fine-tuning and the number of agents (each using a different seed for the random number generator) employed. These issues have not attracted as much attention as new metaheuristics and applications did. However, this trend is changing. In these lines, this chapter has presented two contributions. First, an overview on the PSP, a classification, a methodology and a description of a case study have been presented. Afterwards, an analysis of the computing time and the number of agents on the performance of two simple algorithms has been described. The conclusions drawn are:

- The parameter fine-tuning of a metaheuristic may be a time-consuming and complex problem, but may have a high effect on the quality of the solutions found.
- The literature on the PSP is diverse. Works can be grouped by whether the set of parameter values chosen is instance-specific and/or whether it evolves during the execution of an algorithm.

- A number of statistical learning techniques for solving the PSP have been proposed in the literature, but most works focus on DOE and/or lineal regression models. Thus, many options remain to be tested.
- SMEs may significantly benefit from DPCS by obtaining a higher number of computing resources seizing the underutilized resources.
- Computing time tends to have a small marginal effect on the performance of metaheuristics when it is set to more than a few seconds.
- The number of agents executing a given metaheuristic may have an important effect on its performance, but the marginal improvement decreases as the number of agents increases.

Part III

CONCLUSIONS, FUTURE RESEARCH AND CONTRIBUTIONS

Chapter 10

Conclusions and future work

10.1 Final conclusions

This thesis has explored the combination of statistical learning and simulation with metaheuristics for solving combinatorial optimization problems (COPs). It includes both methodological contributions and applications in a wide range of relevant and challenging fields.

First, an extensive review on works using statistical learning and metaheuristics as a solving approach has been presented. There is a high number of works, which are extremely different. Two groups are created: metaheuristics for improving statistical learning, and statistical learning for enhancing metaheuristics. Works in the first group can be classified according to the purpose of the statistical learning technique: classification, regression, clustering, and rule mining. On the other hand, the second group is split into two smaller groups: specifically-located hybridizations (including parameter fine-tuning, initialization, evaluation, population management, operators, and local search), and global hybridizations (reduction of search space, algorithm selection, hyperheuristics, cooperative strategies, and new types of metaheuristics).

Then, a novel hybrid methodology integrating statistical learning in metaheuristic frameworks has been proposed. It is designed to address COPs with dynamic inputs, which depend on the structure of the solution. A number of potential applications in popular fields have been identified, and an illustrative experiment has been carried out.

Applications to transportation constitute the main topic in applications. The multi-depot vehicle routing problem has been introduced, and three novel extensions have been addressed. First, the uncertainty regarding demands has been considered. It may significantly increase the total expected costs if no measures are undertaken to reduce the probability of route failures. A simheuristic approach considering safety stocks has been designed and tested. Afterwards, a version considering the maximization of benefits, heterogeneous depots and customers' preferences has been studied. In order to solve it, a methodology combining predictive models and a metaheuristic has been put forward. The third extension considers the introduction of sustainability indicators in the objective function and presents an analysis relying on visualization techniques to study the relationship between the different indicators. The aim is to take into account the negative impacts of transport activities. Later, the waste collection problem has been addressed, presenting methodologies for both the deterministic and stochastic versions. Finally, the heterogeneous site-dependent asymmetric vehicle routing problem with stochastic demands has been tackled, and a simheuristic methodology based on a successive approximations method has been applied. The potential applications of these problems in real-life have been described.

Regarding production, the distributed permutation flow-shop scheduling problem with stochastic times has been introduced. It describes the realistic scenario where there is a product composed of several intermediate products that have to be assembled at a given moment. These intermediate products are processed in independent distributed manufacturing factories, and each sub-problem is modelled as a permutation flow-shop scheduling problem. An approach relying on a simheuristic algorithm based on the iterated local search metaheuristic has been presented and tested.

Metaheuristics are becoming popular in finance. A survey on metaheuristics in portfolio optimization and risk management has been presented. Afterwards, the deterministic and stochastic versions of the portfolio optimization problem have been addressed. This problem is a strategy of selection of financial assets and determination of the optimal weights allocated to those assets that results in a desired portfolio return and associated minimum level of risk. The stochastic version deals with returns and covariances modelled as random variables.

Finally, two issues related to computing have been studied: the parameter fine-tuning, and the effect of the number of agents, and the maximum computing time on the performance of metaheuristics. While a methodology is presented and applied for the first issue, an exhaustive set of computational experiments have been carried out to gain insights into the second.

10.2 Directions for future work

Numerous lines of future research stem from this thesis. They are summarized in the following proposals:

- Extend the methodology of learnheuristics to address stochastic and/or multi-objective optimization problems, and develop an online version, in which information regarding new inputs can be used to improve the predictive model, and a blended version, in which predictions from several models are averaged, not necessarily giving the same weight to each of them.
- Design and test more approaches relying on learnheuristics for problems in dynamic fields such as telecommunications, volunteer computing or finance.
- Several rich vehicle routing problems have been addressed. Many realistic characteristics may be added, which may increase the complexity of the problems. It would be interesting to study the efficiency of repairing procedures when unexpected events take place. In addition, large supply chains with flexible structures with more agents than clients and depots could be included.
- Production systems have dramatically changed during the last decades, and some gaps remain in the literature. The most natural extension of the problem analyzed is to study the effects of dependent processing times.
- There is a high number of non-trivial optimization problems in finance. The uncertainty/risk of this field calls for the combination of optimization techniques and predictive models, and/or online optimization. Moreover, it would be interesting to analyze the impact of the width of the sample period and associated bear and bull market activity periods for the problems studied.
- In the computing arena, the calibration of parameters is still an open problem, since there is no single methodology accepted by the scientific community. Even more, there is no general agreement about the best techniques to use. However, journals devoted to applications of metaheuristics are becoming more demanding regarding this issue. In addition, the effects of the number of computational experiments and the maximum computing time on the performance requires more attention. Similarly, the use of parallel and distributed paradigms is just emerging in the field of metaheuristics. In an increasingly complex world where real-time solving approaches are required and statistical learning techniques may help to develop more intelligent/reactive approaches, these paradigms will play a relevant role.

Chapter 11

List of publications and presentations

This chapter lists the publications and presentations related to this thesis. It includes the accepted or in process of reviewing journal papers, and works in conferences and seminars developed in the last three years.

Ln′ indicates that the corresponding work can be found in the Appendix *L*.

11.1 Journal papers

First, the following articles have been submitted to ISI JCR and Elsevier-Scopus journals:

Indexed in ISI JCR

A1'. **Calvet, L.**, J. De Armas, D. Masip, and A. A. Juan (2017). "Learnheuristics: Hybridizing metaheuristics with machine learning for optimization problems with solution-dependent inputs". In: *Open Mathematics* (indexed in ISI SCI, 2015 IF = 0.512, Q3; 2015 SJR = 0.521, Q2).

A2'. Gruler, A., C. Quintero, **L. Calvet**, and A. A. Juan (2017). "Waste collection under uncertainty: A simheuristic based on variable neighborhood search". In: *European Journal of Industrial Engineering* 11.2 (indexed in ISI SCI, 2015 IF = 0.718, Q4; 2015 SJR = 1.000, Q1).

A3'. Pages, A., Ramalhinho, H., Juan, A., and **Calvet, L.** (2017). "Designing E-commerce supply chains: A stochastic facility-location approach". In: *International Transactions in Operational Research*, doi: 10.1111/itor.12433 (indexed in ISI SCI, 2015 IF = 1.255, Q2; 2015 SJR = 1.179, Q1).

A4'. **Calvet, L.**, A. Ferrer, I. Gomes, A. A. Juan, and D. Masip (2016). "Combining statistical learning with metaheuristics for the multi-depot vehicle routing problem with market segmentation". In: *Computers and Industrial Engineering* 94, pp. 93–104 (indexed in ISI SCI, 2015 IF = 2.086, Q1; 2015 SJR = 1.63, Q1).

A5'. **Calvet, L.**, A. A. Juan, C. Serrat, and J. Ries (2016). "A statistical learning based approach for parameter fine-tuning of metaheuristics". In: *Statistics and Operations Research Transactions* 40.1, pp. 201–224 (indexed in ISI SCI, 2015 IF = 0.414, Q4; 2015 SJR = 0.409, Q3).

Under review

B1'. **Calvet, L.**, R. Kizys, A. A. Juan, and J. Doering (submitted). "A VNS-based simheuristic methodology for the stochastic portfolio optimization problem". In: *Journal of the Operational Research Society*.

B2'. **Calvet, L.**, M. Lopeman, J. De Armas, G. Franco, and A. A. Juan (submitted). "Statistical and machine learning approaches for the minimization of trigger errors in earthquake catastrophe bonds". In: *Statistics and Operations Research Transactions*.

B3'. **Calvet, L.**, D. Wang, and A. A. Juan (submitted). "A simheuristic algorithm for the stochastic multi-depot vehicle routing problem". In: *International Transactions in Operational Research*.

B4'. Doering, J., **L. Calvet**, A. Fito, R. Kizys, and A. A. Juan (submitted). "Metaheuristics for realistic portfolio optimisation and risk management: Current state and future trends". In: *Annals of Operations Research*.

B5'. Doering, J., **L. Calvet**, R. Kizys, A. Fito, and A. A. Juan (submitted). "Rich portfolio optimization with stocks and individual commodity futures contracts". In: *Journal of Futures Markets*.

B6'. Gruler, A., T. Perez, **L. Calvet**, and A. A. Juan (submitted). "A simheuristic algorithm for time-dependent waste collection management with stochastic travel times". In: *Transportation Science*.

B7'. Hatami, S., **L. Calvet**, V. Fernandez-Viagas, J. Framinan, and A. A. Juan (submitted). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *International Transactions in Operational Research.*

B8'. Kizys, R., A. A. Juan, B. Sawik, and **L. Calvet** (submitted). "ARPO: An iterated local search algorithm for portfolio optimization under realistic constraints". In: *Quantitative Finance*.

B9'. Reyes, L., **L. Calvet**, C. Talens, A. A. Juan, and J. Faulin (submitted). "Sustainable Urban Freight Transport: a multi-depot vehicle routing problem considering different cost dimensions". In: *Journal of Heuristics*.

Indexed in Elsevier-Scopus

C1'. **Calvet, L.** and A. A. Juan (2015). "Educational data mining and e-learning analytics: An overview of goals, quantitative methods, and time-line evolution". In: *International Journal of Educational Technology in Higher Education* 12.3 (indexed in ISI ESCI, 2014 SJR = 0.215, Q3).

C2. **Calvet, L.**, A. A. Juan, R. Kizys, and J. De Armas (2016). "A SimILS-based methodology for a portfolio optimization problem with stochastic returns". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 3–11 (indexed in ISI Web of Science and Scopus, 2014 SJR = 0.244, Q3).

C3'. **Calvet, L.**, A. Pages, O. Travesset, and A. A. Juan (2016). "A simheuristic for the heterogeneous site-dependent asymmetric VRP with stochastic demands". In: *Springer Lecture Notes in Computer Science / LNAI* 9868, pp. 408-417 (indexed in ISI Web of Science and Scopus, 2014 SJR = 0.339, Q2).

C4. De Armas, J., **L. Calvet**, G. Franco, M. Lopeman, and A. A. Juan (2016). "Minimizing trigger error in parametric earthquake catastrophe bonds via statistical approaches". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 167-175 (indexed in ISI Web of Science and Scopus, 2014 SJR = 0.244, Q3).

C5. Doering, J., A. A. Juan, R. Kizys, A. Fito, and **L. Calvet** (2016). "Solving realistic portfolio optimization problems via metaheuristics: A survey and an example". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 22–30 (indexed in ISI Web of Science and Scopus, 2014 SJR = 0.244, Q3).

11.2 Conferences and seminars

Some works have been presented in conferences or seminars:

Indexed in ISI-WOS or Elsevier-Scopus

D1. **Calvet, L.**, V. Fernandez-Viagas, J. Framinan, and A. A. Juan (2016). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *Proceedings of the 2016 Winter Simulation Conference*. Washington D. C., USA, pp. 2347–2357.

Peer-review conferences

D2. **Calvet, L.**, J. Doering, R. Kizys, A. A. Juan, and A. Fito (2016). "The stochastic portfolio optimization problem: A formulation and a hybrid methodology". In: *OR58 Annual Conference*. Portsmouth, UK, pp. 127–128.

D3. **Calvet, L.**, A. A. Juan, and N. Schefers (2015). "Solving the multi-depot vehicle routing problem considering uncertainty and risk factors". In: *Proceedings of the ICRA6 / Risk 2015 Int. Conference*. Barcelona, Spain, pp. 187-194.

D4. **Calvet, L.**, A. A. Juan, and C. Serrat (2015). "Técnicas estadísticas aplicadas a la calibración de parámetros de metaheurísticas". In: *Proceedings of the X Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados*. Mérida, Spain, pp. 409–416.

D5. **Calvet, L.**, M. Mateo, A. A. Juan, and C. Laroque (2016). "Optimizing starting times in parallel multiple production lines with stochastic processing times and a shared deadline". In: *Proceedings of the 15th International Conference on Project Management and Scheduling*. Valencia, Spain.

