B. M. COŞAR

ATILIM UNIVERSITY 2021

NEW GREEDY ALGORITHMS TO OPTIMIZE THE CURRICULUM-BASED COURSE TIMETABLING PROBLEM

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

BATUHAN MUSTAFA COŞAR

A MASTER OF SCIENCE THESIS
IN
THE DEPARTMENT OF COMPUTER ENGINEERING

MAY 2021

NEW GREEDY ALGORITHMS TO OPTIMIZE THE CURRICULUM-BASED COURSE TIMETABLING PROBLEM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

BY

BATUHAN MUSTAFA COŞAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

MAY 2021

Approval of the Graduate School of Natural and Applied Sciences, Atilim University.

Prof.Dr. Ender Keskinkılıç
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of **Master of Science in Computer Engineering Department, Atilim University**.

Assoc.Prof.Dr. Gökhan Şengül
Head of Department

This is to certify that we have read the thesis New Greedy Algorithms to Optimize the Curriculum-based Course Timetabling Problem submitted by BATUHAN MUSTAFA COŞAR and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Tansel Dökeroğlu
Co-Supervisor

Asst. Prof. Dr. Bilge SAY
Supervisor

Examining Committee Members:

Assoc. Prof. Dr. Gökhan ŞENGÜL
Computer Eng. Dept., ATILIM University

Assoc. Prof. Dr. Tansel DÖKEROĞLU
Software Eng. Dept., TED University

Asst. Prof. Dr. Bilge SAY
Software Eng. Dept., ATILIM University

Assoc. Prof. Dr. Ender SEVİNÇ
Computer Eng. Dept., Ankara Science University

Asst. Prof. Dr. Cansu Çiğdem EKİN
Computer Eng. Dept., ATILIM University

Date: May 05, 2021

I declare and guarantee that all data, knowledge and information in this document has been obtained, processed and presented in accordance with academic rules and ethical conduct. Based on these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : BATUHAN MUSTAFA COŞAR

Signature :

ABSTRACT

New Greedy Algorithms to Optimize the Curriculum-based Course Timetabling Problem

Coşar, Batuhan Mustafa

M.S., Department of Computer Engineering

Supervisor : Asst. Prof. Dr. Bilge SAY

Co-Supervisor : Assoc. Prof. Dr. Tansel Dökeroğlu

May 2021, 88 pages

This thesis presents a set of new greedy algorithms for the optimization of the well-known ”Curriculum-Based Course Timetabling” (CB-CTT) problem, which is a sub-type of the ”Course Timetabling” problem. The main goal of the study is to minimize the total number of soft constraint violations while preserving the satisfaction of hard constraints (feasible solutions). Since the problem is NP-Hard and large instances of the problem cannot be solved in practical times, greedy algorithms that work to produce acceptable results in a few seconds are good alternatives to brute-force and evolutionary algorithms that spend hours of execution times to search for an optimal solution. Instead of using a single heuristic as it is performed by many greedy algorithms, we define and execute 120 greedy heuristics on the same problem instance simultaneously and report the overall best result, which would produce better results than which is obtainable by using a single greedy heuristic algorithm. The best results with respect to the No Free Lunch Theory, which states that the costs of greedy heuristics should be comparable on average, are reported. Our proposed greedy algorithms use the Largest-First, Smallest-First, Best-Fit, Average-weight first heuristics, and the Highest Unavailable course-first heuristics simultaneously while assigning the courses to the available rooms that are ordered by their capacity according to the

above four different criteria. In order to evaluate the performance of our proposed algorithm, we carry out experiments on 21 problem instances from the Second International Timetabling Competition (ITC-2007) benchmark set. The experimental results verify that the proposed greedy algorithms can report zero hard constraint violations (feasible solutions) for 18 problems with significantly reduced soft-constraint values.

Keywords: Course-timetabling, Greedy Algorithms, Heuristics

ÖZ

Müfredat Bazlı Ders Zamanlama Tablosu Çizelgeleme Problemi Eniyilemesi İçin Yeni Açgözlü Algoritmalar

Coşar, Batuhan Mustafa

Yüksek Lisans, Bilgisayar Mühendisliği

Tez Yöneticisi : Dr. Öğretim Üyesi Bilge SAY

Ortak Tez Yöneticisi : Doç. Dr. Tansel Dökeroğlu

Mayıs 2021, 88 sayfa

Bu tez, ”Ders Zaman Çizelgesi Oluşturma” probleminin bir alt versiyonu olarak bilinen ”Müfredata Dayalı Ders Zaman Çizelgesi Oluşturma” (CB-CTT) probleminin optimizasyonu için yeni açgözlü algoritmalar sunmaktadır. Çalışmanın temel amacı, sert kısıtlamaların (uygulanabilir çözümler) doğruluğunu korurken, yumuşak kısıt ihlallerinin toplam sayısını en aza indirmektir. Problem NP-Zor bir problem olduğundan ve büyük örneklerinin pratik zamanlarda çözülmesi için çok uzun süreler gerektirdiğinden, birkaç milisaniye içinde kabul edilebilir sonuçlar üreten açgözlü algoritmalar, arama yapmak için saatler süren eniyileme süreleri harcayan kaba kuvvet ve evrimsel algoritmalara göre daha iyi bir alternatif oluşturmaktadır. Pek çok açgözlü algoritma geliştirildi ve tek bir sezgisel yöntem kullanmak yerine, aynı problem örneğinde 120 açgözlü yöntem tanımlanıp çalıştırıldı ve daha iyi sonuçlar rapor edildi. Açgözlü algoritmaların maliyetlerinin ortalama olarak karşılaştırılabilir olması gerektiğini belirten Ücretsiz Öğle Yemeği Yok (No Free Lunch) Teorisine uygun olarak en iyi sonuçlar çalışmanın sonunda rapor edilmiştir. Önerdiğimiz açgözlü algoritmalarımız; En Büyük-Önce, En Küçük-Önce, En İyi-Uygun-Önce, Ortalama-ağırlıklı Önce sezgisel yöntemleri ve En Yüksek Kullanılamayan ders-ilk sezgisel yöntemlerini kullanarak

dersleri kapasitelerine göre sıralanan mevcut odalara atar. Önerilen algoritmamızın performansını değerlendirmek için, İkinci Uluslararası Zaman Çizelgesi Oluşturma Yarışması (ITC-2007) setinden 21 problem örneği üzerinde deneyler yapıldı. Deneysel sonuçlar, önerilen açgözlü algoritmaların, önemli ölçüde azaltılmış yumuşak kısıtlama değerleriyle sıfır sert sınırlama ihlallerini bildirebileceğini doğrulanmaktadır.

Anahtar Kelimeler: Ders çizelgesi oluşturma, Açgözlü Algoritmalar, Sezgisel Yöntemler

to my family.

ACKNOWLEDGMENTS

First of all, I would first like to thank my supervisor, Dr. Bilge Say, whose guidance has been invaluable in formulating the research questions and methodology. Her insightful comments and feedback have guided me in the right direction and greatly improved the quality of the research work and the prepared thesis.

Secondly, I would like to thank my co-supervisor, Assoc. Prof. Dr. Tansel Dökeroğlu, for communicating his experience and original directions of research for the difficult course timetabling problem. Dr. Dökeroğlu has provided his insightful comments and attack techniques to solve his difficult problem and successfully complete my thesis.

I would like to thank my thesis defense jury members for inspecting this thesis and giving their valuable comments which have greatly improved its quality. Finally, I thank all of my Professors and colleagues in the Computer Engineering and Software Engineering Departments for making me feel at home throughout my university education.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

In an academic institution, faculty members are assigned to teach one or more courses every academic term. A lecturer can teach one or more sections of one or more courses, and it is mandatory that the courses and sections taught by the same faculty member must not be assigned to the same time slot, requiring that faculty member to be in two places at the same time. There is a limited number of classrooms where a course/section can meet and the maximum number of students that can be seated in each classroom varies and it is defined by an integer. There are a fixed number of time periods during which a course can meet in a classroom on any one of the five working days of a week (Monday through Friday), and teaching is done by the lecturer assigned to teach that course. Each course/section meets on the same day and period of the week during the full academic semester. In order to facilitate the consideration that a student's courses must be on different day-periods for the student to be able to attend all of his/her courses, the concept of curriculum is introduced to the problem which forbids courses in the same curriculum to be scheduled on the same day period. Lecturers can have unavailable periods during which no lecture can be scheduled for that lecturer. The features of the course timetabling problem description above are defined as hard constraints (HC) to accommodate course scheduling constraints of students and lecturers, as well as several desirable features as soft constraints (SC).

Today, the course timetabling problem is well understood and several methods exist to find feasible solutions for it. What makes this problem an interesting research subject is the possibility of defining and maximally satisfying desirable soft constraints such as curriculum compactness, matching classroom size with number of students

as closely as possible, minimizing the number of used classrooms, minimizing walking distance between classrooms that students must cover to attend the next lecture in their curriculum, and many other such constraints that can be described. In order to be able to search for schedules that would better satisfy such soft constraints, it is necessary to quickly achieve feasible solutions, and actually since each greedy heuristic will provide a different solution, these alternative feasible solutions can be explored to better satisfy the soft constraints.

1.1 Course Timetabling as a Scheduling Problem

Scheduling occurs in several contexts in scientific literature. Scheduling processes and jobs on a computer system aims to give priority to shorter jobs so as to minimize the waiting times of jobs [1]. When requests are made for various resources on a computer system, the Operating System must service these requests sequentially in some order, which is usually determined by the requesting processes' priorities. Job shop scheduling is an optimization problem in computer science and operations research in which jobs are assigned to resources at certain times. The Course Timetabling problem has Rooms and Instructors as shared resources that must be allocated together at the same time and place in order to be able to schedule an hour of a course. In order to simplify the scheduling problem, each day is divided into a number of periods (usually 50 minutes long, and the periods are separated by 10-minute breaks) [2]. The number of days on which courses can be scheduled varies between 5 and 6, and the number of periods is 5, 6, and 9 in the benchmark problems used in the experiments. The classrooms are another important resource for the timetabling problem because their sizes can vary greatly and the courses must be assigned to classrooms such that the student capacity of a classroom must be bigger than or equal to the number of students registered in that course.

The timetabling problem has four hard constraints [3] which are: Hard constraint-1 (HC1): all lectures of a course must be scheduled to a distinct time period. Hard constraint-2 (HC2): there can be only one course/section assigned to a classroom for each day-period. Hard constraint-3 (HC3): lectures of courses in the same curriculum and courses taught by the same lecturer must be assigned to different day-periods.

Hard constraint-4 (HC4): no lectures can be assigned to a day-period if the lecturer of that course/section is unavailable, where unavailability for a course is a list of (day,period) pairs representing the fact that the instructor of that course is not available at those periods for teaching the course.

An academic institution's administration requires the assignment of courses of an academic semester to instructors (most of the instructors will be full-time faculty members, but some of them could also be part-time, externally hired personnel). Next, once the assignment of the instructors of each course/section is done, it is now necessary to assign the courses to day-periods without violating any one of the hard constraints defined above. Administrators can, simultaneously, have other desirable goals that will increase the satisfaction of students, instructors and administrators by providing them with more comfortable schedules such as, not requiring long consecutive teaching hours, eliminating frequent travels to university for just a single hour of teaching, or having to wait many hours between two lectures on the same day. Teaching loads of lecturers must be in line with their contractual teaching loads, a number of classrooms used for scheduling all courses can be kept at a minimum to reduce physical maintenance costs and efficient sharing of a building by several departments, night classes, and others [4]. It is preferable to assign a full-time faculty member to teach a course because he/she will be paid the same salary even if not teaching the full course load in an academic semester [3].

Other possible considerations are the personal preferences of the faculty staff in specific course assignments [5], or lecture hour preferences of lecturers [6]. Some faculty members might prefer early hours in the morning, while others might arrange their daily activities to teach late in the afternoon. It would be in the interest of the administrators to increase the satisfaction of faculty members by avoiding the assigning of a lecturer's courses to undesirable day-periods. What makes this difficult and time-consuming problem even more challenging is the frequently occurring unexpected changes in the curriculum such as requiring to open an additional section for a course or having to open an unplanned course for the new semester, which might require re-shuffling of courses between faculty members since only a small number of faculty members would be prepared and/or willing to teach a course at such a short notice. It is very helpful to use a software tool that will quickly list alternative schedules for

lecturers based on the new allocation of courses to lecturers. Once the heuristics defined in this course timetabling research, including this thesis are shown to produce good feasible results, these additional considerations can be evaluated and algorithms to better satisfy them can be explored.

In [5] it is suggested that, we can define the assignment of lecturers to courses as a zero-one assignment problem, which decides first the assignment of faculty members to courses at the department planning level. Second, the lecturer-course pair is assigned to the available day-periods,the choice of day-periods must be done in such a way that the resulting schedule will have an improvement to favor a minimum number of course preparations. In this thesis it is assumed that the assignment of courses to teachers has already been performed, there is an additional soft constraint in our formulation which requires a course's lecture hours to be divided to at least the required "minimum working days".

1.2 Advantages of Greedy Algorithms in Course Timetabling Problem

Greedy algorithms are efficient polynomial time algorithms that work with a heuristic to solve a hard problem locally. It is not always possible to obtain optimal solutions using the greedy algorithms [7]. However, they are still good approaches to generate approximate solutions in reasonable amounts of time [8]. For example, the well-known Travelling Salesman problem has a high computation cost and by visiting the nearest cities first, we can get a solution of the problem using a greedy approach easily. It uses a limited number of steps in the solution and works in a deterministic way.

In this thesis, we present a set of greedy algorithms (Greedy-CB-CTT) for the optimization of the well-known Curriculum-Based Course Timetabling (CB-CTT) problem. The Curriculum concept groups courses in department and/or program and is used to model the fact that a student following a given Curriculum will register to all of the courses in that Curriculum, therefore all day-periods assigned to all courses in the same curriculum must be distinct as the same student cannot be in two different classrooms at the same day-period. The CB-CTT algorithms try to find the

best weekly assignment of university course lectures to classrooms and day-periods [9]. The main goal of our study is to minimize the total number of soft constraint violations while preserving the satisfaction of hard constraints. Since the problem is NP-Hard and large instances of the problem cannot be solved optimally in practical times, the greedy algorithms that work in seconds are still good alternatives to brute-force and evolutionary algorithms that spend hours of execution times to discover optimal or best achievable solutions. Instead of using a single heuristic as it is performed by many greedy algorithms, we define and evaluate many heuristics on the same problem instance simultaneously and report the overall best result, which is better than using a single-heuristic greedy algorithm [10]. The best results of all greedy heuristics are reported because the No Free Lunch Theorem (NFL) [11] states that there cannot be a single optimization algorithm that will always give the best results for all optimization problems. Specially designed different heuristic approaches can provide good solutions for different problem instances.

