Computational Intelligence Methods and Applications

Christian Blum

Construct, Merge, Solve & Adapt

A Hybrid Metaheuristic for Combinatorial Optimization

Springer

Christian Blum

Construct, Merge, Solve & Adapt

A Hybrid Metaheuristic for Combinatorial Optimization

Springer

Christian Blum
IIIA-CSIC
Bellaterra, Spain

ISSN 2510-1765 ISSN 2510-1773 (electronic)
Computational Intelligence Methods and Applications
ISBN 978-3-031-60102-6 ISBN 978-3-031-60103-3 (eBook)
https://doi.org/10.1007/978-3-031-60103-3

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

This book is lovingly dedicated to my mother, Maria Blum (1949–2020), whose unwavering emotional support and belief in me have empowered me to confront life's challenges with optimism and resilience.

Preface

The work on CMSA started in 2015 during my years as an Ikerbasque Research Fellow at the University of the Basque Country in San Sebastian. It originated from the observation that large neighborhood search (LNS) algorithms based on the partial destruction of an incumbent solution at each iteration sometimes underperform in the context of optimization problems in which solutions contain rather few solution components. (As an example, think about a multi-dimensional knapsack problem instance with very tight resource constraints.) Our intention, in the context of the Ph.D. thesis of Pedro Pinacho Davidson, was then to develop an alternative hybrid algorithm that would work well in those cases in which LNS showed problems. In the meanwhile, two other Ph.D. students from my group at the IIIA-CSIC in Bellaterra (Barcelona), Mehmet Anıl Akbay and Jaume Reixach, have been working on different aspects of CMSA. Moreover, the initial paper on CMSA (published under the title "Construct, merge, solve & adapt: A new general algorithm for combinatorial optimization", which was published in 2016 in the journal *Computers and Operations Research*, has received 106 citations (Google Scholar, February 2024). Moreover, to date, CMSA has been applied to 20 different combinatorial optimization problems.

I am also happy to say that our work on CMSA has received two awards in recent years. The first one was the *best paper award* at the ECOM track of the GECCO 2016 conference for a paper on the application of CMSA to the multi-dimensional knapsack problem. The second award was the one for *The Best Methodological Contribution in Operations Research* jointly given by the Spanish Society of Statistics and Operations Research (SEIO) and the BBVA Foundation in 2021.

This book aims to give an account of the current state of the research efforts on CMSA. After shortly introducing the general line of research and the tools to be used in the book, the first chapter provides a didactical introduction to standard CMSA in the context of the minimum dominating set problem. In addition, the C++ program code used for part of the experiments presented in this chapter is offered in Appendix A. The following four chapters are dedicated to important CMSA variants (ADAPT_CMSA and LEARN_CMSA), respectively, to important topics for the practical application of CMSA: the use of set-covering-based ILP models for sub-instance solving, and the application of CMSA to optimization problems that are naturally modeled by non-binary ILPs. Finally, the last chapter outlines research lines that have not yet received much attention. Moreover, several avenues for current and future work are described. I believe that this book will be useful and inspiring for everyone who plans to apply CMSA to a specific optimization problem.

I am very grateful to the following people. The main idea of LEARN_CMSA presented in Chap. 3 of this book was contributed by Pedro Pinacho Davidson, who was my Ph.D. student at the University of the Basque Country, and who is nowadays an associate professor at the Universidad de Concepción, Chile. Moreover, Pedro prepared the initial implementation of LEARN_CMSA for the FFMS problem. The idea of ADAPT_CMSA presented in Chap. 2 was developed together with Mehmet Anıl Akbay, who was my Ph.D. student at the time of preparing this book. The same holds for the use of set-covering-based ILP models for sub-instance solving in the context of the EVRP-TW-SPD problem in Chap. 4. Mehmet provided the CMSA implementation for the EVRP-TW-SPD problem. Moreover, some of the text in this chapter was written based on his original texts. I am also grateful to Camilo Chacón Sartori, one of my latest Ph.D. students, who implemented the web application STNWeb for the generation of the nice and informative STN graphics presented in this book. Last but not least, thanks to Guillem Rodríguez, Jaume Reixach, and Camilo Chacón for proofreading (parts of) the book. Many thanks to all of you!