D6. Juan, A. A., J. Faulin, and **L. Calvet** (2015). "Supporting real-time decision-making in logistics and transportation by combining simulation with heuristics". In: *Proceedings of the 12th Int. Multidisciplinary Modeling & Simulation Conf.* Bergeggi, Italy, pp. 35–38.

D7'. Ruiz, X., **L. Calvet**, J. Ferrarons, and A. A. Juan (2015). "SmartMonkey: a web browser tool for solving combinatorial optimization problems in real time". In: *Proceedings of the 2015 Int. Conf. of the Forum for Interdisciplinary Mathematics*. Barcelona, Spain.

Other international conferences

D8. **Calvet, L.**, A. Ferrer, A. A. Juan, and I. Gomes (2015). "Market segmentation issues in the multi-depot vehicle routing problem". In: *EURO 2015*. Glasgow, UK.

D9. Doering, J., **L. Calvet**, R. Kizys, A. Fito, and A. A. Juan (2016). "Rich portfolio optimization with stocks and individual commodity futures contracts". In: *Portsmouth-Fordham Conference on Banking & Finance*. Portsmouth, UK.

D10. Juan, A. A., J. Faulin, **L. Calvet**, A. Pages, and C. Quintero (2015). "Applications of simheuristics in transportation and logistics". In: *EURO 2015*. Glasgow, UK.

Seminars

D11. **Calvet, L.** (2015). "Hybridizing machine learning and metaheuristics". YouBRA Workshop. Barcelona, Spain.

D12. **Calvet, L.** (2015). "Neural networks for routing problems: review and challenges". Green COOP-CYTED Workshop. Madrid, Spain.

Bibliography

Adenso-Diaz, B. and M. Laguna (2006). "Fine-tuning of algorithms using fractional experimental designs and local search". In: *Operations Research* 54.1, pp. 99–114.

Adibi, M. A. and J. Shahrabi (2013). "A clustering-based modified variable neighborhood search algorithm for a dynamic job shop scheduling problem". In: *The International Journal of Advanced Manufacturing Technology* 70.9, pp. 1955–1961.

Adra, S. F., A. I. Hamody, I. Griffin, and P. J. Fleming (2005). "A hybrid multi-objective evolutionary algorithm using an inverse neural network for aircraft control system design." In: *Congress on Evolutionary Computation*. IEEE, pp. 1–8.

Affolter, K., T. Hanne, D. Schweizer, and R. Dornberger (2016). "Invasive weed optimization for solving index tracking problems". In: *Soft Computing* 20.9, pp. 3393–3401.

Agyei-Ampomah, S., D. Gounopoulos, and K. Mazouz (2014). "Does gold offer a better protection against losses in sovereign debt bonds than other metals?" In: *Journal of Banking and Finance* 40.1, pp. 507–521.

Al-Anzi, F. and A. Allahverdi (2007). "A self-adaptive differential evolution heuristic for two-stage assembly scheduling problem to minimize maximum lateness with setup times". In: *European Journal of Operational Research* 182.1, pp. 80–94.

— (2013). "An artificial immune system heuristic for two-stage multi-machine assembly scheduling problem to minimize total completion time". In: *Journal of Manufacturing Systems* 32.4, pp. 825–830.

Al Kattan, I. and R. Maragoud (2008). "Performance analysis of flowshop scheduling using genetic algorithm enhanced with simulation". In: *International Journal of Industrial Engineering: Theory, Applications and Practice* 15.1, pp. 62–72.

Al-Salem, A. (2004). "A heuristic to minimize makespan in proportional parallel flow shops". In: *International Journal of Computing & Information Sciences* 2.2, pp. 98–107.

Alexander, C. and A. Dimitriu (2004). "Equity indexing: Optimize your passive investments". In: *Quantitative Finance* 4.3, pp. 30–33.

Allaoui, H., S. Lamouri, and M. Lebbar (2006). "A robustness framework for a stochastic hybrid flow shop to minimize the makespan". In: *International Conference on Service Systems and Service Management*. Vol. 2. IEEE, pp. 1097–1102.

Alshraideh, H. and H. Abu Qdais (2016). "Stochastic modeling and optimization of medical waste collection in Northern Jordan". In: *Journal of Material Cycles and Waste Management*, pp. 1–11.

Amihud, Y. (1996). "Unexpected inflation and stock returns revisited–evidence from Israel". In: *Journal of Money, Credit and Banking* 28.1, pp. 22–33.

Andriosopoulos, K., M. Doumpos, N. C. Papapostolou, and P. K. Pouliasis (2013). "Portfolio optimization and index tracking for the shipping stock and freight markets using evolutionary algorithms". In: *Transportation Research Part E: Logistics and Transportation Review* 52, pp. 16–34.

Antonakakis, N. and R. Kizys (2015). *Dynamic spillovers between commodity and currency markets*.

Armañanzas, R. and J. A. Lozano (2005). "A multiobjective approach to the portfolio optimization problem". In: *IEEE Congress on Evolutionary Computation*. Vol. 2. IEEE Press, pp. 1388–1395.

Asta, S. and E. Ozcan (2014). "An apprenticeship learning hyper-heuristic for vehicle routing in HyFlex". In: *2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, pp. 65–72.

Auger, A. and N. Hansen (2005). "Performance evaluation of an advanced local search evolutionary algorithm". In: *2005 IEEE congress on evolutionary computation*. Vol. 2. IEEE, pp. 1777–1784.

Ayodele, A. A. and K. A. Charles (2015). "Portfolio selection problem using generalized differential evolution 3". In: *Applied Mathematical Sciences* 9.42, pp. 2069–2082.

Babaei, S., M. M. Sepehri, and E. Babaei (2015). "Multi-objective portfolio optimization considering the dependence structure of asset returns". In: *European Journal of Operational Research* 244.2, pp. 525–539.

Back, B., T. Laitinen, and K. Sere (1996). "Neural networks and genetic algorithms for bankruptcy predictions". In: *Expert Systems with Applications* 11.4, pp. 407–413.

Bae, S.-T., H.S. Hwang, G.-S. Cho, and M.-J. Goan (2007). "Integrated GA-VRP solver for multi-depot system". In: *Computers & Industrial Engineering* 53, pp. 233–240.

Baker, K. R. and D. Altheimer (2012). "Heuristic solution methods for the stochastic flow shop problem". In: *European Journal of Operational Research* 216.1, pp. 172–177.

Baker, K. R. and D. Trietsch (2011). "Three heuristic procedures for the stochastic, two-machine flow shop problem". In: *Journal of Scheduling* 14.5, pp. 445–454.

Baker, Kenneth R (1974). *Introduction to sequencing and scheduling*. John Wiley & Sons.

Bansal, Y., S. Kumar, and P. Verma (2014). "Commodity futures in portfolio diversification: Impact on investor's utility". In: *Global Business & Management Research* 6.2, pp. 112–121.

Banu, P. K. Nizar and S. Andrews (2015). "Gene clustering using metaheuristic optimization algorithms". In: *International Journal of Applied Metaheuristic Computing* 6.4, pp. 14–38.

Baptista, S., R. C. Oliveira, and E. Zúquete (2002). "A period vehicle routing case study". In: *European Journal of Operational Research* 139, pp. 220–229.

Barreto, S., C. Ferreira, J. Paixao, and B. Sousa (2007). "Using clustering analysis in a capacitated location-routing problem". In: *Eur. J. Oper. Res.* 179.3, pp. 968–977.

Bartz-Beielstein, T., K. E. Parsopoulos, and M. N. Vrahatis (2004). "Design and analysis of optimization algorithms using computational statistics". In: *Applied Numerical Analysis & Computational Mathematics* 1.2, pp. 413–433.

Bastian, C. and A. H. G. R. Kan (1992). "The stochastic vehicle routing problem revisited". In: *European Journal of Operational Research* 56, pp. 407–412.

Battiti, R. and M. Brunato (2005). *Reactive search: machine learning for memory-based heuristics*. Tech. rep. Teofilo F. Gonzalez (Ed.), Approximation Algorithms and Metaheuristics, Taylor & Francis Books (CRC Press.

— (2010). "Reactive search optimization: learning while optimizing". In: *Handbook of Metaheuristics*. Springer, pp. 543–571.

Battiti, R. and G. Tecchiolli (1994). "The reactive tabu search". In: *ORSA journal on computing* 6.2, pp. 126–140.

Bautista, J., E. Fernández, and J. Pereira (2008). "Solving an urban waste collection problem using ants heuristics". In: *Computers & Operations Research* 35, pp. 3020–3033.

Beasley, J. E. (2013). "Portfolio optimisation: Models and solution approaches". In: *Tutorials in operations research* 10, pp. 201–221.

Beasley, J. E., N. Meade, and T.-J. Chang (2003). "An evolutionary heuristic for the index tracking problem". In: *European Journal of Operational Research* 148.3, pp. 621–643.

Bektaş, T. and G. Laporte (2011). "The pollution-routing problem". In: *Transportation Research Part B: Methodological* 45.8, pp. 1232–1250.

Beliën, J., L. De Boeck, and J. Van Ackere (2014). "Municipal solid waste collection and management problems: a literature review". In: *Transportation Science* 48, pp. 78–102.

Belousova, J. and G. Dorfleitner (2012). "On the diversification benefits of commodities from the perspective of euro investors". In: *Journal of Banking and Finance* 36.9, pp. 2455–2472.

Beltrami, E. J. and L. D. Bodin (1974). "Networks and vehicle routing for municipal waste collection". In: *Networks* 4, pp. 65–94.

Benjamin, A. M. and J. E. Beasley (2010). "Metaheuristics for the waste collection vehicle routing problem with time windows, driver rest period and multiple disposal facilities". In: *Computers & Operations Research* Vol. 37, pp. 2270–2280.

Berberoğlu, A. and A. Uyar (2010). "A hyper-heuristic approach for the unit commitment problem". In: *European Conference on the Applications of Evolutionary Computation*. Springer, pp. 121–130.

Berry, K. (2009). "Distributed and Grid Computing via the browser". In:

Bertsimas, D. J. (1988). "Probabilistic combinatorial optimization problems". PhD thesis. Operations Research Center, Massachusetts Institute of Technology.

Bianchi, L., M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto (2006). "Hybrid metaheuristics for the vehicle routing problem with stochastic demands". In: *Journal of Mathematical Modelling and Algorithms* 5, pp. 91–110.

Bianchi, L., D. Marco, L. M. Gambardella, and W. J. Gutjahr (2009). "A survey on metaheuristics for stochastic combinatorial optimization". In: *Natural Computing* 8, pp. 239–287.

Bienstock, D. (1996). "Computational study of a family of mixed-integer quadratic programming problems". In: *Mathematical programming* 74.2, pp. 121–140.

Birattari, M. and J. Kacprzyk (2009). *Tuning metaheuristics: a machine learning perspective*. Vol. 197. Springer.

Birattari, M., Z. Yuan, P. Balaprakash, and T. Stützle (2010). "F-Race and iterated F-Race: an overview". In: *Experimental methods for the analysis of optimization algorithms*. Springer, pp. 311–336.

Borshchev, A. and A. Filippov (2004). "From system dynamics and discrete event to practical agent based modeling: reasons, techniques, tools". In: *Proceedings of the 22nd international conference of the system dynamics society*. Vol. 22.

Boussaïd, I., J. Lepagnot, and P. Siarry (2013). "A survey on optimization metaheuristics". In: *Information Sciences* 237, pp. 82–117.

Branch and Cut. http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/. [Online; accessed March 2016].

Brasileiro, R. C., V. L. F. Souza, B. J. T. Fernandes, and A. L. I. Oliveira (2013). "Automatic method for stock trading combining technical analysis and the artificial bee colony algorithm". In: *2013 IEEE Congress on Evolutionary Computation*, pp. 1810–1817.

Brooks, C. and M. Prokopczuk (2013). "The dynamics of commodity prices". In: *Quantitative Finance* 13.4, pp. 527–542.

Brownlee, A. E. I., O. Regnier-Coudert, J. A. W. McCall, and S. Massie (2010). "Using a Markov network as a surrogate fitness function in a genetic algorithm". In: *IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–8.

Buhrkal, K., A. Larsen, and S. Ropke (2012). "The waste collection vehicle routing problem with time windows in a city logistics context". In: *Procedia - Social and Behavioral Sciences* 39, pp. 241–254.

Burke, E. K., M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward (2010). "A classification of hyper-heuristic approaches". In: *Handbook of metaheuristics*, pp. 449–468.

Burke, E. K., M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu (2013). "Hyper-heuristics: a survey of the state of the art". In: *Eur. J. Oper. Res.* 64.12, pp. 1695–1724.

Cabrera, G., A. A. Juan, D. Lázaro, J. M. Marquès, and I. Proskurnia (2014). "A simulation-optimization approach to deploy Internet services in large-scale systems with user-provided resources". In: *Simulation* 90.6, pp. 644–659.