Our proposed greedy algorithms (Greedy-CB-CTT) orders Courses using Largest-Course-First [12], Smallest-Course-First, Average-Course-First [13], with respect to the number of students enrolled in a course [14]. Other parameters for ordering Courses is the number of unavailable hours listed for a course, number of courses in a curriculum, the number of lecture hours, and the number of minimum working days. The second set of ordering heuristics while assigning the courses, is the available rooms that can be ordered by their capacity according to four different criteria: largest, smallest, average-size and best-fit matching the number of students enrolled in the course being assigned to a classroom. In order to evaluate the performance of our proposed algorithm, we carried out experiments on 21 problem instances from the Second International Timetabling Competition (ITC-2007) benchmark set [15]. The results verify that the proposed greedy algorithms are able to report feasible (i.e., zero hard constraint violations) with significantly reduced soft-constraint values.

The main contributions of this thesis are as follows:

- A set of greedy algorithms are proposed for the solution of the CB-CTT problem.
- Popular task and page ordering heuristics are combined in the Greedy-CB-CTT

algorithms in order to select the results of the best heuristic according to the behaviour of the problem instances.

- Average course-size and average room-size first heuristics are applied to the CB-CTT problem for the first time and the results are reported.
- It was possible to obtain better results than classical largest/smallest first greedy heuristic algorithms, yielding solutions giving significantly lower hard and soft constraint violations.
- All hard constraint violations of the ITC-2007 benchmark set are reduced to zero in a few seconds during the experiments.

In Chapter 2, related studies for state-of-the-art algorithms are presented. The formal problem definition is given in Chapter 3. The details of heuristics and the proposed algorithms are introduced in Chapter 4. The experimental setup, obtained results, and comparison with state-of-the-art algorithms are reported in Chapter 5. Concluding remarks and future work are provided in Chapter 6 of the thesis.

CHAPTER 2

RELATED WORK

In this part of our thesis, we give brief information about related work; namely linear programming, metaheuristics, and greedy methods that could provide solutions for the CB-CTT problem. Wolpert & Macready develop a number of No Free Lunch (NFL) theorems about how to match algorithms to optimization problems [11]. They report that there is a strong correlation between heuristic algorithms and the problems they solve. They claim that any elevated performance of a heuristic over one class of problems is offset by performance over another class. These theorems try to explain in a geometric way what a fitness of an algorithm to the optimization problem means. Applications of the NFL to information-theoretic aspects of optimization processes and benchmark measures of performance are explained. We believe that the NFL theorems imply for our problem that, we can develop a set of heuristics such that each one will produce a different solution for a given CB-CTT problem instance, and by comparing these different solutions we can select and decide the best solution to optimize the CB-CTT problem.

The practical use of course-timetabling algorithms differ from the CB-CTT framework in the sense that a feasible solution may not exist in real life problems. In real-life applications, timetabling hard constraints are ordered according to their importance and solutions are found such that they are satisfied them as much as possible. Next, these partial solutions with remaining lecture assignments are manually processed by academic management staff. They make decisions to have a complete schedule for all of the lectures which will contain an acceptable set of hard-constraint violations. For CB-CTT competition, however, the benchmark problem set are pro-

duced synthetically, based on real university data, and feasible solutions are known to exist.

Tillet[4] presented the results of the feasibility of determining an optimal assignment of lecturers to courses using an operations research model in 1975. It was one of the first applications in this area. University managements carefully plan the hiring of teaching personnel, construction of faculty and department buildings, and timely and sufficient financing of all of these activities [16]–[19]. Such high level problem modeling and management, however, cannot easily be used by academic departments [20], requiring the solution of individuals' problems caused by course-faculty and course-classroom-time scheduling. These complicated and interdependent constraints might cause serious timing and place conflicts, as well as causing emotional stress on faculty members and students. Using professionally developed software and/or an optimization model makes it possible to consider all of the lecture timing constraints of faculty members and assigning classrooms to courses without causing any conflicts. Such software can also try to satisfy as many desirable properties in schedules as possible, such as compactness and make best use of available classrooms by considering classroom sizes and locations.

A recent survey about the university course timetabling problems is prepared by Babaei et al. [21]. The authors give detailed information about the operations research techniques, metaheuristics [22], [23], and other intelligent novel methods for solution of the CB-CTT problem [24], also presenting distributed cooperative search methods that are scalable by their nature and ability to use increasing numbers of computers as problem size grows. MirHassani et al. analyze the main considerations of recent papers on the university course timetabling problems and introduce the hard and soft constraints, as well as most currently used objective functions [25]. Bettinelli et al. give a good classification of problem types in this domain; Examination Timetabling, Post-Enrolment-based Course Timetabling, and Curriculum-Based Course Timetabling, the latter being the focus of this thesis. [9].

The initial works in 1970s on this problem have focused on developing complex mathematical models, determining methods to calculate lower bounds achievable on a given problem instance, and using both exact and heuristic algorithms to find feasible

solutions of real problem instances. As more practical experience is gained and better models on course timetabling problem are developed, the researchers have identified and highlighted challenging research directions on the CB-CTT problem, which has reached its most challenging level by 2010.

In order to model and address faculty preferences for scheduling days and times of their course workload, several techniques have been first proposed in 70s and 80s for describing the criteria used by departments to assign courses to faculty members, and choosing suitable lecture days and hours [26]–[29]. The problems with collecting information and applying faculty preferences [20] is that they require a significant effort on behalf of planners and sometimes the workload of a faculty member may not be decided until the academic registrations of students are completed. Such uncertainty in planning can occur because, for example, after students choose their preferred electives, some of the initially offered elective courses may have to be cancelled due to low student demand, or due to the discovery of a departmental need for new elective courses. Another reason could be, a mandatory course that had a high failure rate in earlier semesters might cause students to ask for re-offering of that course in order to allow them to graduate on time and/or avoid having irregular student status preventing them from registering courses for which the failed course is a prerequisite (must be taken and passed before allowing to register the next course).

Optimal solution techniques based on mathematical programming models can be used for discovering assignments of faculty members to courses and meeting times of those courses such that no hard constraints are violated. Linear programming is one of those mathematical techniques [30], [31]. The constraints of the linear programming model can be formulated such that each course is to be assigned to a faculty member and each faculty member must be assigned his/her minimum teaching load. The optimization goal of linear programming would be to maximize the satisfaction levels of faculty members and students by assigning them favorable teaching and course schedules. Integer programming models [27], [29] have higher time complexity cost. The same constraints can be defined using integer programming where integer decision variables with zero and one values such that each course-section must be assigned to a single faculty member and the sum of the number of courses assigned to a faculty member must be greater than or equal to his/her teaching load. Bagger et

al. present two mixed-integer programming formulations for the Course Timetabling problem [32]. The authors show that their formulations contain underlying network structures by dividing the problem into two separate models and then combine these models using a maximum flow problem. This approach reduces the number of integer variables and improves algorithm performance. Phillips et al. propose a general integer programming-based approach for the minimal perturbation problem in university course timetabling to resolve infeasible parts while minimising the disruption or perturbation to the remainder of the timetable [33]. Pereira & Costa present a linear integer programming model to solve a timetabling problem of a university in Brazil [34]. The integer linear model aims to find solutions that meet lecturers' scheduling constraints (unavailable days-and-hours) which are treated as hard constraints, thereby generating more practical and comfortable schedules for the faculty members. The objective function of the problem calculates higher values for faculty members with higher ranks, thus preferring to assign more experienced faculty to teach the offered courses, thus formulating the preference of higher ranking faculty members as a positive soft constraint.

Mendez-Diaz et al. consider a new timetabling problem for a university in Argentina [35]. The authors describe the Post-Enrollment Course Timetabling Problem (PECTP), with an additional hard constraint of every student must attend at most one event per time-slot and two soft constraints: students should not attend three or more events in successive time-slots, and a student should not be required to attend only one event in a given day. The article proposes an Integer Linear Programming (ILP) model and a heuristic approach for the solution which has two stages where the first stage defines an assignment of classes to time-slots, and the second stage assigns the students to the classes.

Lach & Lubbecke developed an Integer Programming (IP) approach to the university course timetabling problem, in which the CB-CTT problem is divided into, the first part hard constraints(such as student curricula) and the second part considers soft constraints (such as timetables having a convenient structure), where weekly lectures are to be scheduled and assigned to rooms [36].

Hao & Benlic describe a divide-and-conquer approach combined with ILP to calculate

better lower bounds for the CB-CTT [37] problem instances and are able to prove that for 6 out of the 21 benchmark problem instances, "current best lower bounds" are optimal because they are equal to those 6 problems' calculated lower bounds. The experiments verify that this approach also gives improved lower bounds for 12 of the 21 CB-CTT benchmark instances. Kang [38] developed an algorithm using a first-order predicate logic model in Prolog language to solve the timetabling problem with 2-3 variables with a large selection of values from corresponding domains (lecturers, courses, classrooms) having dozens of constraints. Dinkel et al. describe a Decision Support System that considers all constraints of faculty members, rooms sizes, and faculty preferences such as teaching on certain times of day, back-to-back scheduling of several courses/sections, and avoiding certain times of day. Using a network flow problem-based model where master source and master sink nodes are defined and the flow is from sink to source, represents all of the sections scheduled by the model [39].

Kassicieh et al. build a decision support system based on a database containing all of the information about lecturers, courses, and classroom information [40]. A modeling subsystem for mathematically solving and producing feasible schedules and finally an interactive subsystem to change the database, re-executing models to dynamically change assignments and input priorities. Mathaisel & C. Comm address the problem of making changes in room assignments once a solution has been achieved, room change by request of a faculty member can have a domino affect also complicated by room sizes and layout of rooms [41]. The system allows users to use icons to change courses' time and place and performs all necessary updates on database.

Shih & Sullivan proposed a zero-one programming model to maximize the distribution of courses to faculty preferences and also satisfying a maximal number of course-time allocations to the satisfaction of faculty preferences [29]. In order to achieve these goals maximally, a two-step optimization method is proposed by considering these mentioned faculty-course and faculty-time preferences.

Goal programming is a technique commonly used for decision-making applications [42]–[45]. The course-faculty-time assignment problem has many conflicts between its goals, between faculty members willing to teach courses from common pool courses and those courses must be scheduled to be delivered in a common pool of classrooms,

another source of conflict between faculty members who are trying to reserve one of those classrooms at their scheduled teaching hour.

The solutions proposed by Harwood & Lawless and Schniederjans & Kim employ the goal programming technique to solve conflicts occurring in the faculty-course assignment problem [20], [27]. Implementation of the algorithms required in [27] is quite difficult and requires advanced programming skills. The constraints have been modeled as requirements rather than goals, which in turn requires a large amount of data collection. The model in [20] aims to make implementation easier. The goal of departments is to offer a set of courses and also satisfy faculty teaching load requirements, while also accounting for the teaching preferences of faculty members. By concentrating on the single dimension of faculty assignment of courses, the problem becomes simpler, yet its usability is limited. The course assignment to day-period times is not considered as the second dimension of this problem. Geiger presents local search algorithms for the International Timetabling Competition 2007 (ITC 2007) [3]. His heuristics are based on the threshold accepting, dealing with local optima by a deterministic acceptance of inferior solutions. A stochastic search of neighbors is developed removing some lecture assignments and reassigning to new day-period slots randomly. Zhipeng & Hao developed an Adaptive Tabu Search algorithm for the CB-CTT problem [46]. Their Tabu search algorithm uses standard three phases which are initialization, intensification and diversification, and also integrates many features such as an original double Kempe chains (a method used in the study of the graph coloring problems) neighborhood structure, a penalty-guided perturbation operator and an adaptive search mechanism [47].

Yasari et al. study a course timetabling problem for a university. The registration is implemented with pre-registration and drop/add phases [48]. The authors arrange a course timetable considering the courses cancellations risk and the possible changes. Gulcu and Akkan develop algorithms to identify a good Pareto front defined by the solution quality (penalty associated with soft-constraint violations) and a robustness measure [49]. Two multi-objective Simulated Annealing (SA) algorithms are developed. Banbara et al. develop an Answer Set Programming-based approach for the CB-CTT [50], [51].

Akkan & Gulcu focused on the CB-CTT problem and defined a robustness measure for the problem, and then tried to find a set of good solutions in terms of both penalty and robustness values [52]. They developed a model for a bi-criteria optimization problem and solved it by a hybrid Multi-objective Genetic Algorithm (GA), which makes use of Hill Climbing and SA algorithms. The algorithm is tested on the well-known ITC-2007 instances and shown to identify high quality Pareto fronts (the best solution set with respect to the objective function). Thepphakorn & Pongcharoen developed a Particle Swarm Optimisation (PSO) timetabling application to solve the real-world datasets of the University course timetabling problem [53]. Wang et al. proposed a Greedy and Genetic Fusion Algorithm to solve Course Timetabling Problem efficiently, which can obtain the local optimal solution by using Greedy Algorithm and provide a high-quality initial population for GA. The results verify that the algorithm has a faster convergence speed than classical GA [54].

Halaby solved the course timetabling problem of the department of mathematics, Cairo University by encoding it into Max-SAT [55]. Muller & Rudova presented an approach to curriculum-based timetabling to capture complex relations of curriculum-based timetabling problems. The curricula are defined by a model that includes optional courses and course groups among which students are expected to take a subset of courses [56]. Goh et al. utilise a two-stage approach for addressing the post enrolment course timetabling (PE-CTT) problem [57]. They try to find a feasible solution in the first phase. The solution is improved in terms of soft constraint violations in the second phase. Rangel-Valdez et al. develop a method to solve this special case CB-CTT problem using a SA metaheuristic [58].

Nagata & Ono proposed a Tabu search algorithm using a candidate list strategy with random sampling for the university course timetabling problem, where the neighborhood size can be adjusted by a parameter ratio. The authors controlled the trade-off between exploration and exploitation by adjusting the neighborhood size [59]. Nagata presented a local search-based algorithm for the post-enrollment-based course timetabling problem [60]. The neighborhood size is updated by constructing a random partial neighborhood, which is defined as a random subset of the entire neighborhood.

