1 2 9 0

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

Pedro Miguel Duque Rodrigues

Principled Modelling Of The Google Hash Code Problems For Meta-Heuristics

Dissertation in the context of the Master in Informatics Engineering, Specialization in Intelligent Systems, advised by Professor Alexandre B. Jesus and Professor Carlos M. Fonseca, and presented to the Faculty of Sciences and Technology / Department of Informatics Engineering.

September 2023

Abstract

The Google Hash Code programming competition is an yearly event that challenges teams to solve challenging engineering problems within a limited time frame using any necessary tools. These problems, which are inspired by real-world issues and can be approached from both practical and theoretical perspectives, are of particular interest to this work. In the context of this thesis, we hope to solve some of these problems in a principled manner, with a particular focus on the modelling aspect and the clear separation between the concept of models and solvers. Additionally, there is interest in exploring the impact of this strategy on the development of general-purpose meta-heuristic solvers that can tackle these problems in a black-box fashion, making them more accessible for practitioners, researchers and developers.

Keywords

Meta-Heuristics • Modelling • Local Search • Constructive Search • Combinatorial Optimization • Intelligent Systems • Software Engineering

Funding

The work presented in this thesis was carried out in the Algorithms and Optimization Laboratory of the Adaptive Computation group of the Centre for Informatics and Systems of the University of Coimbra (CISUC) and supported by a grant with reference UIDP/00326/2020 from the Foundation for Science and Technology.

Acknowledgements

I would like to thank my supervisors for having more patience than I could ever hope for. And…. myself for completing this thesis without having a mental breakdown :).

Chapter 1

Introduction

1.1 Motivation

The Hash Code programming competition was a yearly event held by Google where teams were challenged to solve a complex problem using any tools, resources, and programming languages of their choice in just four hours. The problems were typically inspired by issues arising in engineering and real-world situations, such as vehicle routing, task scheduling, and Wi-Fi router placement. Essentially, they were posed as "open" research problems for which there exists a variety of solutions of different qualities. Notably, the great majority of these are Combinatorial Optimization (CO) problems concerning the search for the best solutions among a potentially large set of candidate solutions. Thus the adoption of efficient search strategies that take the available time budget into account is of utmost importance. Moreover, in the context of the competition, the contestants must also read, understand the problem, find a suitable representation, and write the code to solve it. As such, not all of the available time can be spent in the solution optimization stage.

A large number of optimization tools of different natures are available to practitioners for solving CO problems. In particular, there are approaches that are guaranteed to yield optimal solutions, although they are seldom appropriate in a competition context due to their poor time performance on large problems, such as the Hash Code challenges.

Albeit, for solving CO problems, a variety of algorithms exist that typically offer a compromise between the runtime and quality of the solutions found. Heuristics are a set of, often problem-specific, procedures that attempt to quickly solve a problem and provide a helpful "rule of thumb" for achieving reasonably good results, usually in a greedy fashion. A superset of these algorithms are called meta-heuristics, and contrary to regular heuristics, these are generic and can be applied to a broad range of problems. Natural processes and phenomena, such as collective behavior, natural selection, and some physical properties of materials, inspire several meta-heuristic search processes making them flexible and adaptable, although more computationally intensive than more conventional heuristics.

In the context of the Google Hash Code competition, competitors frequently make use of greedy and other heuristic strategies tailored to the challenge. Moreover, meta-heuristics in this situation are not as popular due to the time constraints imposed by the competition format. At first glance, the majority of optimization problems are structurally different from one other, which makes it demanding to write general-purpose heuristic solvers that can be easily reused. On the other hand, it might not be worth the effort spent in the implementation of such solvers, particularly as the benefit of using a simple heuristic strategy may outweigh the development and the running cost of meta-heuristic solvers such as evolutionary algorithms.

Optimization algorithms, including meta-heuristics usually follow some strategy that guides the procedure in the search for solutions. Some of the main strategies are constructive search and local search. Constructive search algorithms work by starting with an empty or partially complete solution and building a complete and feasible solution by iteratively adding components based on the solution current state and the problem constraints. In contrast, local search algorithms develop a given initial solution to the problem by introducing small changes that aim to improve it to a good local optimum and ideally a global one. A common usage of these strategies in competitions to start with a constructive search stage, in order to find good solutions quickly and then taking advantage of a local search strategy in an attempt to enhance those solutions even further.