Caceres, J., P. Arias, D. Guimarans, D. Riera, and A. A. Juan (2014). "Rich vehicle routing problem: a survey". In: *ACM Computing Surveys* 47.2. ISSN: 0360-0300.

Cadenas, J. M., M. C. Garrido, and E. Muñoz (2009). "Using machine learning in a cooperative hybrid parallel strategy of metaheuristics". In: *Inform. Sciences* 179.19, pp. 3255–3267.

Calvet, L. (2015a). "Hybridizing machine learning and metaheuristics". YouBRA Workshop. Barcelona, Spain.

— (2015b). "Neural networks for routing problems: review and challenges". Green COOPCYTED Workshop. Madrid, Spain.

Calvet, L. and A. A. Juan (2015). "Educational Data Mining and e-Learning Analytics: an overview of goals, quantitative methods, and time-line evolution". In: *Int. J. of Educational Technology in Higher Education* 12.3.

Calvet, L., A. Ferrer, A. A. Juan, and I. Gomes (2015a). "Market segmentation issues in the multi-depot vehicle routing problem". In: *EURO 2015*. Glasgow, UK.

Calvet, L., A. A. Juan, and N. Schefers (2015b). "Solving the multi-depot vehicle routing problem considering uncertainty and risk factors". In: *Proceedings of the ICRA6 / Risk 2015 Int. Conference*. Barcelona, Spain, pp. 187–194.

Calvet, L., A. A. Juan, and C. Serrat (2015c). "Técnicas estadísticas aplicadas a la calibración de parámetros de metaheurísticas". In: *Proceedings of the X Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados*. Mérida, Spain, pp. 409–416.

Calvet, L., A. Pages, O. Travesset, and A. A. Juan (2016a). "A simheuristic for the heterogeneous site-dependent asymmetric VRP with stochastic demands". In: *Springer Lecture Notes in Computer Science / LNAI*. Vol. 9868, pp. 408–417.

Calvet, L., A. A. Juan, C. Serrat, and J. Ries (2016b). "A statistical learning based approach for parameter fine-tuning of metaheuristics". In: *Statistics and Operations Research Transactions* 40.1, pp. 201–224.

Calvet, L., V. Fernandez-Viagas, J. Framinan, and A. A. Juan (2016c). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *Proceedings of the 2016 Winter Simulation Conference*. Washington D. C., USA, pp. 2347–2357.

Calvet, L., A. Ferrer, I. Gomes, A. A. Juan, and D. Masip (2016d). "Combining statistical learning with metaheuristics for the multi-depot vehicle routing problem with market segmentation". In: *Computers and Industrial Engineering* 94, pp. 93–104.

Calvet, L., M. Mateo, A. A. Juan, and C. Laroque (2016e). "Optimizing starting times in parallel multiple production lines with stochastic processing times and a shared deadline". In: *Proceedings of the 15th International Conference on Project Management and Scheduling*. Valencia, Spain.

Calvet, L., J. Doering, R. Kizys, A. A. Juan, and A. Fito (2016f). "The stochastic portfolio optimization problem: a formulation and a hybrid methodology". In: *OR58 Annual Conference*. Portsmouth, UK, pp. 127–128.

Calvet, L., J. De Armas, D. Masip, and A. A. Juan (2017). "Learnheuristics: Hybridizing metaheuristics with machine learning for optimization problems with solution-dependent inputs". In: *Open Mathematics*.

Calvet, L., D. Wang, and A. A. Juan (submitted[a]). "A simheuristic algorithm for the stochastic multi-depot vehicle routing problem". In: *International Transactions in Operational Research*.

Calvet, L., R. Kizys, A. A. Juan, and J. Doering (submitted[b]). "A VNS-based simheuristic methodology for the stochastic portfolio optimization problem". In: *Journal of the Operational Research Society*.

Calvet, L., M. Lopeman, J. De Armas, G. Franco, and A. A. Juan (submitted[c]). "Statistical and machine learning approaches for the minimization of trigger errors in earthquake catastrophe bonds". In: *Statistics and Operations Research Transactions*.

Campbell, H. G., R. A. Dudek, and M. L. Smith (1970). "A heuristic algorithm for the n job, m machine sequencing problem". In: *Management science* 16.10, B630–B637.

Canakgoz, N. A. and J. E. Beasley (2009). "Mixed-integer programming approaches for index tracking and enhanced indexation". In: *European Journal of Operational Research* 196.1, pp. 384–399.

Cao, D. and M. Chen (2003). "Parallel flowshop scheduling using Tabu search". In: *International Journal of Production Research* 41.13, pp. 3059–3073.

Carazo, A. F., T. Gómez, J. Molina, A. G. Hernández-Díaz, F. M. Guerrero, and R. Caballero (2010). "Solving a comprehensive model for multiobjective project portfolio selection". In: *Computers & operations research* 37.4, pp. 630–639.

Carvalho, A. R., F. M. Ramos, and A. A. Chaves (2011). "Metaheuristics for the feedforward artificial neural network architecture optimization problem". In: *Neural Computing and Applications* 20.8, pp. 1273–1284.

Caserta, M. and E. Quiñonez Rico (2009). "A cross entropy-Lagrangean hybrid algorithm for the multi-item capacitated lot-sizing problem with setup times". In: *Computers & OR* 36.2, pp. 530–548.

Ceberio, J., E. Irurozki, A. Mendiburu, and J. A. Lozano (2012). "A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems". In: *Pattern Recogn.* 1.1, pp. 103–117.

Cesarone, F., A. Scozzari, and F. Tardella (2013). "A new method for mean-variance portfolio optimization with cardinality constraints". In: *Annals of Operations Research* 205.1, pp. 213–234.

Chan, F. T.S., S.H. Chung, and P.L.Y. Chan (2005). "An adaptive genetic algorithm with dominated genes for distributed scheduling problems". In: *Expert Systems with Applications* 29.2, pp. 364–371.

Chan, Y., W.B. Carter, and M.D. Burnes (2001). "A hybrid algorithm for multi-depot vehicle routing problem". In: *Computers & Operations Research* 28.8, pp. 803–826.

Chang, T.-J., N. Meade, J. E. Beasley, and Y. M. Sharaiha (2000). "Heuristics for cardinality constrained portfolio optimisation". In: *Computers & Operations Research* 27.13, pp. 1271–1302.

ChangYoon, L. and K. Way (2001). "Reliability optimization design using a hybridized genetic algorithm with a neural-network technique". In: *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* 84.2, pp. 627–637.

Chao, I., B. Golden, and E. Wasil (1993). "A new heuristic for the multi-depot vehicle routing problem that improves upon best known solutions". In: *American Journal of Mathematical and Management Sciences* 13.3-4, pp. 371–406.

Chapelle, O., B. Schölkopf, and A. Zien (2006). *Semi-supervised learning. Adaptive computation and machine learning series*.

Chavarnakul, T. and D. Enke (2009). "A hybrid stock trading system for intelligent technical analysis-based equivolume charting". In: *Neurocomputing* 72.16, pp. 3517–3528.

Chen, L.-W., P. Sharma, and Y.-C. Tseng (2013). "Dynamic traffic control with fairness and throughput optimization using vehicular communications". In: *IEEE Journal on Selected Areas in Communications* 31.9, pp. 504–512.

Chen, M.-Y. (2011). "A hybrid model for business failure prediction-utilization of particle swarm optimization and support vector machines". In: *Neural Network World* 21.2, p. 129.

Chen, P. and X. Xu (2008). "A hybrid algorithm for multi-depot vehicle routing problem". In: *IEEE International Conference on Service Operations and Logistics, and Informatics*, pp. 2031–2034.

Cheung, C. S. and P. Miu (2010). "Diversification benefits of commodity futures". In: *Journal of International Financial Markets, Institutions and Money* 20.5, pp. 451–474.

Chi, B.-W. and C.-C. Hsu (2012). "A hybrid approach to integrate genetic algorithm into dual scoring model in enhancing the performance of credit scoring model". In: *Expert Systems with Applications* 39.3, pp. 2650–2661.

Chiam, S. C., K. C. Tan, and A. Al Mamun (2013). "Dynamic index tracking via multi-objective evolutionary algorithm". In: *Applied Soft Computing* 13.7, pp. 3392–3408.

Chibeles-Martins, N., T. Pinto-Varela, A. P. Barbosa-Póvoa, and A. Q. Novais (2016). "A multi-objective meta-heuristic approach for the design and planning of green supply chains". In: *Expert Systems with Applications* 47, pp. 71–84.

Chica, M., A. A. Juan, O. Cordon, and W. D. Kelton (submitted). "Why Simheuristics?: Benefits, limitations, and best practices when combining metaheuristics with simulation". In: *Simulation Modelling Practice and Theory*, pp. –.

Choi, S. H. and K. Wang (2012). "Flexible flow shop scheduling with stochastic processing times: a decomposition-based approach". In: *Computers & Industrial Engineering* 63.2, pp. 362–373.

Chong, J. and J. Miffre (2010). "Conditional correlation and volatility in commodity futures and traditional asset markets". In: *Journal of Alternative Investments* 12.3, pp. 61–75.

Chorafas, D. N. (2007). "What is meant by risk management?" In: *Risk Management Technology in Financial Services*, pp. 24–42.

Chow, G., E. Jacquier, M. Kritzman, and K. Lowry (1999). "Optimal portfolios in good times and bad". In: *Financial Analysts Journal* 55.3, pp. 65–73.

Christofides, N. and J. E. Beasley (1984). "The period routing problem". In: *Networks* 14, pp. 237–256.

Chua, J. H., G. Sick, and R. S. Woodward (1990). "Diversifying with gold stocks". In: *Financial Analysts Journal* 46.4, pp. 76–79.

Ciner, C., C. Gurdgiev, and B. M. Lucey (2013). "Hedges and safe havens: An examination of stocks, bonds, gold, oil and exchange rates". In: *International Review of Financial Analysis* 29, pp. 202–211.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Cordeau, J.F. and M. Maischberger (2012). "A parallel iterated tabu search heuristic for vehicle routing problems". In: *Computers & Operations Research* 39.9, pp. 2033–2050.

Cordeau, J.F., M. Gendreau, and G. Laporte (1997). "A tabu search heuristic for periodic and multidepot vehicle routing problems". In: *Networks* 30.2, pp. 105–119.

Corne, D., C. Dhaenens, and L. Jourdan (2012). "Synergies between operations research and data mining: The emerging use of multi-objective approaches". In: *Eur. J. Oper. Res.* 221.3, pp. 469–479.

Coulouris, G. F., J. Dollimore, and T. Kindberg (2005). *Distributed systems: concepts and design*. Pearson education.

Coy, S. P., B. L. Golden, G. C. Runger, and E. A. Wasil (2001). "Using experimental design to find effective parameter settings for heuristics". In: *Journal of Heuristics* 7.1, pp. 77–97.

Crevier, B., J. F. Cordeau, and G. Laporte (2007). "The multi-depot vehicle routing problem with inter-depot routes". In: *European Journal of Operational Research* 176, pp. 756–773.

Crowston, W. B., F. Glover, and J. D. Trawick (1963). *Probabilistic and parametric learning combinations of local job shop scheduling rules*. Tech. rep. DTIC Document.

Cruz, L., E. Fernandez, C. Gomez, G. Rivera, and F. Perez (2014). "Many-objective portfolio optimization of interdependent projects with 'a priori'incorporation of decision-maker preferences". In: *Appl. Math* 8.4, pp. 1517–1531.

Cura, T. (2009). "Particle swarm optimization approach to portfolio optimization". In: *Nonlinear Analysis: Real World Applications* 10.4, pp. 2396–2406.

Czarn, A., C. MacNish, K. Vijayan, B. Turlach, and R. Gupta (2004). "Statistical exploratory analysis of genetic algorithms". In: *Evolutionary Computation, IEEE Transactions on* 8.4, pp. 405–421.

Dalboni, F. L., L. S. Ochi, and L.M.A. Drummond (2003). "On improving evolutionary algorithms by using data mining for the oil collector vehicle routing problem". In: *International Network Optimization Conference*, pp. 182–188.

Danenas, P. and G. Garsva (2015). "Selection of support vector machines based classifiers for credit risk domain". In: *Expert Systems with Applications* 42.6, pp. 3194–3204.

Daskalaki, C. and G. Skiadopoulos (2011). "Should investors include commodities in their portfolios after all? New evidence". In: *Journal of Banking and Finance* 35.10, pp. 2606–2626.

Davalos, S., F. Leng, E. H. Feroz, and Z. Cao (2014). "Designing an if–then rules-based ensemble of heterogeneous bankruptcy classifiers: A genetic algorithm approach". In: *Intelligent Systems in Accounting, Finance and Management* 21.3, pp. 129–153.

David, W.H.E., A. Kusiak, and A. Artiba (1996). "A scheduling problem in glass manufacturing". In: *IIE Transactions (Institute of Industrial Engineers)* 28.2, pp. 129–139.

De Armas, J., L. Calvet, G. Franco, M. Lopeman, and A. A. Juan (2016). "Minimizing trigger error in parametric earthquake catastrophe bonds via statistical approaches". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 167–175.