Wu et al. propose a new time-dependent heuristic for post enrolment-based course

timetabling problem [61]. A constructive phase is proposed to insert events into the timetable while obeying most hard constraints. A hill-climbing phase is designed to ensure the timetable meeting all the hard constraints. Song et al. developed an iterated local search algorithm is proposed to find the feasible solution for the University Course Timetabling Problem [62]. SA and a diversification method are merged. The results on 60 datasets show that the algorithm achieves competitive results. Goh et al. combine different local search algorithms into an iterative two stage procedure, Tabu Search with Sampling and Perturbation and an improved variant of SA [63], their reported results are competitive to best known solutions reported in the literature. Bagger et al. applied Benders' decomposition to the solution of the problem [64]. The objective was to assign a set of lectures to time slots and rooms. The approach works with segmenting the problem into time scheduling and room allocation problems, and the Benders' algorithm was employed to generate cuts that connected the time schedule and room allocation.

Muehlenthaler et al. considered the problem of creating fair course timetables in the setting of a university. The idea is that undesirable arrangements in the course timetable should be distributed in a fair way among the lecturers using a SA algorithm [65], [66]. Kenekayoro & Zipamone developed an ant system for the solution of ITC 2007 dataset [67]. The ant system was able to find feasible solutions in all instances of the dataset and close to optimal solutions in some instances. Bellio et al. propose an effective and robust single-stage SA method for solving the problem. The authors design and apply an extensive and statistically-principled methodology for the parameter tuning procedure [68], [69].

Zhang et al. developed two discrete migration operators for efficiently evolving the solutions based on the principle of global and local migration in ecogeography-based optimization, and design a repair process for effectively coping with infeasible timetables [70]. Aziz et al. developed a honey-bee mating optimization algorithm to enhance the performance using an adaptive guided variable neighborhood search as a worker(bee) for the course timetabling problem [71]. Soria-Alcaraz et al. described an automated configuration of a generic memetic algorithm to solve the problems in ITC 2007 Timetabling Competition [72]. Lewis & Thompson studied on the issue of solution space connectivity and demonstrate that when this is increased via

specialised neighbourhood operators, the quality of solutions achieved is generally enhanced [73], [74]. Badoni et al. proposed a new hybrid algorithm combining GA with local search and using events based on groupings of students is described to solve the university course timetabling problem. The algorithm produced promising results compared with the results other existing algorithms [75]. Wu prepared an method fusing parallel GA and Map/Reduce programming model to solve the university course-timetabling problem. The results are observed to improve the success rate and the conflict rate in solving the university course-timetabling problem [76]. Soria-Alcaraz et al. explored the implementation of a parallel set of heuristic algorithms based on GAs, Scatter Search and discrete PSO for CTTP problem [77]. Soria-Alcaraz et al. proposed a hyper-heuristic algorithm with an iterated local search procedure that autonomously combines a set of move operators [78]. Acha & Nieuwenhuis applied novel techniques based on propositional satisfiability (SAT) solvers and optimizers to the CB-CTT problem [79]. They obtain new lower bounds for benchmark sets.

Shaker et al. presented a combination of two metaheuristics, great deluge and tabu search for solving the university course timetabling problem. A new strategy has been introduced to control the application of a set of neighbourhood structures using the tabu search and great deluge [80]. Abuhamdah et al. proposed a population-based Local Search heuristic embedded within a local search algorithm [81]. Kalender et al. proposed hyper-heuristic algorithms for a curriculum based course timetabling problem at Yeditepe University [82].

Jaradat et al. presented an investigation of enhancing the capability of the Scatter Search metaheuristic in guiding the search effectively toward elite solutions on the course timetabling problems [83]. Bolaji et al. proposed Artificial Bee Colony (ABC) algorithm for the university course timetabling concerning the assignment of a set of courses to a set of rooms and timeslots according to a set of constraints [84].

Cambazard et al. presented a local search and constraint programming techniques [85]. The authors show the advantage of a list-colouring relaxation of the problem. Nothegger et al. present an Ant Colony Optimization (ACO) where ants construct solutions based on pheromones [86]. The algorithm is parallelized and the solution quality is improved significantly. Gunawan et al. developed an initial solution by a

mathematical programming approach based on Lagrangian relaxation and the solution is improved by a SA algorithm [87]. Al-Betar et al. proposed a key feature of memetic computing method which is a hybrid population-based global search algorithm [88]. The proposed algorithm obtained the best results reported in the literature until then.

Chen & Shih proposed a PSO algorithm and generated satisfactory course timetables that meet the requirements of lecturers and classes according to the various applied constraints [89]. Qaurooni & Akbarzadeh developed an evolutionary algorithm to solve a specific variant of the timetabling problem [90]. They aim to investigate the applicability of evolutionary operators to timetabling. Wu integrates expert systems and constraint programming to generate a novel artificial intelligence approach for a course timetabling system [91]. The author parallelized the execution of the timetabling system in emerging cluster systems.

As can be seen from these works, there is a long list of alternative methods that have attempted to solve the CB-CTT problem optimally and by performing various search methods supported by artificial intelligence and optimization techniques.

CHAPTER 3

PROBLEM DEFINITION

In this chapter, details of the course timetabling problem are given. The concepts of hard and soft constraints of the problem are introduced and the formal definition of the problem is presented.

3.1 Curriculum-based course timetabling (CB-CTT)

The CB-CTT problem is in the class of NP-hard problems and it occurs at the beginning of a semester. It includes the allocation of courses, lectures and students to a fixed set of time slots and rooms [21]. This problem must obey hard and soft constraints while performing the allocation of resources. The possible timetables are obtained after full satisfaction of hard constraints, and, next, soft constraints are better satisfied to increase the quality and satisfaction levels of students and instructors using these timetables [92]–[94]. The scheduling is an NP-complete problem and it cannot be solved in the polynomial time due to the exponential growth of this problem. Figure 3.1 gives the relationship of the CB-CTT problem with the other scheduling problems. The number of constraints in this problem differs from one institution to another. Therefore, the main aim of algorithms is to minimize the penalties of constraints [93], [95].

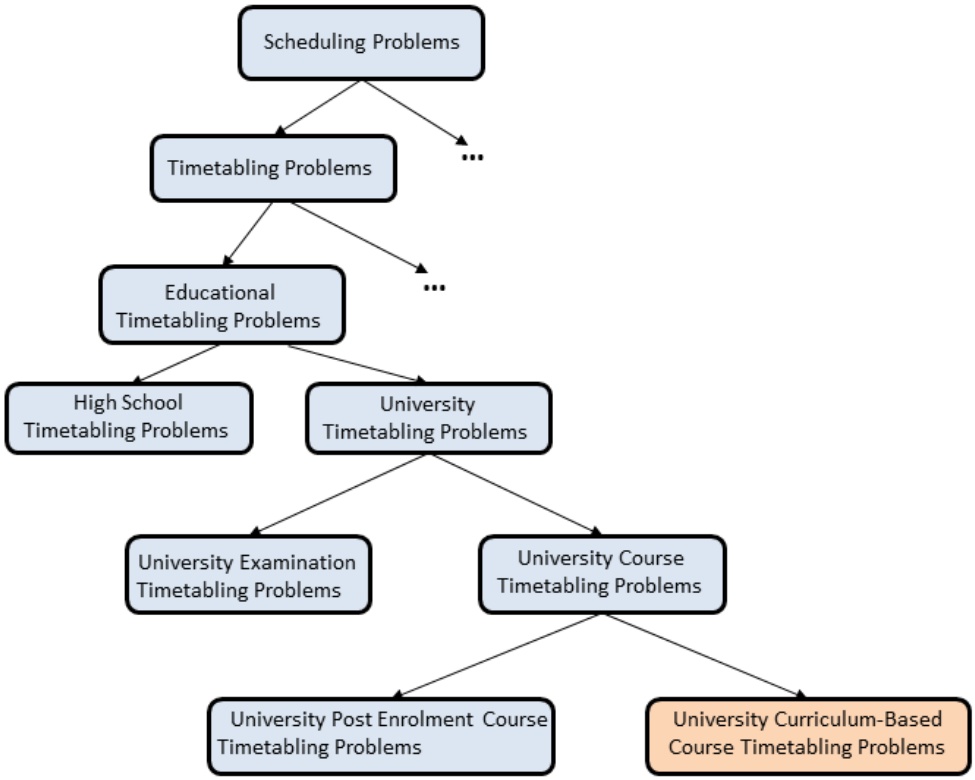

Figure 3.1: The relationship of the CB-CTT problem with other scheduling problems.

3.1.1 Problem description

The problem of scheduling a weekly timetable for lectures of a set of courses taught by a University department (a curriculum) is called the curriculum-based course timetabling (CB-CTT) problem. In CB-CTT, all of the lectures of courses must be scheduled to be taught in a certain day-period, and also one of the available rooms (preferably of sufficient size) while also taking care not to violate a set of physical constraints [46] such as not having two or more courses scheduled in the same room at the same period, courses taught by the same instructor must be scheduled at different periods, thus a schedule satisfying all of the hard constraints is called a feasible schedule. A feasible timetable requires all lectures of all courses to be assigned a time slot and a room, first of all. These are called the hard constraints(HC1-HC4) and all of these constraints must be satisfied in order to obtain a feasible timetable.

Once the hard constraints are satisfied, the next step is to try to satisfy, as much as possible, the soft constraints, which are mostly preferences of instructors in their choice of teaching days and hours, as well as the University departments requiring teaching loads of faculty members to be filled, and attending lectures for a minimum number of days of the week. Each of the soft constraint (SC1-SC4) violations causes a penalty cost for them, and the second objective of the CB-CTT problem is to minimize the total penalty of a schedule caused by soft constraint violations in a feasible solution.

The four hard constraints and four soft constraints are:

3.1.2 Hard Constraints

HC1. Lectures Each lecture of a course must be scheduled in a distinct time period and a room.

HC2. Room occupancy Any two lectures cannot be assigned in the same time period and in the same room.

HC3. Conflicts An instructor can teach only one course at a given time period, and also the lectures of courses in the same curriculum are assumed to be attended by the same set of students and therefore two courses in the same curriculum cannot be scheduled to the same time period.

HC4. Availability If the lecturer of a course is listed as not available for a given period, then no lectures of the same lecturer can be assigned to that period.

3.1.3 Soft Constraints

SC1. Room capacity: The number of students registered to a course cannot be greater than the capacity of the classroom where the lecture is assigned. The penalty calculated for a room smaller than number of registered students is given by the formula (RoomSize - NumberOfStudents)*2.

SC2. Room stability: In order to make it easier for students to remember in which classroom meets, it is preferred to assign all lectures of a course to the same room.

SC3. Minimum working days: In order to allow students to be able to attend at least part of a course's weekly meetings, it is preferred not to schedule all of the course hours to the same day, and distribute its lectures to a minimum number of days.

SC4. Curriculum compactness: If possible, the courses of a student is preferred to be on the same day and closely distributed within the same day. For example, if in a given curriculum, there is one lecture not adjacent to any other lecture in the same curriculum on the same day, a violation is counted.

A mathematical formulation is useful in that it provides a complete definition of the CB-CTT problem which can also be solved by standard software libraries, such as Simplex.

3.1.4 Problem formulation

We can define the CB-CTT problem in terms of N courses C $=c_1, c_2, ..., c_N$ all of which must be scheduled in a set of P periods T $=t_1, t_2, ..., t_P$, and a set of M rooms, R $=r_1, r_2, ..., r_M$

Each course c_i contains l_i lectures that must be scheduled to time periods.

A time period is a pair of week day (D days, typically monday through friday) and a timeslot (P periods consisting of D days and H daily timeslots (means $P = D * H$). Also, there are S curricula, CR $= Cr_1, Cr_2, ..., Cr_S$ where each curriculum Cr_k is a group of courses that are assumed to share the same students. Typically these curricula will contain must courses of a university department.

I: day of week.
J: time period of day.
K: Group of students
L: Lecturers
M: Courses
N: Classrooms

3.2 Formal problem definition

HC1. Lectures: All lecture hours of a given course must be on a different time period. For any given (day, time, course) the below summation must be zero or one.

$$\sum_{I}\sum_{J} X_{i,j,k,l,m,n} \leq 1 \tag{3.1}$$

HC2. Room occupancy: Any two lectures cannot be assigned in the same period and to meet in the same room. For day I, and time period J, classroom N, the sum of all $X_{i,j,k,l,m,n}$ (day i, time j, curriculum k, course m, classroom n) for a certain (day,time, classroom) must be zero or one.

$$\sum_{I}\sum_{J}\sum_{N} X_{i,j,k,l,m,n} \leq 1 \tag{3.2}$$

HC3. Conflicts An instructor can teach only one course at a given time period, and also the lectures of courses in the same curriculum are assumed to be attended by the same set of students and therefore any two courses in the same curriculum cannot be scheduled to the same time (day, period).

HC3.a. Lecturer, L, for a given day I, and time period, J, can be assigned to teach at most once. The variable $X_{i,j,k,l,m,n}$ (day i, time j, curriculum k, lecturer l, course m, classroom n) can have only two values 0 or 1, meaning a lecturer is assigned to teach that course on that day period or not. The summation below for any given Lecturer must be zero or one.

$$\sum_{k}\sum_{m}\sum_{n} X_{i,j,k,l,m,n} \leq 1 \tag{3.3}$$

HC3.b. Two courses in the same curriculum cannot be scheduled to the same time period. For given curriculum K, day I, and time J, the below summation must be zero or one.

$$\sum_{k}\sum_{m}\sum_{n} X_{i,j,k,l,m,n} \leq 1 \tag{3.4}$$

HC4. Availability If the lecturer of a course (Lecturer L) is not available at a given period (day I, time J), then no lectures of that lecturer can be assigned to that period.

$$\sum_{I,J,L} X_{i,j,k,l,m,n} = 0, where(I,J)inUnavailableSet(L) \tag{3.5}$$

CHAPTER 4

PROPOSED HEURISTICS AND GREEDY ALGORITHMS

The CB-CTT problem is an NP-Hard problem and even finding feasible solutions for the hard constraints is very difficult. Therefore, in this thesis it is preferred, first, to develop several greedy heuristics that will be able to satisfy the hard constraints and evaluate the performance of these greedy heuristics on standard benchmark data.

In order to be able to develop algorithms that can be used for quickly developing timetables, it will be very practical to have several heuristics that can provide solutions satisfying 2 or 3 of the hard constraints. Then, human experts can work on these ”mostly finished” timetables and get rid of the remaining hard constraint violations. Special algorithms reducing soft constraint violations can also be developed separately and can perform small modifications on the resulting timetables where all hard constraints have been eliminated.

The proposed Greedy Heuristics are defined in terms of classroom sizes and course sizes.

4.1 Course ordering heuristics

Highest unavailable hours first heuristic: A high number of unavailable hours listed for a course makes it very difficult to find a suitable time period for that course. For this reason, one of the course ordering heuristics sorts courses in non-increasing order of their unavailable hours. This way, it becomes easier to schedule such courses since they won’t be conflicting any other course in the same curriculum.