To end, promising research remains to be done in the context of the CMSA algorithm. Together with the optimization group at the IIIA-CSIC in Bellaterra (Barcelona), I will take on this endeavor during the coming years. We certainly hope that other research groups on metaheuristics and their hybrids will join this effort in the quest for increasingly efficient CMSA variants.

Sant Esteve Sesrovires, Spain Christian Blum
February 2024

Acknowledgments

During the time of writing this book, the author was supported as principal investigator of the following three grants funded by MCIN/AEI/10.13039/501100011033: PID2019-104156GB-I00, PID2022-136787NB-I00, and TED2021-129319B-I00. Moreover, the author would like to acknowledge the support of the Spanish National Research Council (CSIC) of which the author is a Senior Research Scientist.

Contents

Acronyms

ACO	Ant Colony Optimization
AI	Artificial Intelligence
BA	Bacterial Algorithm
BIP	Binary Integer Programming
BKPWC	Bounded Knapsack Problem With Conflicts
CD	Critical Difference
CMSA	Construct, Merge, Solve & Adapt
CP	Constraint Programming
DNA	Deoxyribonucleic Acid
EA	Evolutionary Algorithm
EV	Electric Vehicle
EVRP	Electric Vehicle Routing
EVRP-TW-SPD	Electric Vehicle Routing Problem with Time Windows and Simultaneous Pickups and Deliveries
FFMS	Far From Most String
IG	Iterated Greedy
ILP	Integer Linear Programming
KP	Knapsack Problem
LNS	Large Neighborhood Search
LP	Linear Programming
MaxSAT	Maximum Satisfiability Problem
ML	Machine Learning
MCSP	Minimum Common String Partition
MDKP	Multi-Dimensional Knapsack Problem
MDS	Minimum Dominating Set
MPIDS	Minimum Positive Influence Dominating Set
OR	Operations Research
PBIG	Population-Based Iterated Greedy
PSO	Particle Swarm Optimization
RFLCS	Repetition-Free Longest Common Subsequence
SA	Simulated Annealing

SPD	Simultaneous Pickup and Delivery
STN	Search Trajectory Network
TS	Tabu Search
TSP	Travelling Salesman Problem
TW	Time Window
VNS	Variable Neighborhood Search
VSBP	Variable-Sized Bin Packing
WID	Weighted Independent Domination

Chapter 1
Introduction to CMSA

Abstract Construct, Merge, Solve & Adapt (CMSA) is an award-winning, hybrid algorithm for solving hard combinatorial optimization problems. The main idea consists in the iterated application of an exact approach—such as, for example, an integer linear programming (ILP) solver—to sub-instances of the original problem instances to be solved. These sub-instances are extended at each iteration by adding solution components from a set of valid solutions that are obtained either by probabilistic solution construction or by any other means. In this first chapter, we will give an introduction to CMSA including related work and the application of basic CMSA variants to a well-known combinatorial optimization problem known as the Minimum Dominating Set (MDS) problem in undirected graphs. In addition, we will describe all the tools that are used for the experimental evaluation of the algorithms presented in this book. This includes the parameter tuning software called `irace`, an R-based tool for the statistical comparison of multiple algorithms called `scmamp`, and a web-based tool for the graphical comparison of multiple algorithms called `STNWeb`.

1.1 Introduction to Optimization

Optimization refers to the process of finding a best solution or outcome from a set of possible choices, generally to maximize or minimize a particular objective or criterion. It is a fundamental concept in various fields, including mathematics, engineering, economics, computer science, and more. In fact, in our increasingly technological world, the need for solving hard optimization problems has been growing constantly over the last decades. Optimization problems are prevalent in numerous practical applications across different fields. Examples are to be found, among others, in the following major fields. For each one, we provide an exemplifying reference.