De Jong, K. (2007). "Parameter setting in EAs: a 30 year perspective". In: *Parameter setting in evolutionary algorithms*. Springer, pp. 1–18.

De Lima, Francisco Chagas, Jorge Dantas De Melo, and Adrião Duarte Doria Neto (2008). "Using the Q-learning algorithm in the constructive phase of the GRASP and reactive GRASP metaheuristics". In: pp. 4169–4176. ISBN: 9781424418213.

Deb, K., S. Bandaru, D. Greiner, A. Gaspar-Cunha, and C. C. Tutum (2014). "An integrated approach to automated innovization for discovering useful design principles: Case studies from engineering". In: *Applied Soft Computing* 15, pp. 42–56.

Delucchi, M. A. and D. R. McCubbin (2010). *External costs of transport in the US*. Tech. rep.

Demir, E., T. Van Woensel, and T. de Kok (2014). "Multidepot distribution planning at logistics service provider Nabuurs BV". In: *Interfaces* 44.6, pp. 591–604.

Deng, G.-F., W.-T. Lin, and C.-C. Lo (2012). "Markowitz-based portfolio selection with cardinality constraints using improved particle swarm optimization". In: *Expert Systems with Applications* 39.4, pp. 4558–4566.

Derigs, U. and N. Nickel (2003). "Meta-heuristic based decision support for portfolio optimization with a case study on tracking error minimization in passive portfolio management". In: *OR Spectrum* 25.3, pp. 345–378.

Derigs, U. and N. H. Nickel (2004). "On a local-search heuristic for a class of tracking error minimization problems in portfolio management". In: *Annals of Operations Research* 131.1-4, pp. 45–77.

Dhaenens, C. and L. Jourdan (2016). *Metaheuristics for Big Data*. Wiley. ISBN: 9781119347606.

Di Gaspero, L., G. Di Tollo, A. Roli, and A. Schaerf (2011). "Hybrid metaheuristics for constrained portfolio selection problems". In: *Quantitative Finance* 11.10, pp. 1473–1487.

Di Tollo, G. and A. Roli (2008). "Metaheuristics for the portfolio selection problem". In: *International Journal of Operations Research* 5.1, pp. 13–35.

Díaz-Manríquez, A., G. Toscano-Pulido, and W. Gómez-Flores (2011). "On the selection of surrogate models in evolutionary optimization algorithms". In: *2011 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, pp. 2155–2162.

Dobslaw, F. (2010). "A parameter tuning framework for metaheuristics based on design of experiments and artificial neural networks". In: *World Academy of Science, Engineering and Technology* 64, pp. 213–216.

Doering, J., L. Calvet, R. Kizys, A. Fito, and A. A. Juan (2016a). "Rich portfolio optimization with stocks and individual commodity futures Contracts". In: *Portsmouth-Fordham Conference on Banking & Finance. Portsmouth, UK*.

Doering, J., A. A. Juan, R. Kizys, A. Fito, and L. Calvet (2016b). "Solving realistic portfolio optimization problems via metaheuristics: a survey and an example". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 22–30.

Doering, J., L. Calvet, A. Fito, R. Kizys, and A. A. Juan (submitted[a]). "Metaheuristics for realistic portfolio optimization and risk management: current state and future trends". In: *Annals of Operations Research*.

Doering, J., L. Calvet, R. Kizys, A. Fito, and A. A. Juan (submitted[b]). "Rich portfolio optimization with stocks and individual commodity futures contracts". In: *Journal of Futures Markets*.

Doerner, K., W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer (2004). "Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection". In: *Annals of operations research* 131.1-4, pp. 79–99.

Doerner, K. F., W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer (2006). "Pareto ant colony optimization with ILP preprocessing in multiobjective project portfolio selection". In: *European Journal of Operational Research* 171.3, pp. 830–841.

Dominguez, O., D. Guimarans, A. A. Juan, and I. de la Nuez (2016a). "A Biased-Randomised Large Neighbourhood Search for the two-dimensional Vehicle Routing Problem with Backhauls". In: *European Journal of Operational Research*.

Dominguez, O., A. A. Juan, J. Barrios B.and Faulin, and A. Agustin (2016b). "Using biased randomization for solving the two-dimensional loading vehicle routing problem with heterogeneous fleet". In: *Annals of Operations Research* 236.2, pp. 383–404.

Dondo, R. and J. Cerdá (2009). "A hybrid local improvement algorithm for large-scale multi-depot vehicle routing problems with time windows". In: *Computers & Chemical Engineering* 33, pp. 513–530.

Dorigo, M. (1992). "Optimization, learning and natural algorithms". PhD thesis. Politecnico di Milano, Italy.

Dror, M. and P. Trudeau (1986). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 23, pp. 228–235.

Eiben, A. E., R. Hinterding, and Z. Michalewicz (1999). "Parameter control in evolutionary algorithms". In: *Evolutionary Computation, IEEE Transactions on* 3.2, pp. 124–141.

Escalante, H. J., V. Ponce-López, S. Escalera, X. Baró, A. Morales-Reyes, and J. Martínez-Carranza (2016). "Evolving weighting schemes for the bag of visual words". In: *Neural Comput. Appl.* 0, pp. 1–15.

Escobar, J. W., R. Linfati, P. Toth, and M. G. Baldoquin (2014). "A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem". In: *Journal of Heuristics* 20, pp. 483–509.

Euchi, J. (2014). "Hybrid estimation of distribution algorithm for a multiple trips fixed fleet vehicle routing problems with time windows". In: *International Journal of Operational Research* 21.4, p. 433. ISSN: 1745-7645.

European Commission (2015). *EU Transport in figures*. Statistical pocketbook.

European Union (2011a). *Eurostat regional yearbook 2014. Statistical books*. Tech. rep. Publications Office of the European Union.

— (2011b). *Roadmap to a single European Transport Area – towards a competitive and resource-efficient transport system*. Tech. rep. Publications Office of the European Union.

Eurostat (2015). *Sustainable development in the European Union: 2015 monitoring report of the EU sustainable development strategy*. Publications office of the European Union.

Evans, C., K. Pappas, and F. Xhafa (2013). "Utilizing artificial neural networks and genetic algorithms to build an algo-trading model for intra-day foreign exchange speculation". In: *Mathematical and Computer Modelling* 58.5, pp. 1249–1266.

Fama, E. F. (1981). "Stock returns, real activity, inflation, and money". In: *The American Economic Review* 71.4, pp. 545–565.

Farmer, J. D., N. H. Packard, and A. S. Perelson (1986). "The immune system, adaptation, and machine learning". In: *Phys. D* 2.1-3, pp. 187–204.

Faulin, J., A. Juan, F. Lera, and S. Grasman (2011). "Solving the capacitated vehicle routing problem with environmental criteria based on real estimations in road transportation: a case study". In: *Procedia-Social and Behavioral Sciences* 20, pp. 323–334.

Faulin, J., A. A. Juan, S. E. Grasman, and M. J. Fry (2012). *Decision making in service industries: A practical approach*. CRC Press.

Feo, T. A. and M. G. C. Resende (1989). "A probabilistic heuristic for a computationally difficult set covering problem". In: *Oper. Res. Lett.* 8.2, pp. 67–71.

Feo, T. A and M. G. C. Resende (1995). "Greedy randomized adaptive search procedures". In: *Journal of global optimization* 6.2, pp. 109–133.

Fernández, A. and S. Gómez (2007). "Portfolio selection using neural networks". In: *Computers and Operations Research* 34.4, pp. 1177–1191.

Fernandez, E., C. Gomez, G. Rivera, and L. Cruz-Reyes (2015). "Hybrid metaheuristic approach for handling many objectives and decisions on partial support in project portfolio optimisation". In: *Information Sciences* 315, pp. 102–122.

Fernández-Caballero, J. C., F. J. Martinez, C. Hervas, and P. A. Gutierrez (2010). "Sensitivity versus accuracy in multiclass problems using memetic pareto evolutionary neural networks". In: *IEEE T. Neural Network.* 21.5, pp. 750–770.

Fernandez-Viagas, V. and J. M. Framinan (2015c). "NEH-based heuristics for the permutation flowshop scheduling problem to minimise total tardiness". In: *Computers & Operations Research* 60, 27—36.

Ferone, D., A. Facchiano, A. Marabotti, and P. Festa (2016). "A new GRASP metaheuristic for biclustering of gene expression data". In: *PeerJ PrePrints*.

Figueira, G., M. Furlan, and B. Almada-Lobo (2013). "Predictive production planning in an integrated pulp and paper mill". In: *Manufacturing Modelling, Management, and Control*. Vol. 7. 1, pp. 371–376.

Fikar, C., A. A. Juan, E. Martinez, and P. Hirsch (2016). "A discrete-event driven metaheuristic for dynamic home service routing with synchronised trip sharing". In: *European Journal of Industrial Engineering* 10, pp. 323–340.

Framinan, J. M. and P. Perez-Gonzalez (2015). "On heuristic solutions for the stochastic flowshop scheduling problem". In: *Journal of Operational Research* 246.2, 413—420.

Freitas, A. (2002). "Data mining and knowledge discovery with evolutionary algorithms". In: *Advances in Evolutionary Computation* 105, pp. 819–845.

— (2008). "A review of evolutionary algorithms for data mining". In: *Soft Computing for Knowledge Discovery and Data Mining*. Springer, pp. 79–111.

Gao, J. and R. Chen (2011). "A hybrid genetic algorithm for the distributed permutation flowshop scheduling problem". In: *International Journal of Computational Intelligence Systems* 4.4, pp. 497–508.

Gao, J., R. Chen, and W. Deng (2013). "An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem". In: *International Journal of Production Research* 51.3, pp. 641–651.

García, V., A. I. Marqués, and J. S. Sánchez (2015). "An insight into the experimental design for credit risk and corporate bankruptcy prediction systems". In: *Journal of Intelligent Information Systems* 44.1, pp. 159–189.

Gardi, F., T. Benoist, J. Darlay, B. Estellon, and R. Megel (2014). *Mathematical programming solver based on local search*. John Wiley & Sons.

Garey, M. R., D. S. Johnson, and R. Sethi (1976). "The complexity of flowshop and jobshop scheduling". In: *Mathematics of Operations Research* 1.2, pp. 117–129.

Garrett, I. and N. Taylor (2001). "Portfolio diversification and excess comovement in commodity prices". In: *The Manchester School* 69.4, pp. 351–368.

Gaspar-Cunha, A. and Armando S. Vieira (2004). "A hybrid multi-objective evolutionary algorithm using an inverse neural network". In: pp. 25–30.

Gaspar-Cunha, A., G. Recio, L. Costa, and C. Estébanez (2014). "Self-adaptive MOEA feature selection for classification of bankruptcy prediction data". In: *The Scientific World Journal* 2014.

Gass, S. I. and A. A. Assad (2005). "Model world: tales from the time line—the definition of OR and the origins of Monte Carlo simulation". In: *Interfaces* 35.5, pp. 429–435.

Geman, H. and C. Kharoubi (2008). "WTI crude oil futures in portfolio diversification: the time-to-maturity effect". In: *Journal of Banking and Finance* 32.12, pp. 2553–2559.

Gendreau, M., G. Laporte, and R. Séguin (1995). "An exact algorithm for the vehicle routing problem with stochastic demands and customers". In: *Transportation Science* 29, pp. 143–155.

— (1996). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 88, pp. 3–12.

Ghasemzadeh, F. and N. P. Archer (2000). "Project portfolio selection through decision support". In: *Decision support systems* 29.1, pp. 73–88.

Ghezail, F., H. Pierreval, and S. Hajri-Gabouj (2010). "Analysis of robustness in proactive scheduling: a graphical approach". In: *Computers & Industrial Engineering* 58.2, pp. 193–198.

Ghiani, G., F. Guerriero, G. Improta, and R. Musmanno (2005). "Waste collection in Southern Italy: solution of a real-life arc routing problem". In: *International Transactions in Operational Research* 12, pp. 135–144.

Ghiani, G., C. Mourão, L. Pinto, and D. Vigo (2014). "Routing in waste collection applications". In: *Arc routing: problems, methods, and applications*. Ed. by A. Corberán and G. Laporte. Philadelphia: SIAM Monographs on Discrete Mathematics and Applications, pp. 351–370.

Gillett, B.E. and J.G. Johnson (1976). "Multi-terminal vehicle-dispatch algorithm". In: *Omega* 4.6, pp. 711–718.

Gilli, M., D. Maringer, and E. Schumann (2011). *Numerical methods and optimization in finance*. Academic Press.

Glover, F. (1977). "Heuristics for integer programming using surrogate constraints". In: *Decision Science* 8.1, pp. 156–166.

— (1986). "Future paths for integer programming and links to artificial intelligence". In: *Computers & Operations Research* 13.5, pp. 533–549.

— (1989). "Tabu search—part I". In: *ORSA Journal on computing* 1.3, pp. 190–206.

— (2000). "Multi-start and strategic oscillation methods—principles to exploit adaptive memory". In: *Computing tools for modeling, optimization and simulation*, pp. 1–23.