Table 4.1: Course ordering heuristics.

Id	Heuristic Description
1	Course with most students first
2	Course with average number of students first
3	Course with least number of students
4	Course with highest number of unavailable day/periods
5	Course with most number of hours first, if equal then largest unavailable hours
6	Course with most number of unavailable hours first, if equal more students
7	Course in curriculums with most courses first, next courses with more students
8	Course in curriculums with most unavailable hours first, next more students
9	Courses with highest number of lecture hours
10	Course with highest number of minimum working days, if equal more lecture hours

Largest, smallest and average course first: As the names imply in these course ordering heuristics, the number of students registered to courses are used to decide which courses are assigned a classroom first. By assigning courses that will have a larger impact on the resulting timetables first, there will be no need to change these first decisions because of conflicts with less important courses (in terms of number of registered students). This will result in greedy algorithms that will run very quickly, but may not always be guaranteed to come with the best and/or feasible timetables.

The first course to be assigned a classroom can be chosen as the largest size, the smallest size, or closest to the average size courses. This is achieved by sorting courses in terms course sizes and choosing the first, the last, or the median course in the sorted list. Then, the chosen course is removed from the list of courses and the process is repeated until all of the courses are assigned a classroom.

4.2 Room ordering heuristics

In this heuristic the classrooms are assigned to courses by considering their student capacities, resulting in best-fit (the classroom with smallest capacity that is large enough

Table 4.2: Room ordering heuristics.

Id	Heuristic description
1	Largest Room First
2	Smallest Room First
3	Average Room First
4	Best Fitting (with enough capacity) Room

for the number of students in that course), worst-fit (the largest classroom size), and average size which is the median size classroom among all of the available classrooms sorted in order of classroom sizes.

The Classroom-Heuristics used for choosing a classroom are combined with Course-Heuristics that choose the course to be assigned to that classroom, finally a timetable is decided by choosing an available day-time period in that classroom. Since there are 4 possible ordering choices for Classrooms and 10 orders for choosing a Course, there are potentially 40 greedy heuristics that must be investigated to choose the one(s) that give the best expected results.

Tables 4.2 and 4.1 give the names of the heuristics used in our proposed algorithm and their id's.

4.3 The proposed greedy algorithms (Greedy-CB-CTT)

The Greedy-CB-CTT algorithm makes use of the above heuristics that are defined in the previous sections of this chapter. N_{rooms}, $N_{courses}$, and $N_{curriculum}$ depict the number of available classrooms, number of courses to be scheduled, and number of curricula with a set of courses respectively. They are the input data of the algorithm. The output of the algorithm is the list of courses assigned to the classrooms. Mainly, the algorithm works with two main nested loops of rooms and the courses. The ordering heuristics of the classrooms and courses are selected with 40 different methods, because four classroom and ten course ordering alternatives exist. Inside these two loops, available classrooms are selected according to the minimum hard and soft violation values of

the assignment choices. Each of these alternatives are executed in a single thread and the best of overall results are collected at the end of the optimization process of the algorithm.

The pseudocode of the Greedy-CB-CTT is given in Algorithm 2. The flowchart of the proposed greedy algorithm is given in Figure 4.1.

The AssignDayPeriod Algorithm 1 function uses the given Room and Course identifiers to determine a suitable (Day,Period) to schedule a lecture which doesn't violate any of the hard constraints. It is used in all of the heuristic Algorithms 2-4 as it is a shared functionality. The CB-CTT model allows each problem instance to define its MAXDAY (5 or 6 in COMP benchmark) which means Saturday could also be added as teaching day. MAXPERIOD is the number of lecture slots available on all days, which is 6 or 7 for COMP benchmark, but can be defined by problem instance as a dynamic parameter. The beginning and ending hour of lectures of a day are not specified. It could be such that University management could decide to place the first slot of lectures early at 8:30am, while another University could prefer 8:00am or 9:00am. All other lecture slots will be placed by adding breaks and lectures.

The Algorithm 2 is a naive algorithm which assigns lectures of a course to the first available (Day,Period) slot without attempting to divide lectures of a course to minimum working days. Since the restriction for minimum working days is not enforced this heuristic must find feasible solutions more easily and acts as a lower bound.

The Algorithm 3 is a more intelligent algorithm as it first decides how to divide total lecture hours of a course into the designated minimum working days for that course. Then, it will call the AssignDayPeriod to assign lectures of a course to a particular day. Since minimum working days is a soft constraint, the solutions returned by this algorithm will have lower penalty scores compared to Algorithm 2.

The Algorithm 4 exploits the Curriculum based structure of the CB-CTT model. Another important soft constraint that must be minimized in CB-CTT benchmark instances is to make sure that lectures of courses belonging to the same Curriculum must be scheduled on the same day and, if possible, lectures of all courses in the same curriculum must be scheduled consecutively. Since this heuristic schedules courses in

Algorithm 1: AssignDayPeriod(RID,CID,LH) Pseudocode for finding a (Day, Period) for given Course and Room

```
1  Input:
     RID : see if room has available (day,period)
     CID : see if course can be assigned (day,period)
     LH : number of lecture hours to be assigned Output: (day,period) on
     SUCCESS, or FAILURE

2  for each day in Range(NumberOfDays) do
3      for each period in Range(NumberOfPeriods − LH) do
4          if (day,period)∈ DRoom[rid].Assigned then
5              continue # try next period
6          if (day,period)∈ DCourse[cid].UnavailableSet then
7              continue # try next period
8          if (day,period)∈ DCourse[cid].Assigned then
9              continue # try next period
10         if (day,period) ∈ DTeacher[Tid].Assigned then
11             continue # try next period
12         if (day,period)∈) DCurriculum[Curid]].Assigned then
13             continue # try next period
14         for hour in Range(LH) do
15             add (day,period+hour) to DCourse[cid].Assigned
16             add (day,period+hour) to DCurriculum[Curid]].Assigned
17             add (day,period+hour) to DTeacher[Tid].Assigned
18             add (day,period+hour) to DRoom[rid].Assigned
19         Return SUCCESS_Day_Period
20 Return FAIL
```

Algorithm 2: Pseudocode of the proposed HEUR-1 Greedy Algorithm

Input: *RoomList* ordered list of classrooms

CourseList : ordered list of courses to be scheduled

DCourse['*course* − *id*'] : Assigned: set-of-assigned-day-periods ; Curricula: set-of-curricula containing that course; Unav: set-of-unavailable-day-periods

Output: The classroom and (day,period) assigned to a course. The resulting Hard and Soft constraint violation reasons and calculated penalties.

DCourse['*course* − *id*'].*Assigned* = NULLSET

DRoom['*room* − *id*'].*Assigned* = NULLSET

DCurriculum['*curr* − *id*'].*Assigned* = NULLSET

for *(cid in CourseList)* **do**

    for *(rid in RoomList)* **do**

        LH= DCourse[cid].LectureHours

        for *hour in Range(LH)* **do**

            AssignDayPeriod(rid,cid,1) # assign 1 hour at a time

Return DCourse[].Assigned # (day,period) sets for all courses

Algorithm 3: Pseudocode of the proposed MinWorkingDays HEUR-2 Greedy Algorithm

Input:

RoomList ordered list of classrooms

CourseList : ordered list of courses to be scheduled

DCourse['course-id']:

Assigned: set-of-assigned-day-periods

Curricula: set-of-curricula with that course

Unavailable: set-of-unavailable-day-periods

Output: The classroom and (day,period) assigned to a course. The resulting Hard and Soft constraint violation reasons and calculated penalties.

$DCourse['course-id'].Assigned$ = NULLSET

$DRoom['room-id'].Assigned$ = NULLSET

$DCurriculum['curr-id'].Assigned$ = NULLSET

for *(cid in CourseList)* **do**

    PARTS= DivideIntoMinWorkingDays(DCourse[cid].LectureHours)

    for *(rid in RoomList)* **do**

        for *(partHours in PARTS)* **do**

            AssignDayPeriod(rid,cid,partHours)

Return DCourse[].Assigned # (day,period) sets for all courses

Algorithm 4: Pseudocode of the proposed Curriculum-first based HEUR-3 Greedy Algorithm

Input:

RoomList ordered list of classrooms

CurriculumList : ordered list of Curriculums

DCourse['course-id']:

Assigned, Curricula, Unavailable: as above

Output: The classroom and (day,period) assigned to a course. The resulting Hard and Soft constraint violation reasons and calculated penalties.

$DCourse['course-id'].Assigned$ = NULLSET

$DRoom['room-id'].Assigned$ = NULLSET

$DCurriculum['curr-id'].Assigned$ = NULLSET

$FINISHED = NULLSET$

for *(curid in OrderCurriculums(CurriculumList))* **do**

  for *(cid in OrderCourses(DCurriculum[curid].CourseList))* **do**

    if *cid* ∈ *FINISHED* **then**

      continue

    PARTS= DivideIntoMinWorkingDays(DCourse[cid].LectureHours

    for *(rid in RoomList)* **do**

      for *(partHours in PARTS)* **do**

        AssignDayPeriod(rid,cid,partHours)

Return DCourse[].Assigned # (day,period) sets for all courses

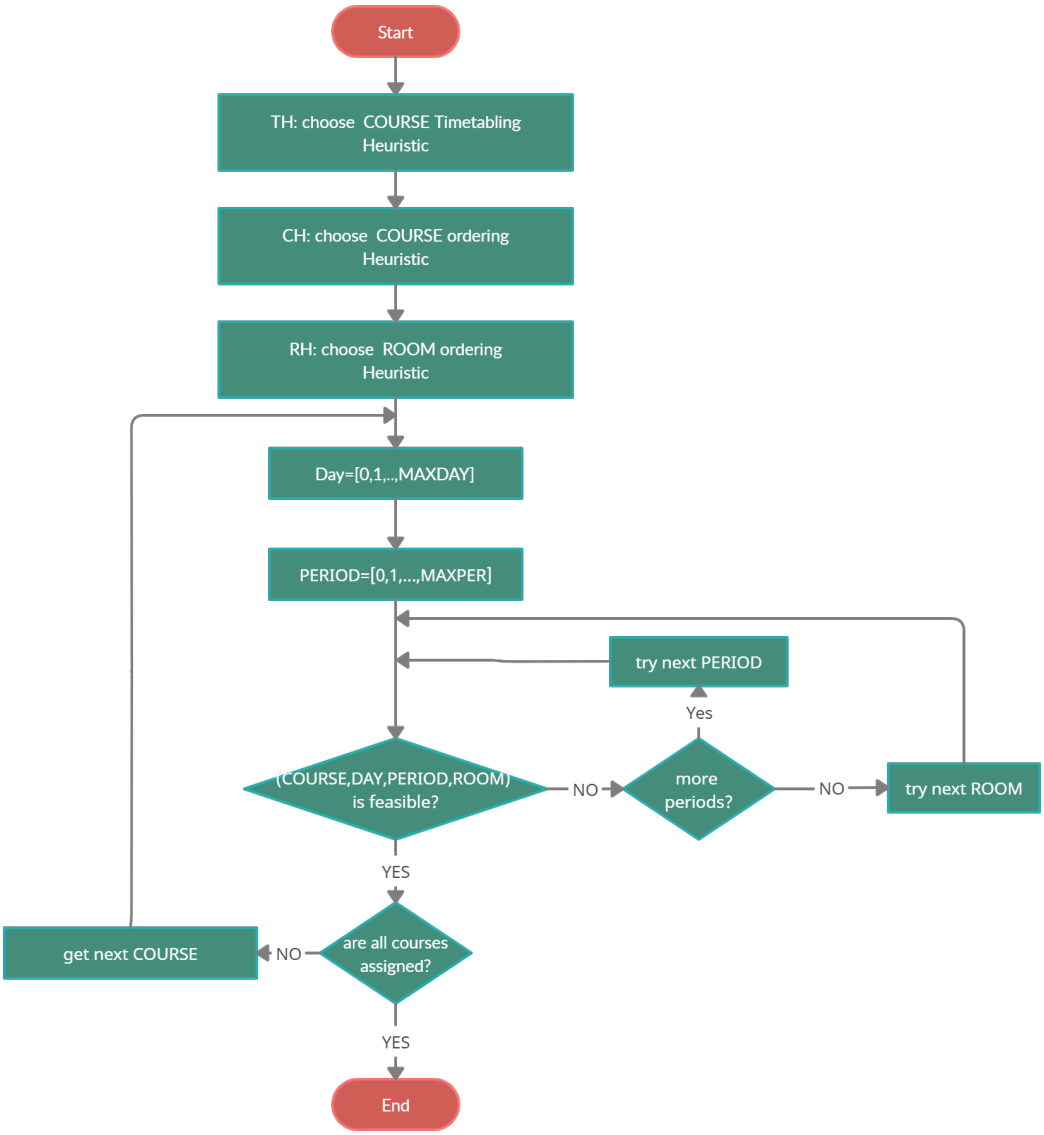

Figure 4.1: The flowchart of the Greedy-CB-CTT algorithm.

the same curriculum in a bundle, it attempts to find a day which has enough free hours and can accommodate all of the lectures of the same Curriculum courses assigned to that day. Of course the total lecture hours of Curriculum courses assigned to a single day must be at most MAXPERIOD which is the maximum available lecture slots on a single day.

4.4 Data structures used in the proposed algorithm

During the optimization of the problem, we need to perform many search operations on the lists of Curricula (*cur*) and Constraints (*cons*). This main search operation was the most time consuming operation of the algorithm. Therefore, In this part of the proposed algorithm, we use a hash array (a ”dictionary” in Python language) to check if a certain piece of information exists in the data structure.

For each Room, a hash array is used storing (roomid, day, period) triplet as the index and the value assigned to this index is Course-id. Using this hash array it requires only a simple hash access to check if a given ”room” is available for a certain ”day” and ”period”.

For Courses, two set data structures are used, the first consists of (day,period) to store the assigned times for a course, so that no other session of the same course will be assigned to the same time, which is a hard constraint. The second set structure contains the unavailable (day,period) values which cannot be assigned to the course because of teacher unavailability, this is also a hard constraint.

For each Curriculum, a set data structure is kept which stores the (day,period) pairs which have been assigned to any course in that curriculum. Storing this information is necessary because no other course in the same curriculum must be assigned to a (day,period) which has been assigned to another course in the same curriculum. This is also a hard constraint, and it is easily checked by using a set data structure by just checking if the set already contains an element with the same (day,period) value.