1. **Supply Chain Optimization** [29]: Companies aim to optimize their supply chains by determining the most efficient way to source, produce, and deliver

C. Blum, *Construct, Merge, Solve & Adapt*, Computational Intelligence Methods and Applications, https://doi.org/10.1007/978-3-031-60103-3_1

goods while minimizing costs and maintaining inventory levels to meet customer demand.

2. **Portfolio Optimization** [53]: In the realm of finance, investors aim to optimize their investment portfolios by selecting the most suitable mix of assets to maximize returns while effectively mitigating risks. This entails the pursuit of an ideal asset allocation.
3. **Production Scheduling** [84]: Manufacturers want to optimize production schedules to minimize production costs, reduce lead times, and meet customer demand reliably. This requires determining the optimal allocation of resources and scheduling production runs.
4. **Transportation Routing** [101]: In logistics and transportation, companies aim to derive optimal routes and delivery schedules for delivery vehicles, ships, or airplanes to minimize fuel costs, reduce travel times, and increase delivery efficiency. Increasingly complex optimization problems must be solved in so-called electric vehicle routing problems.
5. **Project Management** [61]: Project managers seek to optimize project schedules and resource allocation to complete projects on time and within budget. Critical path analysis and resource leveling are techniques used, for example, for the optimization of project schedules.
6. **Network Design** [82]: Companies and service providers need to optimize the design and layout of their networks—such as, for example, telecommunications networks, computer networks, or transportation networks—to maximize efficiency and minimize costs.
7. **Inventory Management** [96]: Retailers and manufacturers optimize inventory levels to balance the costs of holding excess inventory against potentially lost sales due to the lack of stock. This is often achieved through so-called Economic Order Quantity (EOQ) and Just-In-Time (JIT) inventory systems.
8. **Energy Management** [75]: Organizations and companies aim to optimize energy consumption in buildings and manufacturing processes to reduce energy costs and minimize environmental impact. This involves scheduling equipment and systems to operate at their most energy-efficient levels.
9. **Agricultural Planning** [24]: Farmers and agricultural organizations optimize crop planting, irrigation, and harvesting schedules to maximize yield, minimize resource usage, and adapt to changing weather conditions.
10. **Sensor Placement** [65]: In environmental monitoring, for example, optimizing the placement of sensors or surveillance cameras to maximize coverage and minimize costs is crucial.
11. **Drug Design** [58]: In the field of drug development, several optimization problems arise, often with the goal of identifying and developing effective and safe pharmaceutical compounds. Some of these optimization problems concern (1) compound screening, (2) optimizing the molecular structure of a compound to improve its potency, selectivity, and safety, and (3) clinical trial design.
12. **Staff Rostering and Resource Allocation** [42]: The optimal assignment of personnel to shifts and the need for an allocation of resources arises in a wide range of organizations and industries. Hospitals and healthcare facilities, for

example, must optimize staff scheduling, operating room allocation, and patient appointment scheduling to improve patient care and reduce costs.

13. **Traffic Management** [10]: Cities use optimization to manage traffic flow, for example, through optimizing traffic signal timings, leading to reduced congestion and improved traffic efficiency.

Obviously, these are just examples, and optimization is used in many more fields and scenarios to improve decision-making, resource allocation, and overall efficiency. Moreover, optimization is an essential factor in many fields of research. In fact, without efficient optimization techniques, many fields of research would not be able to advance at the same speed as they are doing today.

Modelling an Optimization Problem

In order to solve an optimization problem, it must first be modeled in a mathematical, respectively technical, way. Key elements of an optimization problem model include the following ones.