Göçken, M., M. Özçalıcı, A. Boru, and A. T. Dosdoğru (2016). "Integrating metaheuristics and artificial neural networks for improved stock price prediction". In: *Expert Systems with Applications* 44, pp. 320–331.

Golden, B. L., S. Raghavan, and E. A. Wasil (2008). *The vehicle routing problem: latest advances and new challenges*. Vol. 43. Springer Science & Business Media.

Golden, B.L., T.L. Magnanti, and H.Q. Nguyen (1977). "Implementing vehicle routing algorithms". In: *Networks* 7.2, pp. 113–148.

Golmakani, H. R. and M. Fazel (2011). "Constrained portfolio selection using particle swarm optimization". In: *Expert Systems with Applications* 38.7, pp. 8327–8335.

Gonzalez, S., A. A. Juan, D. Riera, M. Elizondo, and J. Ramos (2016). "A simheuristic algorithm for solving the arc routing problem with stochastic demands". In: *Journal of Simulation*.

Goodson, J. C. (2015). "A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands". In: *European Journal of Operational Research* 241, pp. 361–369.

Gordini, N. (2014). "A genetic algorithm approach for SMEs bankruptcy prediction: Empirical evidence from Italy". In: *Expert Systems with Applications* 41.14, pp. 6433–6445.

Gorgulho, A., R. Neves, and N. Horta (2011). "Applying a GA kernel on optimizing technical analysis rules for stock picking and portfolio composition". In: *Expert systems with Applications* 38.11, pp. 14072–14085.

Grasas, A., A. Juan, J. Faulin, J. De Armas, and H. Ramalhinho (submitted). "Biased Randomization of Heuristics using Skewed Probability Distributions: a survey and some applications". In: *Computers and Industrial Engineering*, pp. –.

Gruler, A., A. A. Juan, and M. Steglich (2015). "Proceedings of the 2015 Metaheuristics International Conference. Agadir, Morocco". In: *A heuristic approach for smart waste collection management*.

Gruler, A., C. Quintero, L. Calvet, and A. A. Juan (2017). "Waste collection under uncertainty: A simheuristic based on variable neighborhood search". In: *European Journal of Industrial Engineering* 11.2.

Gruler, A., T. Perez, L. Calvet, and A. A. Juan (submitted). "A simheuristic algorithm for time-dependent waste collection management with stochastic travel times". In: *Transportation Science*.

Guastaroba, G. and M. G. Speranza (2012). "Kernel search: An application to the index tracking problem". In: *European Journal of Operational Research* 217.1, pp. 54–68.

Gulczynski, D., B. L. Golden, and E. Wasil (2011). "The multi-depot split delivery vehicle routing problem: An integer programming-based heuristic, new test problems, and computational results". In: *Computers & Industrial Engineering* 61, pp. 794–804.

Gumustekin, S., T. Senel, and M. A. Cengiz (2014). "A comparative study on Bayesian optimization algorithm for nutrition problem". In: *J. Food Nutr. Res.* 2.12, pp. 952–958.

Gunawan, A., H. C. Lau, and E. Wong (2013). "Real-world parameter tuning using factorial design with parameter decomposition". In: *Advances in Metaheuristics*. Springer, pp. 37–59.

Gutjahr, W. J., S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk (2008). "Competence-driven project portfolio selection, scheduling and staff assignment". In: *Central European Journal of Operations Research* 16.3, pp. 281–306.

Gutjahr, W. J., S. Katzensteiner, C. Reiter P.and Stummer, and M. Denk (2010). "Multi-objective decision analysis for competence-oriented project portfolio selection". In: *European Journal of Operational Research* 205.3, pp. 670–679.

Hahn, G. J. and N. Doganaksoy (2012). *A career in statistics: beyond the numbers*. John Wiley & Sons.

Han, H. and E. Ponce-Cueto (2015). "Waste collection vehicle routing problem: literature review". In: *Traffic & Transportation* 27, pp. 345–358.

Hansen, P., Mladenović N., Brimberg J., and Moreno J. A. (2010). "Variable neighborhood search". In: *Handbook of metaheuristics*. Ed. by Gendreau M. and Potvin J.-Y. 2nd. Springer, pp. 61–86.

Hardouvelis, G. A. (1987). "Macroeconomic information and stock prices". In: *Journal of Economics and Business* 39.2, pp. 131–140.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning*. 2nd ed. Springer.

Hatami, S., R. Ruiz, and C. Andrés-Romano (2013). "The distributed assembly permutation flowshop scheduling problem". In: *International Journal of Production Research* 51.17, pp. 5292–5308.

— (2015). "Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times". In: *International Journal of Production Economics* 169, pp. 76–88.

Hatami, S., L. Calvet, V. Fernandez-Viagas, J. Framinan, and A. A. Juan (submitted). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *International Transactions in Operational Research*, pp. –.

He, G. and N. Huang (2012). "A modified particle swarm optimization algorithm with applications". In: *Applied Mathematics and Computation* 219.3, pp. 1053–1060.

— (2014). "A new particle swarm optimization algorithm with an application". In: *Applied Mathematics and Computation* 232, pp. 521–528.

Hemmelmayr, V., Karl F. Doerner, R. F. Hartl, and S. Rath (2013). "A heuristic solution method for node routing based solid waste collection problems". In: *Journal of Heuristics* 19, pp. 129–156.

Hemmelmayr, V., K. F. Doerner, R. F. Hartl, and D. Vigo (2014). "Models and algorithms for the integrated planning of bin allocation and vehicle routing in solid waste management". In: *Transportation Science* 48, pp. 103–120.

Hillier, D., P. Draper, and R. Faff (2006). *Do precious metals shine? An investment perspective*.

Ho, W., G.T.S. Ho, P. Ji, and H.C.W. Lau (2008). "A hybrid genetic algorithm for the multi-depot vehicle routing problem". In: *Engineering Applications of Artificial Intelligence* 21.4, pp. 548–557.

Holland, J. H. (1962). "The compact genetic algorithm". In: *Journal of the ACM* 3.9, pp. 297–314.

Hooker, J. N. (1995). "Testing heuristics: We have it all wrong". In: *Journal of Heuristics* 1.1, pp. 33–42.

Hruschka, E. R., R. J. G. B. Campello, and A. A. Freitas (2009). "A survey of evolutionary algorithms for clustering". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 39.2, pp. 133–155.

Hsu, C.-Y., P.-C. Chang, and M.-H. Chen (2015). "A linkage mining in block-based evolutionary algorithm for permutation flowshop scheduling problem". In: *Computers & Industrial Engineering* 83, pp. 159–171.

Hu, X.-B. and X.-Y. Huang (2004). "Solving TSP with characteristic of clustering by ant colony algorithm". In: *Acta Simulata Systematica Sinica* 12, p. 014.

Hutter, F., H. H. Hoos, K. Leyton-Brown, and T. Stützle (2009). "ParamILS: an automatic algorithm configuration framework". In: *Journal of Artificial Intelligence Research* 36.1, pp. 267–306.

Ismail, Z. and I. Irhamah (2008). "Solving the vehicle routing problem with stochastic demands via hybrid genetic algorithm-tabu search". In: *Journal of Mathematics and Statistics* 4, pp. 161–167.

Ismail, Z. and S.L. Loh (2009). "Ant colony optimization for solving solid waste collection scheduling problems". In: *Journal of Mathematics and Statistics* 5, pp. 199–205.

Jabbarpour, M. R., R. Md. Noor, and R. H. Khokhar (2015). "Green vehicle traffic routing system using ant-based algorithm". In: *Journal of Network and Computer Applications* 58, pp. 294–308.

Jaffe, J. F. (1989). "Gold and gold stocks as investments for institutional portfolios". In: *Financial Analysts Journal* 45.2, pp. 53–59.

Jensen, G. R., R. R. Johnson, and J. M. Mercer (2002). "Tactical asset allocation and commodity futures". In: *The Journal of Portfolio Management* 28.4, pp. 100–111.

Jeong, S.-J., K.-S. Kim, and Y.-H. Lee (2009). "The efficient search method of simulated annealing using fuzzy logic controller". In: *Expert Systems with Applications* 36.3, pp. 7099–7103.

Jin, Y. (2005). "A comprehensive survey of fitness approximation in evolutionary computation". In: *Soft. Comput.* 9.1, pp. 3–12.

Jin, Y. and B. Sendhoff (2004). "Reducing fitness evaluations using clustering techniques and neural network ensembles". In: *Genetic and Evolutionary Computation Conference*. Springer, pp. 688–699.

Johnson, D. S. (2002). "A theoretician's guide to the experimental analysis of algorithms". In: *Data structures, near neighbor searches, and methodology: fifth and sixth DIMACS implementation challenges* 59, pp. 215–250.

Johnson, S.M. (1954). "Optimal two- and three-stage production schedules with setup times included". In: *Naval research logistics quarterly* 1, pp. 61–68.

Jourdan, L., D. Corne, D. Savic, and G. Walters (2005). "Preliminary investigation of the learnable evolution model for faster/better multiobjective water systems design". In: *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, pp. 841–855.

Jourdan, L., C. Dhaenens, and E.-G. Talbi (2006). "Using datamining techniques to help metaheuristics: a short survey". In: *International Workshop on Hybrid Metaheuristics*. Springer Berlin Heidelberg. Gran Canaria, Spain, pp. 57–69.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, M. Gilibert, and X. Vilajosana (2009). "Using oriented random search to provide a set of alternative solutions to the capacitated vehicle routing problem". In: *Operations Research and Cyber-Infrastructure*. Springer, New York, USA, pp. 331–346.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., J. Faulin, S. Grasman, D. Riera, J. Marull, and C. Mendez (2011b). "Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands". In: *Transportation Research Part C: Emerging Technologies* 19.5, pp. 751–765.

Juan, A. A., J. Faulin, A. Ferrer, H. Lourenço, and B. Barrios (2013a). "MIRHA: multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems". In: *TOP* 21, pp. 109–132.

Juan, A. A., J. Faulin, J. Jorba, J. Caceres, and J. M. Marques (2013b). "Using parallel & distributed computing for real-time solving of vehicle routing problems with stochastic demands". In: *Annals of Operations Research* 207.1, pp. 43–65.

Juan, A. A., B. Barrios, E. Vallada, D. Riera, and J. Jorba (2014a). "A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times". In: *Simulation Modelling Practice and Theory* 46, pp. 101–117.

Juan, A. A., S. E. Grasman, J. Caceres-Cruz, and T. Bektaş (2014b). "A simheuristic algorithm for the single-period stochastic inventory-routing problem with stock-outs". In: *Simulation Modelling Practice and Theory* 46, pp. 40–52.

Juan, A. A., J. Faulin, J. Caceres, B. Barrios, and E. Martinez (2014c). "A successive approximations method for the heterogeneous vehicle routing problem: Analyzing different fleet configurations". In: *European Journal of Industrial Engineering* 8.6, pp. 762–788.

Juan, A. A., J. Cáceres-Cruz, S. Gonzalez, D. Riera, and B. Barrios (2014d). "Biased randomization of classical heuristics". In: *Encyclopedia of Business Analytics and Optimization, IGI Global* 1, pp. 314–324.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Juan, A. A., J. Faulin, L. Calvet, A. Pages, and C. Quintero (2015b). "Applications of simheuristics in transportation and logistics". In: *EURO 2015*. Glasgow, UK.

Juan, A. A., I. Pascual, D. Guimarans, and B. Barrios (2015c). "Combining biased randomization with iterated local search for solving the multidepot vehicle routing problem". In: *International Transactions in Operational Research* 22.4, pp. 647–667.

Juan, A. A., J. Faulin, and L. Calvet (2015d). "Supporting real-time decision-making in logistics and transportation by combining simulation with heuristics". In: *Proceedings of the 12th Int. Multidisciplinary Modeling & Simulation Conf.* Bergeggi, Italy, pp. 35–38.

Kadziński, M., T. Tervonen, M. K. Tomczyk, and R. Dekker (2017). "Evaluation of multi-objective optimization approaches for solving green supply chain design problems". In: *Omega* 68, pp. 168–184.

Kahn, K. B., S. E. Kay, R. J. Slotegraaf, and S. Uban (2013). *The PDMA handbook of new product development*. Third. John Wiley & Sons, Inc. ISBN: 9780470648209.

Kanda, J. Y., A. C. P. L. F. Carvalho, E. R. Hruschka, and C. Soares (2011). "Using meta-learning to recommend meta-heuristics for the traveling salesman problem". In: *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*. Vol. 1. IEEE, pp. 346–351.

Karakatic, S. and V. Podgorelec (2015). "A survey of genetic algorithms for solving multi depot vehicle routing problem". In: *Applied Soft Computing* 27, pp. 519–532.

Katragjini, K., E. Vallada, and R. Ruiz (2013). "Flow shop rescheduling under different types of disruption". In: *International Journal of Production Research* 51.3, pp. 780–797.

Kennedy, J. and R. C. Eberhart (1995). "Particle swarm optimization". In: *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948.