For every Teacher, a set of (day,period) values is stored to record a course taught by that instructor has been assigned the given (day,period). It is a hard constraint not to have any two courses taught by the same Teacher assigned to the same (day,period).

These hash-array and set data structures reduce the search times to check for these hard constraint violations in O(1) time, instead of O(N) time, at the expense of O(N) space where size of N is Number_of_Days X Number_of_Periods.

4.5 The complexity analysis of the proposed greedy algorithm

The CB-CTT problem is NP-Hard and solving the problem using a brute-force algorithm is going take exponential times according to the size of the problem. Therefore, evolutionary, greedy, or heuristic algorithms are preferred to obtain approximate solutions. In this part of our thesis, we explain the theoretical complexity analysis of our proposed algorithms. As it may be observed from the execution times of our proposed algorithm, the results can be achieved in a very short time for all of the 21 benchmark problem instances used in ITC-2007. The algorithms proposed in this thesis are greedy algorithms and the best advantage of this approach is that it will produce reasonable solutions in polynomial times. The steps of our heuristics are given below. NC is number of Courses and NR is number of Rooms.

Step 1: Sort Courses according to heuristic course order: O(Nlog N) where N is the number of courses.

Step 2: Sort Rooms according to heuristic room order: O(Nlog N), where N is the number of rooms.

Step 3: for each Course
Step 4: for each Room
Step 5: find (day, period) such that all hard constraints are satisfied: O(1) * O(1)

Steps 1 and 2 are O(N logN) because they require sorting. Steps 3,4,5 are O(NC * NR * NDays * NPeriods) since each course is considered once, and for each Course each Room is again considered only once, and NDays*NPeriods is also a constant value. Verifying hard constraints is O(1) because they involve a simple hash array lookup.

The overall time complexity becomes:
O(NClogNC) + O(NRlogNR) + O(NC*NR)*O(1).
This means the running time of our heuristics are equivalent to sorting algorithms where problem size is defined by the larger of number of courses (NC) and number of rooms (NR). However, if NC and NR are comparable in size then the third term O(NCxNR) becomes O(N^2).

CHAPTER 5

EXPERIMENTAL SETUP AND THE EVALUATION OF RESULTS

In this part of our thesis, we give detailed information about the benchmark datasets that are used in the experiments. The software and hardware settings of the experiments and the computational results of the experiments in terms of execution time, hard and soft violations of the problem instances are presented. At the end of the chapter, a comparison of the results is presented with state-of-the-art algorithms. A comprehensive evaluation and a discussion are provided. A validator (written in the C++ programming language) for the solutions is provided by the competition committee of the ITC-2007 [15]. Our results have been verified using this validator executable in our experiments to avoid any implementation errors. The algorithm is developed using Python programming language. The PC has 8 GB memory and a I7 2.4 GHz processor.

5.1 The ITC-2007 benchmark problem instances

There are 21 problem instances available in this benchmark problem set [15]. All instances are real data and come from the University of Udine. For all instances there exists at least one feasible solution (no hard constraint violations), but at present it is not known what the optimal value is for the soft constraints. In order to model cases in which the number of timeslots is not the same for all days (e.g. Saturday afternoon free), there might be periods unavailable for all courses. Furthermore, for all instances there cannot be two curricula composed by exactly the same courses.

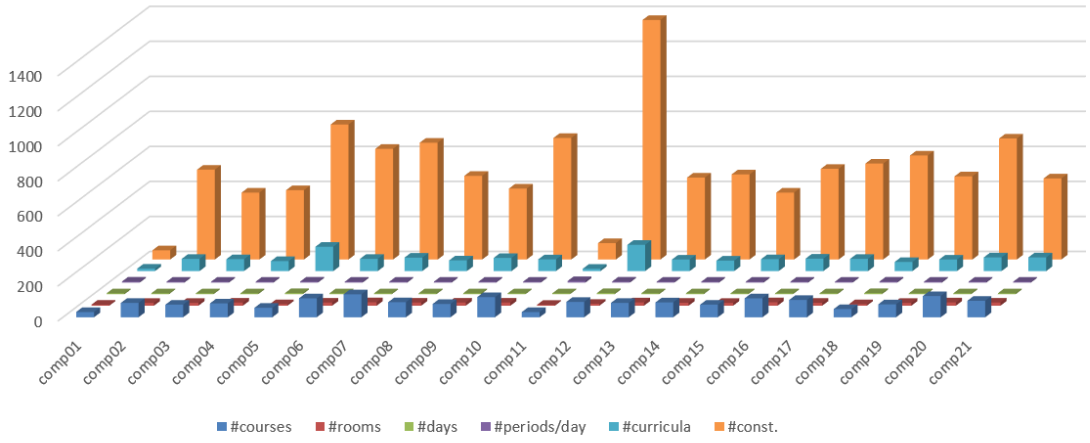

Figure 5.1: The visual representation of ITC-2007 benchmark instances.

Each instance comes in a single ASCII text file, and the files are named as comp01.ctt, comp02.ctt ,..., comp21.ctt. Each file contains a file header and four sections: courses, rooms, curricula (containing groups of courses), and constraints (pairs of unavailable (day,period) for a course). The header provides all scalar values and each section provides the arrays for that specific aspect of the problem.

The details of the benchmark problem instances (comp01 through comp21) used in our experiments are given in Table 5.1. The parameters of the problem instances are summarized in eight columns. The columns, in order, contain problem name, #courses, #rooms, #days, #periods_per_day, #curricula, and #Constraints. The range of the data given in the columns are 30 to 121 for #courses, 5 to 20 #rooms, 5 to 6 #days, 5 to 6 #periods_per_day, #curricula (where courses are assigned in groups to curricula, where a course can occur in more than one curriculum) varying from 13 to 150, and is called #constraints (the number of unavailable periods where a given course cannot be scheduled) varying from 53 to 1368. comp01 is one of the easiest problem instances with the smallest number of courses, 30, only 6 classrooms, 14 curricula, and just 53 unavailable periods in the problem set. Two of the hardest problem instances are comp10 and comp12, and these instances have 115 and 88 courses, 67 and 150 curricula, 18 and 11 rooms, 694 and 1368 constraints, respectively.

Table 5.1: The details of the ITC-2007 benchmark problem instances. (See Figure 5.1 for visual representation of the problem instances.)

problem	name	#courses	#rooms	#days	#periods/day	#curricula	#constraints
comp01	Fis0506-1	30	6	5	6	14	53
comp02	Ing0203-2	82	16	5	5	70	513
comp03	Ing0304-1	72	16	5	5	68	382
comp04	Ing0405-3	79	18	5	5	57	396
comp05	Let0405-1	54	9	6	6	139	771
comp06	Ing0506-1	108	18	5	5	70	632
comp07	Ing0607-2	131	20	5	5	77	667
comp08	Ing0607-3	86	18	5	5	61	478
comp09	Ing0304-3	76	18	5	5	75	405
comp10	Ing0405-2	115	18	5	5	67	694
comp11	Fis0506-2	30	5	5	9	13	94
comp12	Let0506-2	88	11	6	6	150	1368
comp13	Ing0506-3	82	19	5	5	66	468
comp14	Ing0708-1	85	17	5	5	60	486
comp15	Ing0203-1	72	16	5	5	68	382
comp16	Ing0607-1	108	20	5	5	71	518
comp17	Ing0405-1	99	17	5	5	70	548
comp18	Let0304-1	47	9	6	6	52	594
comp19	Ing0203-3	74	16	5	5	66	475
comp20	Ing0506-2	121	19	5	5	78	691
comp21	Ing0304-2	94	18	5	5	78	463

5.2 The execution results with three heuristics (Heur-1, Heur-2, Heur-3)

In this part, we give the results of 120 algorithms in Tables 5.2, 5.3, and 5.4 for 21 problem instances respectively. The name of the algorithm (ALG), the total sum of hard constraint points (Total Hard), the total sum of soft constraint points (Total Soft), the total sum of constraint points including both hard and soft points (Total Sum), the number of feasible solutions with zero hard constraint points found by the algorithm (Feasibles Found), the total number of best solutions obtained compared to other algorithms (Min at), and the number of worst solutions found by the algorithm compared to other algorithms (Max at) are the results given in the Tables.

In Table 5.5, the best feasible results with lowest penalty score and the algorithm that reports it for the problem instance is reported. We were able to obtain a feasible solution (the sum of the hard constraint violation penalty scores is zero) for all problem instances, except three, for our 21 instance benchmark set. For the three instances that none of the 40 greedy heuristics could provide a zero violation feasible solution, the hard constraint violation penalty score were only 2 (comp02), 1 (comp19), and 1(comp21).

The results of the experiments performed with Heur-1 heuristic on 21 benchmark datasets are presented in Tables A.1 through A.40 in Appendix to give detailed information to interested readers A. Since it takes a lot of space to report the results of 120 algorithms, we could only give 40 summarized results of this algorithm in Table 5.5. In Appendix B, The output files for the solutions of comp01, comp12, and comp18 with Heur-1: 1/5, Heur-1: 1/1, and Heur-2: 4/6 algorithms, respectively are also provided. These output files are input to the validator and the hard/soft violations are obtained according to the rules of the competition ITC-2007.

5.3 The results of classical greedy algorithms

In this section, we present the solution of 3 greedy algorithms commonly used in the literature. The algorithms are largest course to largest room first, smallest course to smallest room first, and the best-fit greedy algorithms. In Table 5.6, we give the results

Table 5.2: Summary of execution results with Heur-1.

ALG	Total Hard	Total Soft	Total Sum	#Feasible Found	Min at	Max at
1/1	33	17357	17390	9	9	0
1/2	161	93810	93971	4	0	0
1/3	223	117891	118114	1	0	0
1/4	168	87125	87293	3	0	0
1/5	31	17143	17174	8	0	0
1/6	30	44302	44332	9	0	0
1/7	37	47221	47258	7	0	0
1/8	21	43532	43553	12	0	0
1/9	164	85818	85982	3	0	0
1/10	107	50004	50111	3	0	0
2/1	33	203164	203197	9	0	0
2/2	161	127556	127717	4	0	0
2/3	223	69768	69991	1	0	0
2/4	168	126205	126373	3	0	0
2/5	31	203128	203159	8	0	0
2/6	30	178353	178383	9	0	0
2/7	37	172171	172208	7	0	0
2/8	21	180140	180161	12	0	0
2/9	164	133798	133962	3	0	0
2/10	107	163546	163653	3	0	0
3/1	33	114531	114564	9	0	0
3/2	161	96753	96914	4	0	0
3/3	223	108703	108926	1	0	0
3/4	168	108826	108994	3	0	0
3/5	31	114501	114532	8	0	0
3/6	30	115418	115448	9	0	0
3/7	37	117688	117725	7	0	0
3/8	21	117974	117995	12	0	0
3/9	164	113545	113709	3	0	0
3/10	107	111433	111540	3	0	0
4/1	35	17420	17455	10	0	0
4/2	149	18323	18472	3	0	0
4/3	207	18454	18661	2	0	0
4/4	139	18243	18382	2	0	0
4/5	26	17505	17531	9	0	0
4/6	30	17129	17159	11	1	0
4/7	27	17726	17753	12	0	0
4/8	17	17362	17379	14	0	0
4/9	138	17846	17984	3	0	0
4/10	94	17443	17537	3	0	0
AVG	94.7	86471.4	86566.1	6.2	0.2	0.0

Table 5.3: Summary of execution results with Heur-2.

ALG	Total Hard	Total Soft	Total Sum	Feasibles Found	Min at	Max at
1/1	151	16063	16214	3	12	0
1/2	196	91752	91948	2	0	0
1/3	272	114225	114497	2	0	0
1/4	213	85241	85454	1	0	0
1/5	96	15440	15536	5	0	0
1/6	82	42794	42876	5	0	0
1/7	96	43876	43972	7	0	0
1/8	105	40369	40474	9	0	0
1/9	214	82768	82982	1	0	0
1/10	226	46525	46751	3	0	0
2/1	146	201039	201185	4	0	0
2/2	196	125667	125863	2	0	0
2/3	270	68651	68921	2	0	0
2/4	210	124988	125198	2	0	0
2/5	93	200642	200735	6	0	0
2/6	82	175948	176030	5	0	0
2/7	96	169858	169954	7	0	0
2/8	105	177875	177980	9	0	0
2/9	210	130794	131004	2	0	0
2/10	228	159863	160091	3	0	0
3/1	151	112604	112755	3	0	0
3/2	196	93880	94076	2	0	0
3/3	266	103957	104223	2	0	0
3/4	210	106250	106460	2	0	0
3/5	96	112317	112413	5	0	0
3/6	82	113640	113722	5	0	0
3/7	96	113919	114015	7	0	0
3/8	105	115806	115911	9	0	0
3/9	211	109223	109434	1	0	0
3/10	220	108871	109091	4	0	0
4/1	146	15214	15360	1	0	0
4/2	220	15456	15676	3	2	0
4/3	267	17073	17340	2	0	0
4/4	226	16117	16343	2	0	0
4/5	97	14994	15091	5	1	0
4/6	83	15789	15872	8	0	0
4/7	111	14876	14987	7	0	0
4/8	104	15406	15510	8	0	0
4/9	203	16296	16499	2	0	0
4/10	261	16824	17085	3	0	0
AVG	165.9	84072.2	84238.1	4.0	0.4	0.0

Table 5.4: Summary of execution results with Heur-3.

ALG	Total Hard	Total Soft	Total Sum	Feasibles Found	Min at	Max at
1/1	101	54762	54863	5	0	21
1/2	108	74588	74696	5	0	0
1/3	92	79033	79125	6	0	0
1/4	91	28580	28671	7	1	0
1/5	133	68897	69030	3	0	0
1/6	121	44540	44661	6	0	0
1/7	111	42302	42413	5	0	0
1/8	104	97504	97608	7	0	0
1/9	120	97493	97613	6	0	0
1/10	144	94052	94196	5	0	0
2/1	101	161756	161857	5	0	0
2/2	108	136584	136692	5	0	0
2/3	92	136629	136721	6	0	0
2/4	91	182805	182896	7	0	0
2/5	133	150438	150571	3	0	0
2/6	121	171017	171138	6	0	0
2/7	111	170362	170473	5	0	0
2/8	104	119101	119205	7	0	0
2/9	120	118645	118765	6	0	0
2/10	144	118772	118916	5	0	0
3/1	101	119350	119451	5	0	0
3/2	108	123152	123260	5	0	0
3/3	92	120438	120530	6	0	0
3/4	91	116705	116796	7	0	0
3/5	133	115472	115605	3	0	0
3/6	121	112982	113103	6	0	0
3/7	111	110423	110534	5	0	0
3/8	104	115251	115355	7	0	0
3/9	120	116472	116592	6	0	0
3/10	144	114926	115070	5	0	0
4/1	90	15120	15210	9	0	0
4/2	97	15654	15751	6	0	0
4/3	91	16006	16097	6	0	0
4/4	84	15479	15563	5	0	0
4/5	151	15375	15526	2	1	0
4/6	113	15331	15444	6	0	0
4/7	133	15395	15528	5	0	0
4/8	174	16228	16402	5	0	0
4/9	144	16455	16599	4	0	0
4/10	143	16310	16453	4	1	0
AVG	114.9	86759.6	86874.5	5.4	0.1	0.5

Table 5.5: The best results and the algorithms that report the solutions.

