UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITÀ DEGLI STUDI DI PADOVA

SCHOOL OF ENGINEERING DEPARTMENT OF INFORMATION ENGINEERING

MASTER DEGREE IN COMPUTER ENGINEERING

Enhancing the ACS Heuristic: Parameter Tuning and Computational Experiments for Solving MIP Problems

Supervisor:
PROF. DOMENICO SALVAGNIN

Candidate:
FRANCESCO BISCACCIA CARRARA
2120934

Academic Year 2024/2025
Date TBD

Abstract

Contents

Chapter 1

Introduction

1.1 Mixed Integer Programming (MIP)

Mixed Integer Programming (MIP) is a powerful optimization technique used to model complex real-world problems where discrete decisions are essential—such as resource allocation, scheduling, and logistics. MIP models aim to find the best solution according to a given objective function, which is either maximized or minimized. A general MIP problem can be formulated as:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b \\ & x_i \in \mathbb{Z} \quad \forall i \in I \\ & x \in \mathbb{R}^n \end{aligned}$$

Here, some decision variables are constrained to take integer values, which increases the model's complexity.

1.1.1 Heuristic in MIP solving

The presence of integer variables significantly increases the computational complexity of MIP problems, especially as the number of variables and constraints grows. Solvers such as IBM ILOG CPLEX and GUROBI incorporate a variety of built-in strategies to improve performance. Among these strategies are heuristics, which are specialized algorithms designed to quickly find feasible (but not necessarily optimal) solutions. Although heuristics do not guarantee optimality, they are computationally efficient and can significantly accelerate the solution process by providing good initial solutions or guiding the search within the solution space

1.2 Alternating Criteria Search (ACS)

Finding high-quality feasible solutions is a key aspect of the discrete optimization process, and constructing them has been one of the main focuses of research in the last few decades. Primal heuristics differ in whether they require a starting feasible solution or not. In the first case, the heuristics are called starting heuristics, and state-of-the-art techniques rely on Large Neighborhood Search (LNS) to find high-quality feasible solutions. The effectiveness of such strategies relies on the power of the MIP solver to solve the subproblems generated during the LNS heuristic. Improvement heuristics, in contrast, require a feasible starting solution and aim to improve its quality with respect to the objective. The simplest improvements are the 1-opt and 2-opt methods, but—as with starting heuristics—there are many improvement heuristics based on LNS ideas. The neighborhoods explored are defined based on branch-and-bound information—such as the best incumbent and the LP relaxation. This is essentially a limitation in terms of diversification: the strategies may reach good solutions, but it is difficult to find them in the early stages of the search. Therefore, to address this issue, the Alternating Criteria Search (ACS)—or Parallel ACS, its straightforward parallel implementation—can be utilized. The search neighborhoods are defined based on randomization instead of branch-and-bound information, allowing for a wider exploration of the search space. The aim of this strategy is to find high-quality feasible solutions at the beginning of the search, increasing the heuristic's effectiveness.

1.2.1 Parallelization of ACS

Parallelism can be exploited in MIP optimization to accelerate computation and enhance solver performance, in terms of execution time and scalability. The exploration of the branch-and-bound tree in parallel is now incorporated in most state-of-the-art MIP solvers—such as CPLEX or GUROBI—and it essentially consists of solving multiple subproblems of the original MIP simultaneously. However, this strategy may not scale well to a large number of cores. To address this limitation, the proposed heuristic—Parallel ACS (PACS)—is a parallel algorithm that integrates elements of both starting solution and improvement heuristics, offering the capability of generating starting solutions and improving them with respect to the original objective. To leverage parallelism, Parallel ACS performs a large number of LNS runs simultaneously over a diversified set of neighborhoods, with the aim of increasing the chances of finding higher-quality solutions. After each LNS exploration, the strategy consolidates the local improvements through a recombination phase. PACS combines parallelism and diversified LNS in order to address large instances of MIPs arising from various application domains.

1.2.2 Limitations of the PACS Approach

Although the experiments about the PACS strategy bring evidence about the effectiveness on hard MIP instances, there are several considerations that must be addressed. First of all, the PACS algorithm was executed on an 8-node computing cluster, each equipped with two Intel Xeon X5650 6-core processors and 24 GB of memory, totaling 96 cores and 192 GB of RAM. Such computing resources are not typically available in consumer-grade or general-purpose environments. Furthermore, as discussed in the original study, the PACS strategy could assist the MIP solver in accelerating convergence. This suggests that integrating PACS heuristics within a MIP solver could provide a meaningful contribution to solver performance. Second, before the PACS execution, an initial calibration phase is required to tune the parameters prior to execution. This undermines the objective of designing a self-contained, efficient heuristic. With these considerations in mind, the goal of this thesis is to refine the PACS strategy to make it more compatible with MIP solvers. In particular, the aim is to establish a fixed, robust parameter configuration that remains effective regardless of the input instance.