Kenyon, A. S. and D. P. Morton (2003). "Stochastic vehicle routing with random travel times". In: *Transportation Science* 37, pp. 69–82.

Khabzaoui, M, C Dhaenens, and E.-G. Talbi (2004). "A multicriteria genetic algorithm to analyze DNA microarray data". In: *Cec2004: Proceedings of the 2004 Congress on Evolutionary Computation, Vols 1 and 2*, pp. 1874–1881.

Khabzaoui, M., C. Dhaenens, and E.-G. Talbi (2008). "Combining evolutionary algorithms and exact approaches for multi-objective knowledge discovery". In: *RAIRO Operations Research* 42.1, pp. 69–83.

Kianfar, K., S. M. T. F. Ghomi, and A. O. Jadid (2012). "Study of stochastic sequence-dependent flexible flow shop via developing a dispatching rule and a hybrid GA". In: *Engineering Applications of Artificial Intelligence* 25.3, pp. 494–506.

Kim, B., S. Kim, and S. Sahoo (2006). "Waste collection vehicle routing problem with time windows". In: *Computers & Operations Research* Vol. 33, pp. 3624–3642.

Kim, M.-J. and I. Han (2003). "The discovery of experts' decision rules from qualitative bankruptcy data using genetic algorithms". In: *Expert Systems with Applications* 25.4, pp. 637–646.

Kim, M.-J. and D.-K. Kang (2010). "Ensemble with neural networks for bankruptcy prediction". In: *Expert Systems with Applications* 37.4, pp. 3373–3379.

— (2012). "Classifiers selection in ensembles using genetic algorithms for bankruptcy prediction". In: *Expert Systems with applications* 39.10, pp. 9308–9314.

Kirkos, E. (2015). "Assessing methodologies for intelligent bankruptcy prediction". In: *Artificial Intelligence Review* 43.1, pp. 83–123.

Kirkpatrick, S. (1984). "Optimization by simulated annealing: Quantitative studies". In: *Journal of statistical physics* 34.5-6, pp. 975–986.

Kizys, R., A. A. Juan, B. Sawik, and L. Calvet (submitted). "ARPO: an iterated local search algorithm for portfolio optimization under realistic constraints". In: *Quantitative Finance*.

Koç, Ç., T. Bektaş, O. Jabali, and G. Laporte (2014). "The fleet size and mix pollution-routing problem". In: *Transportation Research Part B: Methodological* 70, pp. 239–254.

Komaki, G.M., E. Teymourian, V. Kayvanfar, and Z. Booyavi (2017). "Improved discrete cuckoo optimization algorithm for the three-stage assembly flowshop scheduling problem". In: *Computers and Industrial Engineering* 105, pp. 158–173.

Koulamas, C. and G. J. Kyparisis (2001). "The three-stage assembly flowshop scheduling problem". In: *Computers & Operations Research* 28.7, pp. 689–704.

Kozeny, V. (2015). "Genetic algorithms for credit scoring: Alternative fitness function performance comparison". In: *Expert Systems with Applications* 42.6, pp. 2998–3004.

Krink, T. and S. Paterlini (2011). "Multiobjective optimization using differential evolution for real-world portfolio optimization". In: *Computational Management Science* 8.1-2, pp. 157–179.

Krink, T., S. Mittnik, and S. Paterlini (2009). "Differential evolution and combinatorial search for constrained index-tracking". In: *Annals of Operations Research* 172.1, pp. 153–176.

Kuhn, M. (2008). "Building predictive models in R using the caret package". In: *Journal of Statistical Software* 28.5.

Kumar, P. R. and V. Ravi (2007). "Bankruptcy prediction in banks and firms via statistical and intelligent techniques–A review". In: *European journal of operational research* 180.1, pp. 1–28.

Kuo, Y. (2010). "Using simulated annealing to minimize fuel consumption for the time-dependent vehicle routing problem". In: *Computers & Industrial Engineering* 59.1, pp. 157–165.

Kurada, R. R., K. K. Pavan, and A. V. Rao (2013). "A preliminary survey on optimized multiobjective metaheuristic methods for data clustering using evolutionary approaches". In: *International Journal of Computer Science & Information Technology* 5.

Lantz, B. (2013). *Machine learning with R*. Packt Publishing.

Laporte, G., F. Louveaux, and H. Mercure (1992). "The vehicle routing problem with stochastic travel times". In: *Transportation Science* 26, pp. 161–170.

Lessmann, S., M. Caserta, and I. M. Arango (2011). "Tuning metaheuristics: a data mining based approach for particle swarm optimization". In: *Expert Systems with Applications* 38.10, pp. 12826–12838.

Leung, Y.-W. and Y. Wang (2001). "An orthogonal genetic algorithm with quantization for global numerical optimization". In: *Trans. Evol. Comp* 5.1, pp. 41–53.

Li, J., E. K. Burke, and R. Qu (2011a). "Integrating neural networks and logistic regression to underpin hyper-heuristic search". In: *Knowl.-based Syst.* 24.2, pp. 322–330.

Li, J., P.M. Pardalos, H. Sun, J. Pei, and Y. Zhang (2015). "Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups". In: *Expert Systems with Applications* 42.7, pp. 3551–3561.

Li, Q. and L. Bao (2014). "Enhanced index tracking with multiple time-scale analysis". In: *Economic Modelling* 39, pp. 282–292.

Liao, C.-J., C.-H. Lee, and H.-C. Lee (2015). "An efficient heuristic for a two-stage assembly scheduling problem with batch setup times to minimize makespan". In: *Computers & Industrial Engineering* 88, pp. 317–325.

Lim, D., Y. Jin, Y.-S. Ong, and B. Sendhoff (2010). "Generalizing Surrogate-assisted Evolutionary Computation". In: *Trans. Evol. Comp* 14.3, pp. 329–355.

Lin, Q., L. Gao, X. Li, and C. Zhang (2015). "A hybrid backtracking search algorithm for permutation flow-shop scheduling problem". In: *Computers & Industrial Engineering* 85, pp. 437–446.

Lin, S. (1965). "Computer solutions of the traveling salesman problem". In: *Bell System Technical Journal* 44.10, pp. 2245–2269.

Lin, S. W., K. C. Ying, and C. Y. Huang (2013). "Minimising makespan in distributed permutation flowshops using a modified iterated greedy algorithm". In: *International Journal of Production Research* 51.16, pp. 5029–5038.

Lin, X., Z. Yang, and Y. Song (2011). "Intelligent stock trading system based on improved technical analysis and echo state network". In: *Expert systems with Applications* 38.9, pp. 11347–11354.

Liu, Q., S. Ullah, and C. Zhang (2011). "An improved genetic algorithm for robust permutation flowshop scheduling". In: *The International Journal of Advanced Manufacturing Technology* 56.1-4, pp. 345–354.

Liu, W.-Y., C.-C. Lin, C.-R. Chiu, Y.-S. Tsao, and Q. Wang (2014). "Minimizing the carbon footprint for the time-dependent heterogeneous-fleet vehicle routing problem with alternative paths". In: *Sustainability* 6.7, pp. 4658–4684.

Louis, S. J. and J. McDonnell (2004). "Learning with case-injected genetic algorithms". In: *Trans. Evol. Comp* 8.4, pp. 316–328.

Louis, Sushil J (2003). "Genetic learning from experience". In: *IEEE Congress on Evolutionary Computation*. Vol. 3, pp. 2118–2125.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

Lu, Y., N. Zeng, X. Liu, and S. Yi (2015). "A new hybrid algorithm for bankruptcy prediction using switching particle swarm optimization and support vector machines". In: *Discrete Dynamics in Nature and Society* 2015.

Lwin, K., R. Qu, and G. Kendall (2014). "A learning-guided multi-objective evolutionary algorithm for constrained portfolio optimization". In: *Applied Soft Computing* 24, pp. 757–772.

Malakahmad, A., P. Md. Bakri, M. R. Md. Mokhtar, and N. Khalil (2014). "Solid waste collection routes optimization via GIS techniques in Ipoh City, Malaysia". In: *Procedia Engineering* Vol. 77, pp. 20–27.

Maniezzo, V., T. Stützle, and S. Vo (2009). *Matheuristics: Hybridizing metaheuristics and mathematical programming*. 1st. Springer Publishing Company, Incorporated.

Mansini, R., W. Ogryczak, and M. G. Speranza (2014). "Twenty years of linear programming based portfolio optimization". In: *European Journal of Operational Research* 234.2, pp. 518–535.

Marim, L. R., M. R. Lemes, and A. D. Pino (2003). "Neural-network-assisted genetic algorithm applied to silicon clusters". In: *Phys. Rev. A* 67.3.

Marinaki, M., Y. Marinakis, and C. Zopounidis (2010). "Honey bees mating optimization algorithm for financial classification problems". In: *Applied Soft Computing* 10.3, pp. 806–812.

Marinakis, Y., M. Marinaki, and N. Matsatsinis (2008). "A stochastic nature inspired metaheuristic for clustering analysis". In: *International Journal of Business Intelligence and Data Mining* 3.1, pp. 30–44.

Maringer, D. and H. Kellerer (2003). "Optimization of cardinality constrained portfolios with a hybrid local search algorithm". In: *OR Spectrum* 25.4, pp. 481–495.

Maringer, D. and O. Oyewumi (2007). "Index tracking with constrained portfolios". In: *Intelligent Systems in Accounting, Finance and Management* 15.1-2, pp. 57–71.

Markov, I., S. Varone, and M. Bierlaire (2016). "Integrating a heterogeneous fixed fleet and a flexible assignment of destination depots in the waste collection VRP with intermediate facilities". In: *Transportation Research Part B: Methodological* 84, pp. 256–273.

Martí, R., M. G. C. Resende, and C. Ribeiro (2013). "Multi-start methods for combinatorial optimization". In: *European Journal of Operational Research* 226.1, pp. 1–8.

Martin, O., S. W. Otto, and E. W. Felten (1992). "Large-step Markov chains for the TSP incorporating local search heuristics". In: *Oper. Res. Lett.* 11.4, pp. 219–224.

Martin, S., D. Ouelhadj, P. Beullens, E. Ozcan, A. A. Juan, and E.K. Burke (2016). "A Multi-Agent Based Cooperative Approach To Scheduling and Routing". In: *Eur. J. Oper. Res.* 254.1, pp. 169–178.

Mazza, D., A. Pagès-Bernaus, D. Tarchi, A. A. Juan, and G. E. Corazza (2016). "Supporting mobile cloud computing in smart cities via randomized algorithms". In: *IEEE Systems Journal*.

McQueen, G. and V. V. Roley (1993). "Stock prices, news, and business conditions". In: *Review of Financial Studies* 6.3, pp. 683–707.

Mendes, L., P. Godinho, and J. Dias (2012). "A Forex trading system based on a genetic algorithm". In: *Journal of Heuristics* 18.4, pp. 627–656.

Metaxiotis, K. and K. Liagkouras (2012). "Multiobjective evolutionary algorithms for portfolio management: a comprehensive literature review". In: *Expert Systems with Applications* 39.14, pp. 11685–11698.

Metropolis, N., A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller (1953). "Equation of state calculations by fast computing machines". In: *Journal of Chemical Physics* 21, pp. 1087–1092.

Michalski, R. S. (2000). "Learnable evolution model: Evolutionary processes guided by machine learning". In: *Mach. Learn.* 38.1-2, pp. 9–40.

Min, S.-H., J. Lee, and I. Han (2006). "Hybrid genetic algorithms and support vector machines for bankruptcy prediction". In: *Expert systems with applications* 31.3, pp. 652–660.

Mirabi, M., S.M.T. Fatemi-Ghomi, and F. Jolai (2010). "Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem". In: *Robotics and Computer-Integrated Manufacturing* 26.6, pp. 564–569.

Mishra, S. K., G. Panda, and R. Majhi (2014). "Constrained portfolio asset selection using multiobjective bacteria foraging optimization". In: *Operational Research* 14.1, pp. 113–145.

Mladenovic, N. (1995). "A variable neighborhood algorithm: A new metaheuristic for combinatorial optimization". In: *Abstracts of papers presented at Optimization Days*, p. 112.

Mladenović, N. and P. Hansen (1997). "Variable neighborhood search". In: *Computers & Operations Research* 24.11, pp. 1097–1100.

Moghaddam, B. F., R. Ruiz, and S. J. Sadjadi (2012). "Vehicle routing problem with uncertain demands: An advanced particle swarm algorithm". In: *Computers & Industrial Engineering* 62, pp. 306–317.

Molina, J., M. Laguna, Rafael. Martí, and R. Caballero (2007). "SSPMO: A scatter tabu search procedure for non-linear multiobjective optimization". In: *INFORMS Journal on Computing* 19.1, pp. 91–100.

Montero, E., M.-C. Riff, and B. Neveu (2014). "A beginner's guide to tuning methods". In: *Applied Soft Computing* 17, pp. 39–51.

Montoya-Torres, J., J. Lopez, S. Nieto, H. Felizzola, and N. Herazo-Padilla (2015). "A literature review on the vehicle routing problem with multiple depots". In: *Computers & Industrial Engineering* 79, pp. 115–129.