Prob. Instance	ALG (roomH/courseH)	Total Hard	Total Soft	Total Penalty
comp01	Heur-1: 1/5	0	234	234
comp02	Heur-2: 4/8	2	800	802
comp03	Heur-3: 4/1	0	600	600
comp04	Heur-2: 4/7	0	574	574
comp05	Heur-1: 4/8	0	1287	1287
comp06	Heur-3: 4/4	0	838	838
comp07	Heur-1: 1/1	0	1005	1005
comp08	Heur-3: 4/5	0	580	580
comp09	Heur-2: 1/5	0	629	629
comp10	Heur-3: 4/1	0	796	796
comp11	Heur-1: 4/3	0	190	190
comp12	Heur-2: 4/8	0	1129	1129
comp13	Heur-3: 4/3	0	612	612
comp14	Heur-2: 4/7	0	668	668
comp15	Heur-3: 4/1	0	600	600
comp16	Heur-1: 4/1	0	899	899
comp17	Heur-1: 1/8	0	2776	2776
comp18	Heur-2: 4/5	0	426	426
comp19	Heur-3: 4/4	1	607	608
comp20	Heur-1: 4/7	0	953	953
comp21	Heur-1: 1/6	1	2017	2018
Avg	—	0.19	867.62	867.81

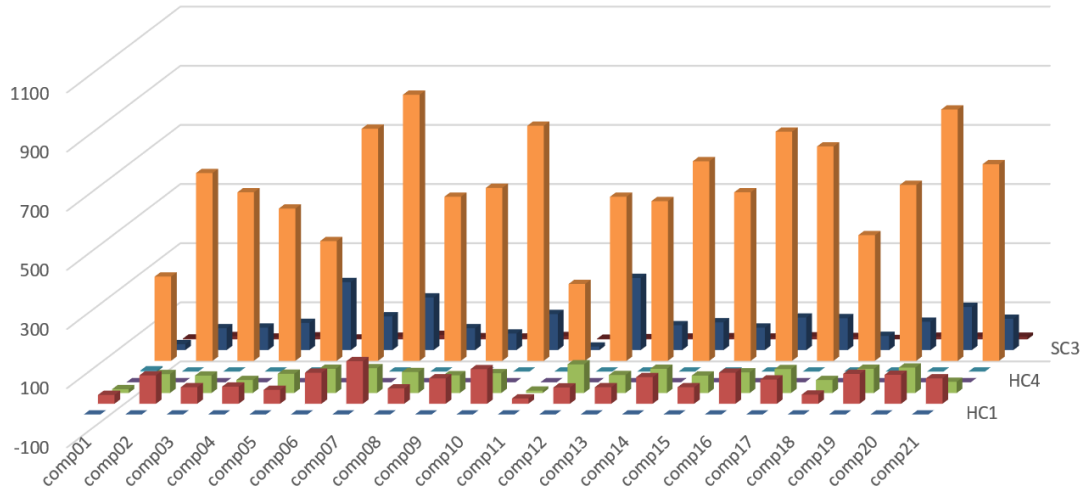

Figure 5.2: The visual representation of largest course, in terms of number of students, to the largest room results.

of Largest course to largest room first greedy algorithm. In Table 5.7, we give the results of smallest course to smallest room first greedy algorithm. Table 5.8 presents the result of the best-fit greedy algorithm that assigns the courses to the rooms with minimum empty spaces.

We think that by summarizing the results in these three tables, it becomes possible to have a brief and good evaluation for our algorithms that can be compared against these 3 naive algorithms. Non of these three algorithms, has been able to locate even a single solution with zero hard violations. It can be observed that the largest course to largest room first algorithm is the one that reports the best solutions in the average. Total point of 713.4 is reported by this algorithm for 21 problem instances. The smallest course to the smallest room first algorithm is the worst one among the three algorithms with highest 3390.4 points. When a comparison is made with the algorithms we have developed, it is seen that the performance of the algorithms we propose is much higher, with a general average score of 867.8 and with the ability to find 18 feasible solutions.

Table 5.6: The results of the greedy algorithm which matches the largest course, in terms of number of students, to the largest room. (see Figure 5.2)

problem	HC1	HC2	HC3	HC4	SC1	SC2	SC3	SC4	TOTAL
comp01	0	30	13	0	4	285	20	3	312
comp02	0	96	64	0	0	635	74	11	720
comp03	0	56	59	0	0	570	76	6	652
comp04	0	58	44	0	0	515	92	10	617
comp05	0	47	65	0	0	405	230	3	638
comp06	0	105	82	0	0	785	114	9	908
comp07	0	144	84	0	0	900	178	15	1093
comp08	0	52	71	0	0	555	74	7	636
comp09	0	86	60	0	0	585	56	6	647
comp10	0	117	67	0	0	795	122	11	928
comp11	0	18	8	0	0	260	12	2	274
comp12	0	55	97	0	0	555	244	4	803
comp13	0	56	61	0	0	540	84	6	630
comp14	0	90	82	0	0	675	94	7	776
comp15	0	56	59	0	0	570	76	6	652
comp16	0	105	70	0	0	775	110	10	895
comp17	0	82	81	0	0	725	108	8	841
comp18	0	31	44	0	0	425	48	1	474
comp19	0	101	82	0	0	595	96	9	700
comp20	0	98	86	0	0	850	146	9	1005
comp21	0	86	38	0	0	665	106	9	780
total	**0**	**1569**	**1317**	**0**	**4**	**12665**	**2160**	**152**	**14981**
average	**0**	**74.7**	**62.7**	**0**	**0.2**	**603.1**	**102.9**	**7.2**	**713.4**

Table 5.7: The results of the greedy algorithm which matches the smallest course to the smallest room first. (see Figure 5.3)

problem	HC1	HC2	HC3	HC4	SC1	SC2	SC3	SC4	TOTAL
comp01	0	15	14	0	483	270	12	4	769
comp02	0	101	66	0	3483	635	124	8	4250
comp03	0	64	57	0	3247	555	122	7	3931
comp04	0	62	56	0	5496	540	64	6	6106
comp05	0	53	62	0	10412	360	310	3	11085
comp06	0	102	85	0	852	780	144	11	1787
comp07	0	114	77	0	344	915	116	14	1389
comp08	0	63	64	0	3013	550	86	9	3658
comp09	0	83	72	0	3477	540	108	7	4132
comp10	0	126	97	0	204	795	98	10	1107
comp11	0	21	7	0	1306	260	12	3	1581
comp12	0	52	87	0	2577	555	224	1	3357
comp13	0	64	67	0	6342	525	94	9	6970
comp14	0	83	73	0	3098	675	92	7	3872
comp15	0	64	57	0	3247	555	122	7	3931
comp16	0	112	61	0	1746	785	116	9	2656
comp17	0	77	91	0	349	720	158	11	1238
comp18	0	17	51	0	2122	410	56	1	2589
comp19	0	93	84	0	1257	570	108	8	1943
comp20	0	125	76	0	2347	840	166	11	3364
comp21	0	90	48	0	673	665	138	8	1484
total	**0**	**1581**	**1352**	**0**	**56075**	**12500**	**2470**	**154**	**71199**
average	**0**	**75.3**	**64.4**	**0**	**2670.2**	**595.2**	**117.6**	**7.3**	**3390.4**

Table 5.8: The results of best-fit greedy algorithm. (see Figure 5.4)

problem	HC1	HC2	HC3	HC4	SC1	SC2	SC3	SC4	TOTAL
comp01	2	26	13	0	88	300	12	3	403
comp02	1	81	76	0	0	655	104	4	763
comp03	1	70	64	0	0	555	78	6	639
comp04	0	53	49	0	0	520	80	8	808
comp05	0	67	45	0	0	430	186	0	616
comp06	0	97	66	0	0	765	116	9	890
comp07	1	167	79	0	0	895	162	9	1066
comp08	1	68	68	0	0	560	78	10	648
comp09	0	62	65	0	0	540	94	4	638
comp10	2	128	76	0	0	805	102	9	916
comp11	0	35	7	0	0	270	0	1	271
comp12	0	90	77	0	0	560	234	3	797
comp13	0	78	59	0	0	530	70	7	607
comp14	0	66	76	0	0	675	118	6	799
comp15	1	70	64	0	0	555	78	6	639
comp16	0	112	63	0	0	775	120	8	903
comp17	3	101	72	0	0	725	126	8	859
comp18	0	57	32	0	0	440	20	0	460
comp19	2	92	52	0	0	595	74	6	675
comp20	0	126	92	0	0	835	144	8	987
comp21	0	140	31	0	0	675	136	10	821
total	**14**	**1786**	**1226**	**0**	**88**	**12660**	**2132**	**125**	**15205**
average	**0.7**	**85.0**	**58.4**	**0**	**4.2**	**602.9**	**101.5**	**5.9**	**724.0**

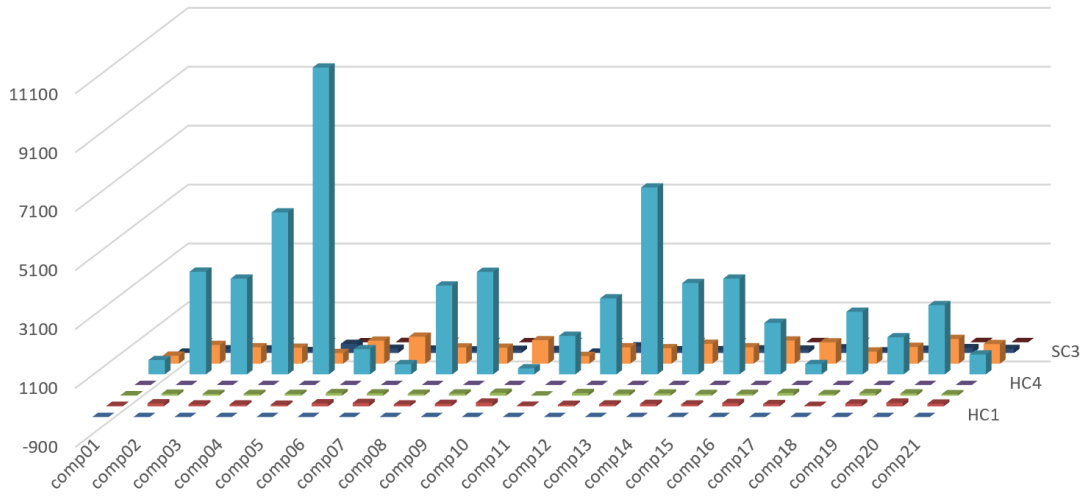

Figure 5.3: The visual representation of the smallest course to the smallest room first results.

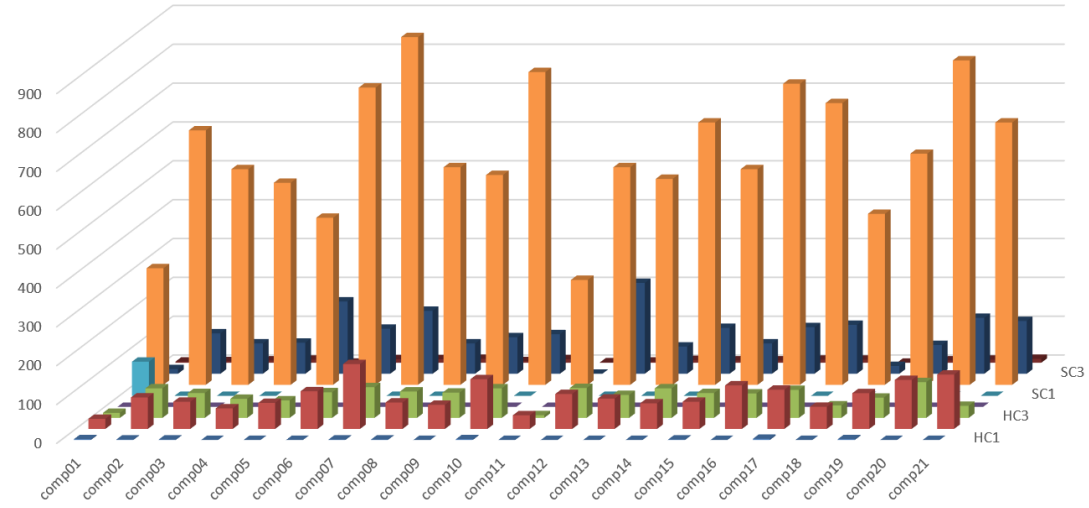

Figure 5.4: The visual representation of best-fit greedy results.

5.4 The execution times of the proposed algorithms

In this part, the execution times of each benchmark instance, using each Course-ordering and Room-ordering heuristic combination, are calculated individually and given in a compact form. Table 5.9 gives the details of 40 Heur-1 algorithms. The

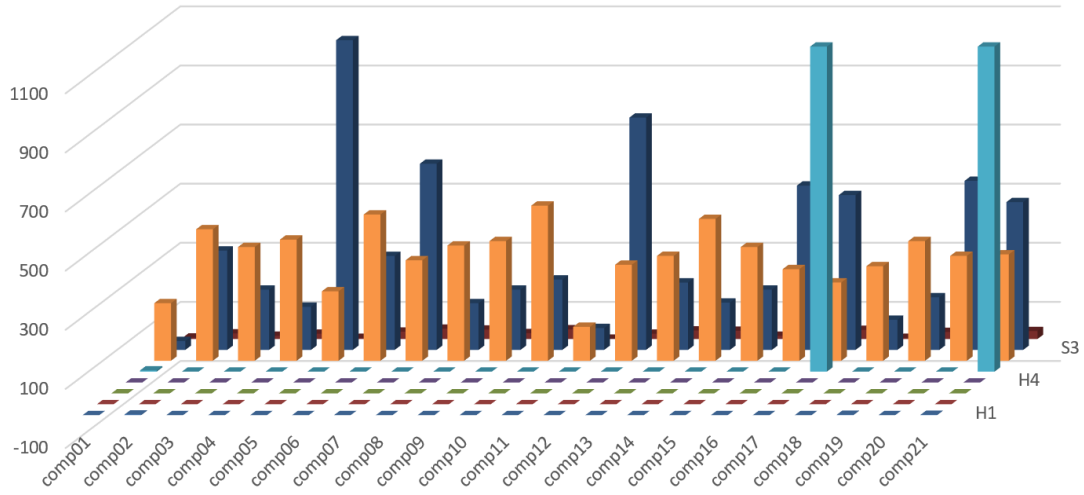

Figure 5.5: The visual representation of some of the selected heuristics from Table 5.9.

total execution time for all the algorithms in this Table is 42.335 seconds. The total execution time for all Heur-1, Heur-2, and Heur-3 algorithms (120 algorithms) for 21 problem instances is 69.978 seconds. Our execution time show that Heur-1 takes 4 times more than Heur-2 and 2 times more time than Heur-3, this is even though Heur-2 and Heur-3 are more elaborate and make extra effort to allocate lecture hours into minimum working days. This means that it becomes even more efficient to find feasible solutions to the CB-CTT problem instances by using more intelligent algorithms rather than performing a blind-search checking each alternative classroom and day-period until all lectures of a course are scheduled without any hard conflicts.