It is evident that algorithms or solvers that utilize these search strategies are highly dependent on the specific problem they are trying to solve. The solver plays a crucial role in the problem-solving process as it is responsible for finding a solution. However, without a comprehensive model that can provide the solver with relevant information about the problem, the solver's effectiveness may be impaired.

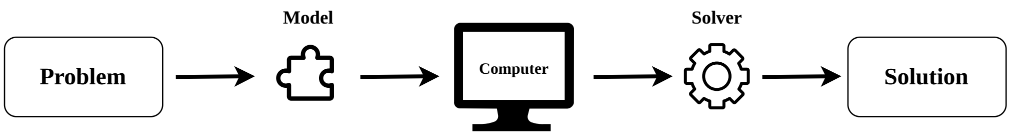

Figure 1.1: Modelling and Problem Solving

The model serves as a means of presenting the various aspects of the problem to the computer, which will subsequently utilize a solver to find a solution **??**. When constructing a model, it is important to include relevant features such as the representation of the problem and solution, a description of how components can be added or removed from the solution, and methods for evaluating the solution, calculating bounds, and obtaining other heuristic information.

A good model should encode in it all the relevant information from the problem and we must ask the right questions to obtain it, adhering to the philosophy that — "*understanding the question is half the answer*". Additionally, if a model were to be built in a standardized way, this would allow the development of a suite of generic and reusable (meta-heuristic) solvers that could find solutions in a black-box fashion. This idea has already been explored to some extent in previous work [?, ?] and will be further pursued.

It is worth noting that the modelling aspect in this field has often been neglected by the community, which has been primarily focused on the development of meta-heuristics algorithms (solvers). This discrepancy can be contrasted with the mathematical perspective, where practitioners and researchers have emphasized the importance of clearly separating the models and the algorithms (solvers), and have placed a significant emphasis on the modelling perspective. This gap highlights the importance of considering the modelling aspect in this field.

Recently, there has also been growing interest within the community in the development of optimization benchmark problems that are both relevant in practical applications and amenable to theoretical analysis. The Hash Code problems may be suitable candidates for this purpose, as they pose significant challenges from a modelling perspective while also being easy to describable. Furthermore, these problems have already been partially solved to some extent in a competitive setting, and there is a wealth of empirical data available on the best known solutions and seldomly the method used to obtain them. Overall, these factors make the Hash Code problems an ideal testing ground for both modelling and the evaluation of meta-heuristics.

1.2 Contribution

The main goal of this work is to develop and implement effective heuristic and meta-heuristic approaches for solving Hash Code problems, with a particular focus on the modelling aspect and the clear separation between solvers and models. By doing so, we hope to develop more structured and efficient problem-solving strategies that can effectively address a range of challenges. Additionally, some effort will be made to address other key areas that are crucial to the success of this work, namely:

- Development and refinement of the frameworks that separate models from solvers currently materialized in an Application Programming Interface (API) designed only for constructive search [?]. The main goal is to optimize and "fine-tune" this API in order to improve its efficacy and utility, by using the Hash Code problems as benchmarks.

- Expansion of the aforementioned API to support local search strategies in a problem-independent manner. This will allow for its application to a wider range of problems and contexts.

- Implementation of a small set of general-purpose meta-heuristic solvers and utilities that can be used to not only generate and test solutions for the various Hash Code benchmark problems, but also to verify the correctness of the results.

Last but not least, the objective of this work is to engage in a critical examination of the strengths and limitations of our adopted approach to problem modelling and solver development. This discussion will be relevant to meta-heuristic researchers, software developers, and practitioners alike. The analysis will consider various performance dimensions, including the effort required for problem modelling and solver development, the computational efficiency of the implemented software, and the quality of the solutions obtained.

1.3 Research Questions

1.4 Outline

The remainder of this document is structured as follows:

- **Chapter ??:** Provides some background on some essential aspects of optimization, search strategies, meta-heuristics, and modelling. Moreover, it presents the modelling frameworks and current state of the art of the API attempts that constitute the core fundamentals that [**?**] supports this work.
- **Chapter ??:** Gives some insight into the Google Hash Code competition and typical problems presented to contestants. Furthermore, it makes a brief categorization of all the problems from previous editions, with particular emphasis on the ones analyzed so far. Finally, it describes the modelling of the problems examined and the results obtained
- **Chapter ??:** Presents a reflection on this work.