Moon, C., J. Kim, and S. Hur (2002). "Integrated process planning and scheduling with minimizing total tardiness in multi-plants supply chain". In: *Computers & Industrial Engineering* 43.1, pp. 331–349.

Moral-Escudero, R., R. Ruiz-Torrubiano, and A. Suárez (2006). "Selection of optimal investment portfolios with cardinality constraints". In: *2006 IEEE International Conference on Evolutionary Computation*. IEEE, pp. 2382–2388.

Moreno-Paredes, J. C., C. Mues, and L. C. Thomas (2013). "Credit scoring and credit control XIII Conference Proceedings". In: chap. A multi-objective decision framework for credit portfolio management.

Moreno-Vega, J. M. and B. Melián (2008). "Introduction to the special issue on variable neighborhood search". In: *Journal of Heuristics* 14.5, p. 403.

Muth, J. F. and G. L. Thompson (1963). *Industrial scheduling*. Prentice-Hall.

Myszkowski, P. B. and A. Bicz (2010). "Evolutionary algorithm in Forex trade strategy generation". In: *Computer Science and Information Technology (IMCSIT), Proceedings of the 2010 International Multiconference on*, pp. 81–88.

Naderi, B. and R. Ruiz (2010). "The distributed permutation flowshop scheduling problem". In: *Computers & Operations Research* 37.4, pp. 754–768.

— (2014). "A scatter search algorithm for the distributed permutation flowshop scheduling problem". In: *European Journal of Operational Research* 239.2, pp. 323–334.

Nance, R. E. and R. G. Sargent (2002). "Perspectives on the evolution of simulation". In: *Operations Research* 50.1, pp. 161–172.

Nawaz, M., J. Enscore, and I. Ham (1983). "A heuristic algorithm for the m-machine, n-job flowshop sequencing problem". In: *Omega* 11, 91—95.

Neirotti, P., A. De Marco, A. C. Cagliano, G. Mangano, and F. Scorrano (2014). "Current trends in smart city initiatives: some stylised facts". In: *Cities* 38, pp. 25–36.

Ni, H. and Y. Wang (2013). "Stock index tracking by Pareto efficient genetic algorithm". In: *Applied Soft Computing* 13.12, pp. 4519–4535.

Niknamfar, A. H. and S. T. A. Niaki (2016). "Fair profit contract for a carrier collaboration framework in a green hub network under soft time-windows: Dual lexicographic max–min approach". In: *Transportation Research Part E: Logistics and Transportation Review* 91, pp. 129–151.

Nikolaev, A. G. and S. H. Jacobson (2010). "Simulated annealing". In: *Handbook of metaheuristics*. Springer, pp. 1–39.

Nolz, P., N. Absi, and D. Feillet (2014). "A stochastic inventory routing problem for infectious medical waste collection". In: *Networks* 63, pp. 82–95.

Nunez-Letamendia, L. (2007). "Fitting the control parameters of a genetic algorithm: An application to technical trading systems design". In: *European journal of operational research* 179.3, pp. 847–868.

Nuortio, T., J. Kytöjoki, H. Niska, and O. Bräysy (2006). "Improved route planning and scheduling of waste collection and transport". In: *Expert Systems with Applications* 30, pp. 223–232.

Ombuki-Berman, B. M., A. Runka, and F. T. Hanshar (2007). "Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms". In: *Proceedings of the third IASTED International Conference on Computational Intelligence*. Anaheim, US, pp. 91–97.

Oreski, S. and G. Oreski (2014). "Genetic algorithm-based heuristic for feature selection in credit risk assessment". In: *Expert systems with applications* 41.4, pp. 2052–2064.

Oreski, S., D. Oreski, and G. Oreski (2012). "Hybrid system with genetic algorithm and artificial neural networks and its application to retail credit risk assessment". In: *Expert systems with applications* 39.16, pp. 12605–12617.

Pages, A., H. Ramalhinho, A. Juan, and L. Calvet (submitted). "Designing E-commerce supply chains: A stochastic facility-location approach". In: *International Transactions in Operational Research*. DOI: 10.1111/itor.12433.

Pathak, B. K., S. Srivastava, and K. Srivastava (2008). "Neural network embedded multiobjective genetic algorithm to solve non-linear time-cost tradeoff problems of project scheduling". In: *Journal of scientific and industrial research* 67.2, pp. 124–131.

Pavón, R., F. Díaz, R. Laza, and V. Luzón (2009). "Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study". In: *Expert Systems With Applications* 36.2, pp. 3407–3420.

Pelikan, M., D. E. Goldberg, and F. G. Lobo (2002). "A survey of optimization by building and using probabilistic models". In: *Comput. Optim. Appl.* 21.1, pp. 5–20.

Pereira, I., A. Madureira, P. B. M. Oliveira, and A. Abraham (2013). "Tuning meta-heuristics using multi-agent learning in a scheduling system". In: *Transactions on Computational Science XXI*. Springer Berlin Heidelberg, pp. 190–210.

Pisinger, D. and S. Ropke (2007). "A general heuristic for vehicle routing problems". In: *Computers and Operations Research* 34.8, pp. 2403–2435.

Polya, G. (1945). *How to solve it: a new aspect of mathematical model*. Tech. rep. Princeton University Press Princeton.

Pongcharoen, P., W. Chainate, and P. Thapatsuwan (2007). "Exploration of genetic parameters and operators through travelling salesman problem". In: *Science Asia* 33.2, pp. 215–222.

Prasanna, S. and D. Ezhilmaran (2015). "Stock market prediction using clustering with meta-heuristic approaches". In: *Gazi University Journal of Science* 28.3, pp. 395–403.

Pukthuanthong, K. and R. Roll (2011). "Gold and the Dollar (and the Euro, Pound, and Yen)". In: *Journal of Banking and Finance* 35.8, pp. 2070–2083.

Quintero-Araujo, C. L., J. P. Caballero-Villalobos, A. A. Juan, and J. R. Montoya-Torres (2016). "A biased-randomized metaheuristic for the capacitated location routing problem". In: *International Transactions in Operational Research*.

Rabbani, M., M. A. Bajestani, and G. B. Khoshkhou (2010). "A multi-objective particle swarm optimization for project selection problem". In: *Expert Systems with Applications* 37.1, pp. 315–321.

Rachev, S. T., B. Racheva-Iotova, S. V. Stoyanov, and F. J. Fabozzi (2010). "Risk management and portfolio optimization for volatile markets". In: *Handbook of Portfolio Construction*, pp. 493–508.

Raft, O.M. (1982). "A modular algorithm for an extended vehicle scheduling problem". In: *European Journal of Operational Research* 11, pp. 67–76.

Ramos, I. C. O., M. C. Goldbarg, E. G. Goldbarg, and A. D. D. Neto (2005). "Logistic regression for parameter tuning on an evolutionary algorithm". In: *Evolutionary Computation, 2005. The 2005 IEEE Congress on*. Vol. 2. IEEE, pp. 1061–1068.

Ramos, T. R. P., M. I. Gomes, and A. P. B.arbosa-Póvoa (2014). "Economic and environmental concerns in planning recyclable waste collection systems". In: *Transportation Research Part E: Logistics and Transportation Review* 62, pp. 34–54.

Ramsey, C. L. and J. J. Grefenstette (1993). "Case-based initialization of genetic algorithms". In: *Proceedings of the 5th International Conference on Genetic Algorithms*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 84–91.

Rasheed, K. and H. Hirsh (2000). "Informed operators: speeding up genetic-algorithm-based design optimization using reduced models". In: *Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation*. Morgan Kaufmann Publishers Inc., pp. 628–635.

Reboredo, J. C. (2013). "Is gold a safe haven or a hedge for the US dollar? Implications for risk management". In: *Journal of Banking & Finance* 37.8, pp. 2665–2676.

Regis, R. G. (2014). "Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions". In: *IEEE Transactions on Evolutionary Computation* 18.3, pp. 326–347.

Reid, S. G. and K. M. Malan (2015). "Constraint handling methods for portfolio optimization using particle swarm optimization". In: *Computational Intelligence, 2015 IEEE Symposium Series on*, pp. 1766–1773.

Renaud, J., G. Laporte, and F.F. Boctor (1996). "A tabu search heuristic for the multi-depot vehicle routing problem". In: *Computers & Operations Research* 23.3, pp. 229–235.

Reyes, L., L. Calvet, A. A. Juan, and J. Faulin (submitted). "Sustainable urban freight transport considering multiple capacitated depots". In: *A*.

Ribas, I., R. Companys, and X. Tort-Martorell (2017). "Efficient heuristics for the parallel blocking flow shop scheduling problem". In: *Expert Systems with Applications* 74, pp. 41–54.

Ribeiro, M. H., A. Plastino, and S. L. Martins (2006). "Hybridization of GRASP metaheuristic with data mining techniques". In: *Journal of Mathematical Modelling and Algorithms* 5.1, pp. 23–41.

Rice, J. R. (1976). "The algorithm selection problem". In: *Adv. Comput.* 15, pp. 65–118.

Ridge, E. and D. Kudenko (2007). "Analyzing heuristic performance with response surface models: prediction, optimization and robustness". In: *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, pp. 150–157.

Ries, J. (2009). "Instance-based flexible parameter tuning for meta-heuristics using fuzzy-logic". PhD thesis. University of Portsmouth.

Ries, J., P. Beullens, and D. Salt (2012). "Instance-specific multi-objective parameter tuning based on fuzzy logic". In: *European Journal of Operational Research* 218.2, pp. 305–315.

Rodammer, F. A. and K. P. White (1988). "A recent survey of production scheduling". In: *IEEE Transactions on Systems, Man and Cybernetics* 18.6, pp. 841–851.

Rousseeuw, P. J. (1987). "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis". In: *Journal of computational and applied mathematics* 20, pp. 53–65.

Roy, B. (2010). "Robustness in operational research and decision aiding: a multi-faceted issue". In: *European Journal of Operational Research* 200.3, pp. 629–638.

Ruiz, R. and T. Stützle (2007). "A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem". In: *European Journal of Operational Research* 177.3, pp. 2033–2049.

Ruiz, X., L. Calvet, J. Ferrarons, and A. A. Juan (2015). "SmartMonkey: a web browser tool for solving combinatorial optimization problems in real time". In: *Proceedings of the 2015 Int. Conf. of the Forum for Interdisciplinary Mathematics*. Barcelona, Spain.

Ruiz-Torrubiano, R. and A. Suárez (2009). "A hybrid optimization approach to index tracking". In: *Annals of Operations Research* 166.1, pp. 57–71.

Salhi, S. and M. Sari (1997). "A multi-level composite heuristic for the multi-depot vehicle fleet mix problem". In: *European Journal of Operational Research* 103, pp. 95–112.

Santos, H. G., L. S. Ochi, E. H. Marinho, and L. M. A. Drummond (2006). "Combining an evolutionary algorithm with data mining to solve a single-vehicle routing problem". In: *Neurocomputing* 70.1, pp. 70–77.

Santos, L. F., M. H. Ribeiro, A. Plastino, and S. L. Martins (2005). "A hybrid GRASP with data mining for the maximum diversity problem". In: *International Workshop on Hybrid Metaheuristics*. Springer, pp. 116–127.

Santos, L. F., S. L. Martins, and A. Plastino (2008). "Applications of the DM-GRASP heuristic: a survey". In: *International Transactions in Operational Research* 15.4, pp. 387–416.

Schaerf, A. (2002). “Local search techniques for constrained portfolio selection problems”. In: *Computational Economics* 20.3, pp. 177–190.

Scozzari, A., F. Tardella, S. Paterlini, and T. Krink (2013). “Exact and heuristic approaches for the index tracking problem with UCITS constraints”. In: *Annals of Operations Research* 205.1, pp. 235–250.

Seidgar, H., M. Kiani, M. Abedi, and H. Fazlollahtabar (2014). “An efficient imperialist competitive algorithm for scheduling in the two-stage assembly flow shop problem”. In: *International Journal of Production Research* 52.4, pp. 1240–1256.

Senjyu, T., A. Y. Saber, T. Miyagi, K. Shimabukuro, N. Urasaki, and T. Funabashi (2005). “Fast technique for unit commitment by genetic algorithm based on unit clustering”. In: *IEE Proceedings-Generation, Transmission and Distribution* 152.5, pp. 705–713.

Shelokar, P. S., V. K. Jayaraman, and B. D. Kulkarni (2004). “An ant colony approach for clustering”. In: *Anal. Chim. Acta* 509.2, pp. 187–195.

Shin, K.-S. and Y.-J. Lee (2002). “A genetic algorithm application in bankruptcy prediction modeling”. In: *Expert Systems with Applications* 23.3, pp. 321–328.

Silvennoinen, A. and S. Thorp (2013). “Financialization, crisis and commodity correlation dynamics”. In: *Journal of International Financial Markets, Institutions and Money* 24, pp. 42–65.

Sluga, A., P. Butala, and G. Bervar (1998). “A multi-agent approach to process planning and fabrication in distributed manufacturing”. In: *Computers & Industrial Engineering* 35.3, pp. 455–458.