5.5 Comparison with state-of-the art algorithms

In this part of our thesis, we compare the experimental performance results of our algorithms with state-of-the-art algorithms in literature. In Table 5.10, we can compare the best-known results reported in the literature with those discovered by our algorithms. The algorithms that we compare against our solutions are obtained from the papers of Muller [96], Lu & Hao [46], Atsuta [97], Geiger [3], and Clark [98]. In this Table, the total scores of the results on 21 problems are presented. The best schedules are given in bold digits. The best algorithms in literature are mostly evolutionary and

Table 5.9: The execution times of selected Heur-1 greedy algorithms in miliseconds.

Test	1	2	3	4	5	6	7	8	9	10
Comp01	20/18/17/9	20/16/17/12	16/16/17/13	16/16/17/12	16/16/20/8	27/21/18/9	20/16/16/12	16/18/16/12	16/18/16/12	18/16/18/9
Comp02	56/58/52/30	63/66/71/32	58/58/58/30	54/56/58/28	57/56/56/28	66/65/60/29	52/49/52/25	62/56/61/30	52/52/53/26	58/56/56/30
Comp03	46/44/49/20	54/54/49/29	52/48/48/28	42/45/46/25	44/45/44/24	50/93/54/24	44/45/44/20	48/47/49/21	49/49/45/26	48/48/48/25
Comp04	52/59/53/30	66/62/62/36	61/64/62/28	56/57/57/29	56/58/53/32	56/56/60/30	56/51/65/32	57/56/56/34	56/57/60/28	60/58/56/32
Comp05	21/20/20/9	32/28/28/12	29/29/32/12	30/26/33/10	20/25/18/12	24/26/25/12	20/21/16/12	21/25/25/8	29/29/28/13	20/24/25/8
Comp06	92/88/88/40	96/98/103/44	90/92/89/40	94/98/98/40	88/92/90/36	101/99/98/40	145/87/86/36	99/94/98/40	103/98/93/40	89/92/92/42
Comp07	124/126/128/66	137/137/136/80	139/141/138/76	128/129/128/70	128/127/126/68	137/134/135/73	122/129/126/69	130/127/132/72	134/128/132/73	139/139/132/71
Comp08	71/68/68/36	76/72/76/46	74/70/72/42	70/74/68/45	73/72/70/41	74/72/77/40	71/72/93/38	74/72/72/37	74/73/73/40	75/69/73/36
Comp09	61/54/63/30	66/65/64/36	56/58/60/28	57/57/56/29	56/56/60/28	61/62/69/27	57/52/56/32	61/58/62/32	59/57/62/31	56/60/58/31
Comp10	88/100/94/48	96/101/100/55	88/95/101/50	90/90/95/50	92/90/94/51	98/98/100/53	86/87/89/49	96/98/92/48	92/92/94/49	86/90/90/48
Comp11	20/18/23/8	20/16/16/13	16/20/21/8	20/21/16/8	20/21/22/9	21/21/18/8	21/17/18/8	16/40/20/8	20/20/21/6	20/21/20/8
Comp12	45/40/44/25	50/48/52/32	48/52/52/28	48/48/48/24	44/45/40/23	56/52/57/24	41/44/40/22	55/52/50/20	57/49/47/25	40/45/42/24
Comp13	64/64/64/32	67/73/68/41	66/64/64/36	64/64/69/36	64/64/66/32	64/68/69/34	60/63/58/32	64/69/64/32	66/68/64/34	65/65/68/33
Comp14	54/52/58/29	57/56/54/34	52/52/57/33	52/53/51/28	56/53/52/32	59/61/62/28	50/50/48/24	56/56/56/28	56/53/58/29	62/62/57/28
Comp15	48/50/48/21	55/50/62/29	49/48/50/25	47/45/44/25	44/44/45/25	50/48/53/25	47/42/45/22	48/44/50/25	47/53/48/22	52/48/49/25
Comp16	100/94/92/46	98/100/102/56	96/94/94/50	98/97/104/50	92/91/90/48	100/102/96/49	90/88/84/41	99/103/98/46	96/99/106/48	94/96/97/53
Comp17	84/80/84/45	88/84/84/48	80/80/81/36	95/84/83/41	78/85/82/38	90/90/90/40	72/76/78/34	85/86/86/40	84/85/82/37	87/80/86/38
Comp18	16/16/16/8	20/21/20/12	21/20/20/9	20/16/21/9	16/16/16/8	19/18/23/9	16/14/17/8	21/20/16/9	20/16/20/8	16/20/16/9
Comp19	52/52/54/29	59/57/58/32	56/55/53/30	48/53/50/28	52/53/55/28	58/56/58/32	48/46/55/30	60/60/58/27	55/65/60/32	65/54/50/32
Comp20	106/109/119/57	140/138/133/60	117/111/114/56	105/105/110/48	108/106/106/47	113/116/116/52	100/100/102/44	110/104/108/49	106/108/108/48	109/110/116/48
Comp21	77/78/77/39	88/93/90/47	85/82/85/44	80/87/82/37	77/76/76/37	82/80/86/38	69/74/76/34	81/75/78/36	78/77/80/41	78/81/77/36
Average	53.8	60.6	56.4	54.8	53.3	58.4	52.0	55.8	55.6	55.1

Table 5.10: Competition results of ITC-2007; the best discovered results on all the 21 competition instances (given in bold face).

Instance	Müller	Lü & Hao	Atsuta	Geiger	Clark	Batuhan
comp01	**5**	**5**	**5**	**5**	10	234
comp02	51	55	**50**	111	111	802
comp03	84	**71**	82	128	119	600
comp04	37	43	**35**	72	72	574
comp05	330	**309**	312	410	426	1287
comp06	**48**	53	69	100	130	838
comp07	**20**	28	42	57	110	1005
comp08	41	49	**40**	77	83	580
comp09	109	**105**	110	150	139	629
comp10	**16**	21	27	71	85	796
comp11	**0**	**0**	**0**	**0**	3	190
comp12	**333**	343	351	442	408	1129
comp13	**66**	73	68	622	113	612
comp14	59	**57**	59	90	84	668
comp15	84	**71**	82	128	119	600
comp16	**34**	39	40	81	84	899
comp17	**83**	91	102	124	152	2776
comp18	83	**69**	68	116	110	426
comp19	**62**	65	75	107	111	608
comp20	**27**	47	61	88	144	953
comp21	**103**	106	123	174	169	2018
avg	**79.8**	80.9	85.8	180	132.5	916

require a lot of time to converge, whereas our algorithms only spend a few seconds while optimizing the given problems.

As we explained in Section 5.3, our intelligent Course and Room ordering heuristics give feasible schedules with lower soft constraint violations than the results of simple traditional greedy heuristics, which are largest course to largest room first, smallest course to smallest room first, and the best-fit which are used for memory allocation in Operating Systems.

The results reported by Geiger in his 2012 paper for comp01 - comp14 are given in Table 5.11 [3]. To the best of our knowledge, these are the best results reported in the literature. The results are obtained after 100,000,000 trials for each problem instances.

Table 5.11: Best results out of 30 runs for benchmark problems reported by Geiger in 2012.

Instance	Seed	HC violations	SC Violations	#Evaluations
comp01	130	0	5	13,072,619
comp02	3	0	108	8,547,980
comp03	3	0	0	9,211,859
comp04	0	0	0	10,352,548
comp05	0	0	0	6,512,059
comp06	3	0	0	8,631,146
comp07	3	0	0	7,673,851
comp08	0	0	0	9,881,464
comp09	0	0	0	9,248,758
comp10	3	0	0	8,386,538
comp11	3	0	0	13,468,229
comp12	0	0	0	6,782,742
comp13	0	0	0	9,838,210
comp14	0	0	0	9,693,538

The feasible solutions are reported with minimum soft constraint violations. Our algorithms are much faster than the Geiger's algorithm. We are not able to include all of these solutions in this thesis since they would take up too many pages, but the fastest results (with zero hard constraint violation values) ever reported for the ITC-2007 problem instances are observed with our algorithms. This is because our algorithms focus on obtaining feasible solutions without any hard constraint violations. The next step, which is not addressed in this thesis, must try to eliminate as many soft constraint violations as possible while taking care not to introduce any hard constraint violations.

CHAPTER 6

CONCLUSION

In the last chapter of this thesis, a general evaluation of the developed greedy algorithms is given, and new ideas for future studies are proposed. In our study, the first goal was to develop many greedy heuristic algorithms that can optimally solve all problems efficiently. As we investigated more about the Curriculum-Based Course Timetabling (CB-CTT) problem and realized how difficult it was to eliminate all soft constraint violations, we decided it would be a significant contribution to develop even feasible solutions (timetables having zero hard constraint violations). We observed that it was not possible to find a single greedy heuristic that would work well for all problem instances and outperform all other heuristics. Therefore, it is difficult to identify and propose a single method to solve all benchmark instances. Our investigations show that it would be more effective to use a different heuristic for different problem instances. Further research is needed to find a method to decide which greedy heuristic would perform better on a given CB-CTT problem instance, and not on other problem instances. This is a future challenge to be investigated for this problem. New benchmark problem instances are also being introduced for this problem. It can be interesting to apply our proposed algorithm to these new problem sets and observe the results, also justifying our experimental findings on different benchmark datasets.

The No Free Lunch Theorem (NFL) [11] tells us that there will always be new ideas, new approaches which will lead to better optimization algorithms to solve a given problem. Instead of being forgotten in a short time, it is far more likely that most of the currently known optimization methods have at least one niche, one area where

they are excellent. During this process, creative ideas are most important. It has been experimentally shown in this thesis that very fast greedy heuristics can help eliminate hard constraint violations in CB-CTT, these results also show that it might as well be possible to find greedy algorithms to greedily eliminate at least a substantial portion of soft constraint violations.

A reasonable trade-off between exploration (diversification) and exploitation (intensification) is a crucial task for the optimization performance, accuracy prediction and convergence speed [99]. There is no clear answer to this question yet. With fitness landscape analysis and information landscapes approaches a better balance between these activities is trying to be decided [100]. The balance between exploration and exploitation does not mean that each will have 50% of the optimization time. This is an issue that should be well set by the heuristic algorithms.

We were able to outperform classical greedy algorithms, largest course to largest room first, smallest course to smallest room first, and the best-fit greedy algorithms [101]. The performances of the algorithms we propose are much higher, with a general average score of 867.80 and with the ability to find feasible solutions for all problem instances. The most important advantage of our algorithms is that they are much faster than evolutionary approaches and work well with even very large problem instances. While the evolutionary algorithms spend hours of computation time, our algorithms can get feasible solutions in a few seconds.

Combining greedy algorithms with evolutionary algorithms is a commonly used technique for the optimization of difficult problems. In future, using the new heuristics developed in this thesis can be a employed to obtain even better solutions to the CB-CTT problem. The solutions discovered by our greedy algorithms can be given as part of the initial solution pool, as input to an evolutionary algorithm. In order to speed up the evolutionary algorithms it is possible to use a high performance multicore parallel machine in order to simultaneously evolve the solutions discovered by an evolutionary algorithm. The Message Passing Interfaces (MPI) framework can be a good environment to develop new parallel algorithms for this well-known problem [102].

REFERENCES

[1] M. Pinedo, *Scheduling: theory, algorithms, and systems*. Upper Saddle River: Prentice-Hall, 2002.

[2] W. Ruegg, *A History of the University in Europe: Volume 3*. Cambridge: Cambridge University Press, 2004.

[3] M. J. Geiger, "Applying the threshold accepting metaheuristic to curriculum based course timetabling - A contribution to the second international timetabling competition ITC 2007," *Annals OR*, vol. 194, no. 1, pp. 189–202, 2012.

[4] P.I.Tillet, "An operations research approach to the assignment of teachers to courses," *Socio-Economic Planning Sciences*, vol. 9, no. 3-4, pp. 101–104, 1975.

[5] M. J. Schniederjans, "A goal programming model to optimize departmental preference in course assignments," *Computers and Operations Research*, vol. 14, no. 2, 1987.

[6] G. B. Harwood, "Optimizing organizational goals in assigning faculty teaching schedules," *Decision Sciences*, vol. 6, no. 3, pp. 513–524, 1975.

[7] R. A. DeVore and V. N. Temlyakov, "Some remarks on greedy algorithms," *Advances in computational Mathematics*, vol. 5, no. 1, pp. 173–187, 1996.

[8] A. R. Barron, A. Cohen, W. Dahmen, R. A. DeVore, *et al.*, "Approximation and learning by greedy algorithms," *The annals of statistics*, vol. 36, no. 1, pp. 64–94, 2008.

[9] A. Bettinelli, V. Cacchiani, R. Roberti, and P. Toth, "An overview of curriculum-based course timetabling," *Top*, vol. 23, no. 2, pp. 313–349, 2015.

[10] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu, "Hyper-heuristics: A survey of the state of the art," *Journal of the Operational Research Society*, vol. 64, no. 12, pp. 1695–1724, 2013.

[11] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.

[12] R. Panigrahy, K. Talwar, L. Uyeda, and U. Wieder, "Heuristics for vector bin packing," *research. microsoft. com*, 2011.

[13] G. Dósa and J. Sgall, "Optimal analysis of best fit bin packing," in *International Colloquium on Automata, Languages, and Programming*, Springer, 2014, pp. 429–441.

[14] K. Fleszar and C. Charalambous, "Average-weight-controlled bin-oriented heuristics for the one-dimensional bin-packing problem," *European Journal of Operational Research*, vol. 210, no. 2, pp. 176–184, 2011.