1. *Objective Function:* A model consists of at least one objective function that quantifies the goal or criterion to be optimized. Such a function may represent a quantity to be maximized (e.g., profit, efficiency, performance) or minimized (e.g., cost, error, time). In the presence of exactly one objective function, we talk about single-objective optimization, while several—usually conflicting—objective functions characterize a multi-objective problem.
2. *Decision Variables:* Optimization problems involve decision variables together with their domains. Each candidate solution to a problem is characterized by a different setting (value assignment) of the decision variables.
3. *Constraints:* They define which candidate solutions (value-assignments of the decision variables) correspond to feasible solutions, in contrast to infeasible solutions. Constraints can be equality constraints (e.g., fixed budget) or inequality constraints (e.g., resource availability).
4. *Optimization Objective:* As already mentioned in the context of describing the concept of an objective function (see above), objective functions might be maximized or minimized. This is called the optimization objective.

Often the *goal of optimization* is to find an optimal solution, which is a feasible solution with an objective function value better or equal to the objective function value of all other feasible solutions. Instead, the goal of optimization might simply be to find a good enough solution in a reasonable computation time.

Optimization problems come in various forms and can be categorized into different types based on their characteristics, constraints, and objectives. In general, we distinguish between *continuous (or numerical) optimization problems* [77] and *discrete (or combinatorial) optimization problems* [80]. Hereby, continuous optimization problems refer to models in which decision variables have continuous (real-valued) domains that may be bounded or unbounded. In discrete, respectively combinatorial, optimization problems, the decision variables are restricted to domains of discrete values. In the following, we provide two examples for each of these problem categories.

1.1.1 Examples of Continuous Optimization Problems

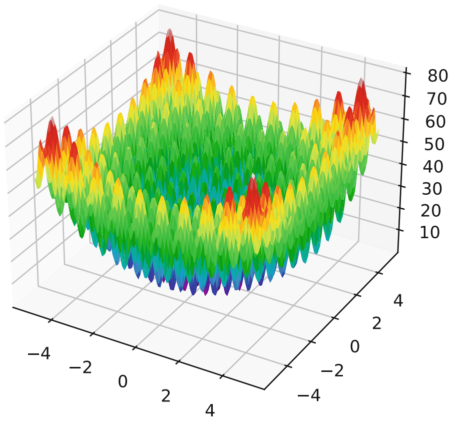
Examples of continuous (or numerical) optimization problems are analytical problems such as the minimization of the **Rastrigin function** which is plotted in Fig. 1.1 in two dimensions. The formula of this function (in two dimensions) is as follows:

$$f(x_1, x_2) = 20 + \sum_{i=1}^{2}(x_i^2 - 10 \cdot \cos(2 \cdot \pi \cdot x_i)) \ , \tag{1.1}$$

with $x_1, x_2 \in [-5.12, 5.12]$. A more practical example of a continuous optimization problem with real-world relevance is the **parameter estimation in nonlinear models** problem. This problem frequently arises in various fields, including science, engineering, economics, and biology, where researchers or analysts need to estimate the parameters of a complex—generally nonlinear—model to fit observed data. Figure 1.2 shows a graphical illustration.

Given the observed data, a model must be chosen. Subsequently, the optimization problem consists of determining the optimal values of the model's parameters in order to best fit the data. The objective is to minimize the difference between the model's predictions and the observed data, typically expressed as the sum of squared residuals (least squares). The decision variables correspond to the model's parameters that need to be estimated. Constraints are based on parameter value restrictions (e.g., bounds or relationships between parameters).