Smith, J. E. (2008). “Self-adaptation in evolutionary algorithms for combinatorial optimisation”. In: *Adaptive and Multilevel Metaheuristics*. Springer, pp. 31–57.

Smith-Miles, K., D. Baatar, B. Wreford, and R. Lewis (2014). “Towards objective measures of algorithm performance across instance space”. In: *Comput. Oper. Res.* 45, pp. 12–24.

Smith-Miles, K. A. (2009). “Cross-disciplinary perspectives on meta-learning for algorithm selection”. In: *ACM Computing Surveys* 41.1, p. 6.

Soleimani, H., H. R. Golmakani, and M. H. Salimi (2009). “Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm”. In: *Expert Systems with Applications* 36.3, pp. 5058–5063.

Solomon, M. M. (1987). “Algorithms for the vehicle routing and scheduling problems with time windows”. In: *Operations Research* Vol. 35, pp. 254–265.

Son, L. H. (2014). “Optimizing municipal solid waste collection using chaotic particle swarm optimization in GIS based environments: A case study at Danang city, Vietnam”. In: *Expert Systems with Applications* 41, pp. 8062–8074.

Sörensen, K. (2015). “Metaheuristics—the metaphor exposed”. In: *International Transactions in Operational Research* 22.1, pp. 3–18.

Stanley, K. O. and R. Miikkulainen (2002). “Evolving neural networks through augmenting topologies”. In: *Evol. Comput.* 10.2, pp. 99–127.

Stewart, Jr. W. R. and B.L. Golden (1983). “Stochastic vehicle routing: A comprehensive approach”. In: *European Journal of Operational Research* 14, pp. 371–385.

Streichert, F., H. Ulmer, and A. Zell (2003). “Evolutionary algorithms and the cardinality constrained portfolio selection problem”. In: *Operations Research Proceedings 2003, Selected Papers of the International Conference on Operations Research (OR 2003)*.

Stummer, C. and M. Sun (2005). “New multiobjective metaheuristic solution procedures for capital investment planning”. In: *Journal of Heuristics* 11.3, pp. 183–199.

Suman, B. and P. Kumar (2006). “A survey of simulated annealing as a tool for single and multiobjective optimization”. In: *The Journal of the Operational Research Society* 10.57, pp. 1143–1160.

Sung, C. S. and J. Juhn (2009). “Makespan minimization for a 2-stage assembly scheduling problem subject to component available time constraint”. In: *International Journal of Production Economics* 119.2, pp. 392–401.

Sung, C. S. and H. A. Kim (2008). “A two-stage multiple-machine assembly scheduling problem for minimizing sum of completion times”. In: *International Journal of Production Economics* 113.2, pp. 1038–1048.

Surekha, P. and S. Sumathi (2011). “Solution to MDVRP using genetic algorithms”. In: *World Applied Programming* 1, pp. 118–131.

Taillard, E. (1990). "Some efficient heuristic methods for the flow shop sequencing problem". In: *European Journal of Operational Research* 47.1, pp. 65–74.

— (1993). "Benchmarks for basic scheduling problems". In: *European Journal of Operational Research* 64, 278—285.

Talbi, E.-G. (2006). *Parallel combinatorial optimization*. Vol. 58. John Wiley & Sons.

— (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

— (2013). "Combining metaheuristics with mathematical programming, constraint programming and machine learning". In: *4OR* 11.2, pp. 101–150.

Tan, K. C., C. Y. Cheong, and C. K. Goh (2007). "Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation". In: *Discrete optimization* 177, pp. 813–839.

Tatarakis, A. and I. Minis (2009). "Stochastic single vehicle routing with a predefined customer sequence and multiple depot returns". In: *European Journal of Operational Research* 197, pp. 557–571.

Tauhid, S., Z. Jannat, and F. Mahmud (2012). "A novel three-phase approach for solving multi-depot vehicle routing problem with stochastic demand". In: *Algorithms Research* 1, pp. 15–19.

Tavares, G., Z. Zsigraiova, V. Semiao, and M. Carvalho (2009). "Optimisation of MSW collection routes for minimum fuel consumption using 3D GIS modelling". In: *Waste Management* 29, pp. 1176–1185.

Team, R Development Core (2008). *R: a language and environment for statistical computing*. R Foundation for Statistical Computing.

Teixeira, J., A. P. Antunes, and J. P. de Sousa (2004). "Recyclable waste collection planning-a case study". In: *European Journal of Operational Research* 158, pp. 543–554.

Teixeira, L. A. and A. L. I. De Oliveira (2010). "A method for automatic stock trading combining technical analysis and nearest neighbor classification". In: *Expert systems with applications* 37.10, pp. 6885–6890.

Tenne, Y. and C.-K. Goh (2010). *Computational intelligence in expensive optimization problems*. Vol. 2. Springer Science & Business Media.

Thangiah, S. R. and S. Salhi (2001). "Genetic clustering: an adaptive heuristic for the multi-depot vehicle routing problem". In: *Applied Artificial Intelligence* 15.4, pp. 361–383.

Theodoridis, S. and K. Koutroumbas (2009). *Pattern recognition*. Vol. 74. John Wiley & Sons.

Thomaidis, N. S. (2011). "A soft computing approach to enhanced indexation". In: *Natural Computing in Computational Finance*. Springer, pp. 61–77.

Tillman, F. A. (1969). "The multiple terminal delivery problem with probabilistic demands". In: *Transportation Science* 3, pp. 192–204.

Tillman, F. A. and T. M. Cain (1972). "An upper bound algorithm for the single and multiple terminal delivery problem". In: *Management Science* 18.11, pp. 664–682.

Tuba, M. and N. Bacanin (2014a). "Artificial bee colony algorithm hybridized with firefly algorithm for cardinality constrained mean-variance portfolio selection problem". In: *Appl. Math. Inf. Sci* 8.6, pp. 2831–2844.

— (2014b). "Upgraded firefly algorithm for portfolio optimization problem". In: *Computer Modelling and Simulation (UKSim), 2014 UKSim-AMSS 16th International Conference on*. IEEE, pp. 113–118.

Tyasnurita, R., E. Ozcan, S. Asta, and R. John (2015). "Improving Performance of a Hyper-heuristic Using a Multilayer Perceptron for Vehicle Routing". In: *15th Annual Workshop on Computational Intelligence*. Springer. Lancaster, UK.

Ubeda, S., F.J. Arcelus, and J. Faulin (2011). "Green logistics at Eroski: A case study". In: *International Journal of Production Economics* 131.1, pp. 44–51.

Urli, B. and F. Terrien (2010). "Project portfolio selection model, a realistic approach". In: *International Transactions in Operational Research* 17.6, pp. 809–826.

Vairaktarakis, G. and M. Elhafsi (2000). "The use of flowlines to simplify routing complexity in two-stage flowshops". In: *IIE Transactions (Institute of Industrial Engineers)* 32.8, pp. 687–699.

Viana, A., J. P. Sousa, and M. A. Matos (2005). "Constraint oriented neighbourhoods – A new search strategy in metaheuristics". In: *Metaheuristics: progress as real problem solvers*. Springer, pp. 389–414.

Vidal, T., T G Crainic, M. Gendreau, N. Lahrichi, and W. Rei (2012). "A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems". In: *Operations Research* 60, pp. 611–624.

Wang, B. (1997). *Integrated product, process and enterprise design*. London. ISBN: 0412620200.

Wang, F., L. H. Philip, and D. W. Cheung (2014a). "Combining technical trading rules using particle swarm optimization". In: *Expert Systems with Applications* 41.6, pp. 3016–3026.

Wang, J., K. Guo, and S. Wang (2010). "Rough set and tabu search based feature selection for credit scoring". In: *Procedia Computer Science* 1.1, pp. 2425–2432.

Wang, J., A.-R. Hedar, S. Wang, and J. Ma (2012). "Rough set and scatter search metaheuristic based feature selection for credit scoring". In: *Expert Systems with Applications* 39.6, pp. 6123–6128.

Wang, J., K. Tang, J. Lozano, and X. Yao (2015). "Estimation of Distribution Algorithm with Stochastic Local Search for Uncertain Capacitated Arc Routing Problems". In: *IEEE T. Evolut. Comput.* 20.c, pp. 1–1.

Wang, L., W. Shen, and Q. Hao (2006). "An overview of distributed process planning and its integration with scheduling". In: *International Journal of Computer Applications in Technology* 26.1/2, p. 3.

Wang, X., H. Kopfer, and M. Gendreau (2014b). "Operational transportation planning of freight forwarding companies in horizontal coalitions". In: *European Journal of Operational Research* 237.3, pp. 1133–1141.

Wang, X., T. Fan, W. Li, R. Yu, D. Bullock, B. Wu, and P. Tremont (2016). "Speed variation during peak and off-peak hours on urban arterials in Shanghai". In: *Transportation Research Part C: Emerging Technologies* 67, pp. 84–94.

Wang, Y. and I.H. Witten (1996). *Induction of model trees for predicting continuous classes*. Tech. rep. Working paper 96/23. Hamilton, New Zealand: University of Waikato.

Woodside-Oriakhi, M., C. Lucas, and J. E. Beasley (2011). "Heuristic algorithms for the cardinality constrained efficient frontier". In: *European Journal of Operational Research* 213.3, pp. 538–550.

Wu, T.-H., C. Low, and J.-W. Bai (2002). "Heuristic solutions to multi-depot location-routing problems". In: *Computers & Operations Research* 29.10, pp. 1393–1415.

Xiao, Y. and A. Konak (2015). "Green vehicle routing problem with time-varying traffic congestion". In: *Proceedings of the 14th INFORMS Computing Society Conference*, pp. 134–148.

Xu, J., S. Y. Chiu, and F. Glover (1998). "Fine-tuning a tabu search algorithm with statistical tests". In: *International Transactions in Operational Research* 5.3, pp. 233–244.

Yalcinoz, T. and H. Altun (2001). "Power economic dispatch using a hybrid genetic algorithm". In: *IEEE Power Engineering Review* 21.3, pp. 59–60.

Yang, S., Q. H. Liu, J. Lu, S. L. Ho, G. Ni, P. Ni, and S. Xiong (2009). "Application of support vector machines to accelerate the solution speed of metaheuristic algorithms". In: *IEEE T. Magn.* 45.3, pp. 1502–1505.

Yang, W. H., K. Mathur, and R. H. Ballou (2000). "Stochastic vehicle routing problem with restocking". In: *Transportation Science* 34, pp. 99–112.

Yao, X. (1999). "Evolving artificial neural networks". In: *Proceedings of the IEEE* 87.9, pp. 1423–1447.

Yoo, S.-H. and S.-B. Cho (2004). "Partially evaluated genetic algorithm based on fuzzy c-means algorithm". In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 440–449.

You, L. and R. T. Daigler (2012). "A Markowitz optimization of commodity futures portfolios". In: *The Journal of Futures Markets* 33.4, pp. 343–368.

Zennaki, M. and A. Ech-Cherif (2010). "A new machine learning based approach for tuning metaheuristics for the solution of hard combinatorial Optimization problems". In: *Journal of Applied Sciences* 10, pp. 1991–2000.

Zhang, H. and R. Ren (2010). "High frequency foreign exchange trading strategies based on genetic algorithms". In: *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*. Vol. 2, pp. 426–429.

Zhang, J., Z.-H. Zhang, Y. Lin, N. Chen, Y.-J. Gong, J.-H. Zhong, H. Chung, Y. Li, and Y.-H. Shi (2011). "Evolutionary Computation Meets Machine Learning: A Survey". In: *Comp. Intell. Mag.* 6.4, pp. 68–75.

Zhang, J., Y. Zhao, W. Xue, and J. Li (2015). "Vehicle routing problem with fuel consumption and carbon emission". In: *International Journal of Production Economics* 170, pp. 234–242.

Zhang, S., C. K. M. Lee, K. Wu, and K. L. Choy (2016). "Multi-objective optimization for sustainable supply chain network design considering multiple distribution channels". In: *Expert Systems with Applications* 65, pp. 87–99.

Zhang, X. and S. Van De Velde (2012). "Approximation algorithms for the parallel flow shop problem". In: *European Journal of Operational Research* 216.3, pp. 544–552.

Zhou, A. and Q. Zhang (2010). "A surrogate-assisted evolutionary algorithm for minimax optimization". In: *IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–7.

Zhou, Q. and X. Cui (2008). "Research on multiobjective flow shop scheduling with stochastic processing times and machine breakdowns". In: *IEEE International Conference on Service Operations and Logistics, and Informatics*. Vol. 2. IEEE, pp. 1718–1724.

Zhou, Z., Y. S. Ong, M. H. Nguyen, and D. Lim (2005). "A study on polynomial regression and gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm". In: *IEEE Congress on Evolutionary Computation*. Vol. 3. IEEE, pp. 2832–2839.

Zhu, H., Y. Wang, K. Wang, and Y. Chen (2011). "Particle swarm optimization for the constrained portfolio optimization problem". In: *Expert Systems with Applications* 38.8, pp. 10161–10169.