Fig. 1.1 Rastrigin function in two dimensions



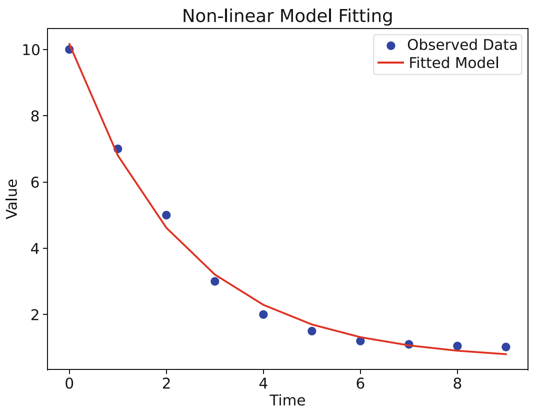


Fig. 1.2 Example of parameter estimation in nonlinear models

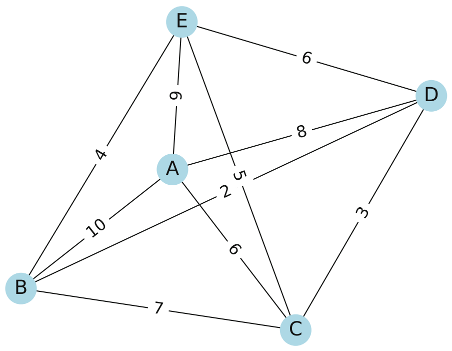


Fig. 1.3 Example of a TSP problem instance with five cities

1.1.2 Examples of Combinatorial Optimization Problems

One of the most emblematic combinatorial optimization problems is the so-called **traveling salesman problem (TSP)**. This problem owes its name to the objective of the problem. A traveling salesman must pass through a number of cities exactly once, before returning to the city in which the journey started. The optimization objective is to minimize the traveled distance. This can be modeled by means of a completely connected graph in which the nodes represent the cities that must be visited, and weights on the edges correspond to the distances between the cities. Each feasible solution corresponds to a Hamiltonian cycle of this graph. Hereby, a Hamiltonian cycle is a cyclic route that contains each vertex exactly once. A graphical illustration is given in Fig. 1.3.

Another well-known combinatorial optimization problem is the so-called **knapsack problem (KP)**. Given is a set of items, whereby each item has a profit and, for example, a weight. Given is also a knapsack with an upper limit for the total weight of the objects it can carry. The objective of the problem is to select a set of items such that they fit into the knapsack—that is, their weights may sum to at most the

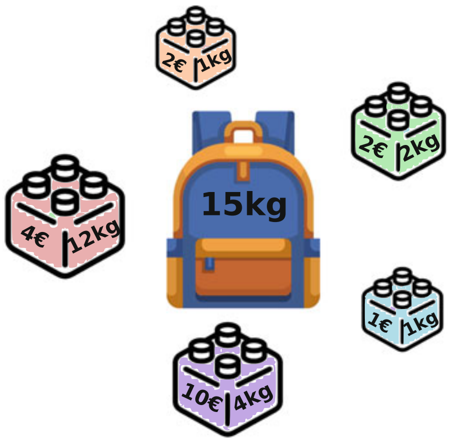


Fig. 1.4 Example of a small knapsack problem instance

weight limit of the knapsack—and the sum of the profits of the selected items is maximized. A graphical illustration is provided in Fig. 1.4.

1.1.3 Modelling an Optimization Problem

As mentioned before, to solve an optimization problem employing an optimization technique—that is, an algorithm—it must first be modeled in a way depending on its characteristics; see [33, 55, 81]. The two continuous optimization examples from Sect. 1.1.1 are modeled as global optimization problems with non-linear objective functions. In the case of a linear objective function, linear constraints, and a convex search space, a continuous optimization problem can be modeled as a linear programming (LP) problem and then be solved by LP techniques from Operations Research (OR). In contrast, the two combinatorial optimization problems outlined in Sect. 1.1.2 can be modeled as *integer linear programming (ILP)* problems, that is, in terms of models that are characterized by linear objective functions and constraints, and decision variables with discrete domains. Note that most optimization problems treated in this book are of this type. However, the general idea of CMSA is also applicable to solving optimization problems modeled in other ways.

For demonstration purposes, we provide two different ILP models of the TSP. Given is a set N of n cities, that is, $N = \{1, \dots, n\}$. Moreover, let $A = \{(i, j) \mid i, j \in N, i \neq j\}$ be the complete set of arcs connecting any ordered pair of cities. Finally, let $c_{ij} > 0$ be the distance for traveling from city i to city j. For modeling this problem, first, the following set of binary decision variables is introduced: $\{x_{ij} \in \{0, 1\} \mid i, j \in N, i \neq j\}$, that is, for each arc (i, j) we introduce a binary decision variable x_{ij}. Hereby, in case $x_{ij} = 1$, arc (i, j) forms part of the solution.