[15] L. Di Gaspero, B. McCollum, and A. Schaerf, "The second international timetabling competition (itc-2007): Curriculum-based course timetabling (track 3)," Tech. Rep., 2007.

[16] L. S. Franz, W. M. Lee, and J. C. Van Horn, "An adaptive decision support system for academic resource planning," *Decision Sciences*, vol. 12, no. 2, pp. 276–293, 1981.

[17] C. Joiner, "Academic planning through the goal programming model," *Interfaces*, vol. 10, pp. 86–92, 1980.

[18] S. Lee and E. Clayton, "A goal programming model for academic recourse planning," *Management Science*, vol. 18, pp. 390–408, 1972.

[19] L. Smith, "Planning models for budgeting teaching resources," *Omega*, vol. 6, pp. 83–88, 1978.

[20] M. Schniederjans and G. Kim, "A goal programming model to optimize departmental preference in course assignments," *Computers and Operations Research*, vol. 14, no. 2, pp. 87–96, 1987.

[21] H. Babaei, J. Karimpour, and A. Hadidi, "A survey of approaches for university course timetabling problem," *Comput. Ind. Eng.*, vol. 86, pp. 43–59, 2015.

[22] I. Boussaïd, J. Lepagnot, and P. Siarry, "A survey on optimization metaheuristics," *Information sciences*, vol. 237, pp. 82–117, 2013.

[23] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms," *Computers & Industrial Engineering*, vol. 137, p. 106 040, 2019.

[24] S. Abdullah, H. Turabieh, B. McCollum, and P. McMullan, "A hybrid metaheuristic approach to the university course timetabling problem," *J. Heuristics*, vol. 18, no. 1, pp. 1–23, 2012.

[25] S. A. MirHassani and F. Habibi, "Solution approaches to the course timetabling problem," *Artif. Intell. Rev.*, vol. 39, no. 2, pp. 133–149, 2013.

[26] J. Breslaw, "A linear programming solution to the faculty assignment problem," *Socio-Economic Planning Science*, vol. 10, pp. 227–230, 1976.

[27] G. Harwood and R. Lawless, "Optimizing organizational goals in assigning faculty teaching schedules," *Decision Sciences*, vol. 6, pp. 513–524, 1975.

[28] R. McClure and C. Wells, "A mathematical programming model for faculty course assignment," *Decision Sciences*, vol. 15, pp. 409–420, 1984.

[29] W. Shih and J. Sullivan, "Dynamic course scheduling for college faculty via zero-one programming," *Decision Sciences*, vol. 8, pp. 711–721, 1977.

[30] G. Andrew and R. Collins, "Matching faculty to courses," *College and University*, vol. 46, pp. 83–89, 1971.

[31] J. Dyer and J. Mulvey, “An integrated optimization/information system for academic department planning,” *Management Science*, vol. 22, pp. 1332–1341, 1976.

[32] N.-C. F. Bagger, S. Kristiansen, M. Sørensen, and T. R. Stidsen, “Flow formulations for curriculum-based course timetabling,” *Annals OR*, vol. 280, no. 1-2, pp. 121–150, 2019.

[33] A. E. Phillips, C. G. Walker, M. Ehrgott, and D. M. Ryan, “Integer programming for minimal perturbation problems in university course timetabling,” *Annals OR*, vol. 252, no. 2, pp. 283–304, 2017.

[34] V. Pereira and H. G. Costa, “Linear integer model for the course timetabling problem of a faculty in rio de janeiro,” *Adv. Operations Research*, vol. 2016, 7597062:1–7597062:9, 2016.

[35] I. Méndez-Díaz, P. Zabala, and J. J. M. Bront, “An ILP based heuristic for a generalization of the post-enrollment course timetabling problem,” *Comput. Oper. Res.*, vol. 76, pp. 195–207, 2016.

[36] G. Lach and M. E. Lübbecke, “Curriculum based course timetabling: New solutions to udine benchmark instances,” *Annals OR*, vol. 194, no. 1, pp. 255–272, 2012.

[37] J.-K. Hao and U. Benlic, “Lower bounds for the ITC-2007 curriculum-based course timetabling problem,” *Eur. J. Oper. Res.*, vol. 212, no. 3, pp. 464–472, 2011.

[38] L. Kang, “A logic approach to the resolution of constraints in timetabling,” *European Journal of Operational Research*, vol. 61, no. 3, pp. 306–317, 1992.

[39] J. Dinkel, J. Mote, and M. Venkataramanan, “An efficient decision support system for academic course scheduling,” *Operations Research*, vol. 37, no. 6, pp. 853–864, 1989.

[40] S. Kassicieh, D. Burleson, and R. Lievano, “Design and implementation of a decision support system for academic scheduling,” *Information and Management*, vol. 11, no. 2, pp. 57–64, 1986.

[41] D. Mathaisel and C. Comm, "Course and classroom scheduling: An interactive computer graphics approach," *Journal of Systems and Software*, vol. 15, no. 2, pp. 149–157, 1991.

[42] D. Anderson, D. Sweeney, and T. Williams, *An Introduction to Management Science: Quantitative Approaches to Decision Making*. Minneapolis, St. Paul: West Publishing, 1994.

[43] S. Lee, *Goal Programming for Decision Analysis*. Philadelphia, PA: Auberback Publishers, 1972.

[44] S. Lee and M. Schniederjans, "Multicriteria assignment problem: A goal programming approach," *Interfaces*, vol. 13, pp. 75–81, 1983.

[45] M. Schniederjans, N. Kwak, and C. Helmer, "An application of goal programming to resolve a site location problem," *Interfaces*, vol. 12, pp. 65–72, 1982.

[46] L. Zhipeng and H. Jin-Kao, "Adaptive tabu search for course timetabling," *European Journal of Operational Research*, pp. 235–244, 2010.

[47] T. Dokeroglu and E. Sevinc, "Memetic teaching–learning-based optimization algorithms for large graph coloring problems," *Engineering Applications of Artificial Intelligence*, vol. 102, p. 104 282, 2021.

[48] P. Yasari, M. Ranjbar, N. Jamili, and M.-H. Shaelaie, "A two-stage stochastic programming approach for a multi-objective course timetabling problem with courses cancelation risk," *Comput. Ind. Eng.*, vol. 130, pp. 650–660, 2019.

[49] A. Gülcü and C. Akkan, "Robust university course timetabling problem subject to single and multiple disruptions," *Eur. J. Oper. Res.*, vol. 283, no. 2, pp. 630–646, 2020.

[50] M. Banbara, K. Inoue, B. Kaufmann, T. Okimoto, T. Schaub, T. Soh, N. Tamura, and P. Wanko, "Teaspoon : Solving the curriculum-based course timetabling problems with answer set programming," *Annals OR*, vol. 275, no. 1, pp. 3–37, 2019.

[51] M. Banbara, T. Soh, N. Tamura, K. Inoue, and T. Schaub, "Answer set programming as a modeling language for course timetabling," *Theory Pract. Log. Program.*, vol. 13, no. 4-5, pp. 783–798, 2013.

[52] C. Akkan and A. Gülcü, "A bi-criteria hybrid genetic algorithm with robustness objective for the course timetabling problem," *Comput. Oper. Res.*, vol. 90, pp. 22–32, 2018.

[53] T. Thepphakorn and P. Pongcharoen, "Variants and parameters investigations of particle swarm optimisation for solving course timetabling problems," in *Advances in Swarm Intelligence - 10th International Conference, ICSI 2019, Chiang Mai, Thailand, July 26-30, 2019, Proceedings, Part I*, Y. Tan, Y. Shi, and B. Niu, Eds., ser. Lecture Notes in Computer Science, vol. 11655, Springer, 2019, pp. 177–187.

[54] K. Wang, W. Shang, M. Liu, W. Lin, and H. Fu, "A greedy and genetic fusion algorithm for solving course timetabling problem," in *17th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2018, Singapore, Singapore, June 6-8, 2018*, IEEE Computer Society, 2018, pp. 344–349.

[55] M. E. Halaby, "Solving the course-timetabling problem of cairo university using max-sat," *CoRR*, vol. abs/1803.05027, 2018.

[56] T. Müller and H. Rudová, "Real-life curriculum-based timetabling with elective courses and course sections," *Annals OR*, vol. 239, no. 1, pp. 153–170, 2016.

[57] S. L. Goh, G. Kendall, and N. R. Sabar, "Simulated annealing with improved reheating and learning for the post enrolment course timetabling problem," *JORS*, vol. 70, no. 6, pp. 873–888, 2019.

[58] N. Rangel-Valdez, J. O. Jasso-Luna, M. H. Rodriguez-Chavez, and G. Bujano-Guzman, "Practical relaxation of a special case of the curriculum-based course timetabling problem," *Progress in AI*, vol. 2, no. 4, pp. 237–248, 2014.

[59] Y. Nagata and I. Ono, "Random partial neighborhood search for university course timetabling problem," in *Parallel Problem Solving from Nature - PPSN XIII - 13th International Conference, Ljubljana, Slovenia, September 13-17, 2014. Proceedings*, vol. 8672, Springer, 2014, pp. 782–791.

[60] Y. Nagata, "Random partial neighborhood search for the post-enrollment course timetabling problem," *Comput. Oper. Res.*, vol. 90, pp. 84–96, 2018.

[61] H. Wu, A. Lin, D. Zhang, and C. P. Mukamakuza, "A new time-dependent algorithm for post enrolment-based course timetabling problem," in *Eighth International Conference on Advanced Computational Intelligence, ICACI 2016, Chiang Mai, Thailand, February 14-16, 2016*, IEEE, 2016, pp. 425–431.

[62] T. Song, S. Liu, X. Tang, X. Peng, and M. Chen, "An iterated local search algorithm for the university course timetabling problem," *Appl. Soft Comput.*, vol. 68, pp. 597–608, 2018.

[63] S. L. Goh, G. Kendall, and N. R. Sabar, "Improved local search approaches to solve the post enrolment course timetabling problem," *Eur. J. Oper. Res.*, vol. 261, no. 1, pp. 17–29, 2017.

[64] N.-C. F. Bagger, M. Sørensen, and T. R. Stidsen, "Benders' decomposition for curriculum-based course timetabling," *Comput. Oper. Res.*, vol. 91, pp. 178–189, 2018.

[65] M. Mühlenthaler and R. Wanka, "Fairness in academic course timetabling," *Annals OR*, vol. 239, no. 1, pp. 171–188, 2016.

[66] M. Mühlenthaler and R. Wanka, "A decomposition of the max-min fair curriculum-based course timetabling problem," *CoRR*, vol. abs/1306.5601, 2013.

[67] P. Kenekayoro and G. Zipamone, "Greedy ants colony optimization strategy for solving the curriculum based university course timetabling problem," *CoRR*, vol. abs/1602.04933, 2016.

[68] R. Bellio, S. Ceschia, L. D. Gaspero, A. Schaerf, and T. Urli, "Feature-based tuning of simulated annealing applied to the curriculum-based course timetabling problem," *CoRR*, vol. abs/1409.7186, 2014.

[69] R. Bellio, L. D. Gaspero, and A. Schaerf, "Design and statistical analysis of a hybrid local search algorithm for course timetabling," *J. Scheduling*, vol. 15, no. 1, pp. 49–61, 2012.

[70] M.-X. Zhang, B. Zhang, and N. Qian, "University course timetabling using a new ecogeography-based optimization algorithm," *Natural Computing*, vol. 16, no. 1, pp. 61–74, 2017.

[71] R. A. Aziz, M. Ayob, Z. Othman, Z. Ahmad, and N. R. Sabar, "An adaptive guided variable neighborhood search based on honey-bee mating optimization algorithm for the course timetabling problem," *Soft Comput.*, vol. 21, no. 22, pp. 6755–6765, 2017.

[72] J. A. Soria-Alcaraz, J. M. Carpio, H. Puga, P. Melin, H. Terashima-Marín, L. Cruz Reyes, and M. A. Sotelo-Figueroa, "Generic memetic algorithm for course timetabling ITC2007," in *Recent Advances on Hybrid Approaches for Designing Intelligent Systems*, ser. Studies in Computational Intelligence, vol. 547, Springer, 2014, pp. 481–492.

[73] R. Lewis and J. M. Thompson, "Analysing the effects of solution space connectivity with an effective metaheuristic for the course timetabling problem," *Eur. J. Oper. Res.*, vol. 240, no. 3, pp. 637–648, 2015.

[74] R. Lewis, "A time-dependent metaheuristic algorithm for post enrolment-based course timetabling," *Annals OR*, vol. 194, no. 1, pp. 273–289, 2012.

[75] R. P. Badoni, D. K. Gupta, and P. Mishra, "A new hybrid algorithm for university course timetabling problem using events based on groupings of students," *Comput. Ind. Eng.*, vol. 78, pp. 12–25, 2014.

[76] L. Wu, "The application of coarse-grained parallel genetic algorithm with hadoop in university intelligent course-timetabling system," *iJET*, vol. 10, no. 8, pp. 11–15, 2015.

[77] J. A. Soria-Alcaraz, J. M. C. Valadez, H. Puga, J. Swan, P. Melin, H. Terashima, and M. A. Sotelo-Figueroa, "Parallel meta-heuristic approaches to the course timetabling problem," in *Design of Intelligent Systems Based on Fuzzy Logic, Neural Networks and Nature-Inspired Optimization*, ser. Studies in Computational Intelligence, vol. 601, Springer, 2015, pp. 391–417.

[78] J. A. Soria-Alcaraz, G. Ochoa, J. Swan, J. M. Carpio, H. Puga, and E. K. Burke, "Effective learning hyper-heuristics for the course timetabling problem," *Eur. J. Oper. Res.*, vol. 238, no. 1, pp. 77–86, 2014.

[79] R. J. A. Achá and R. Nieuwenhuis, "Curriculum-based course timetabling with SAT and maxsat," *Annals OR*, vol. 218, no. 1, pp. 71–91, 2014.

[80] K. Shaker, S. Abdullah, A. Alqudsi, and H. Jalab, "Hybridizing meta-heuristics approaches for solving university course timetabling problems," in *Rough Sets and Knowledge Technology - 8th International Conference, RSKT 2013, Halifax, NS, Canada, October 11-14, 2013, Proceedings*, vol. 8171, Springer, 2013, pp. 374–384.

[81] A. Abuhamdah, M. Ayob, G. Kendall, and N. R. Sabar, "Population based local search for university course timetabling problems," *Appl. Intell.*, vol. 40, no. 1, pp. 44–53, 2014.

[82] M. Kalender, A. Kheiri, E. Ozcan, and E. K. Burke, "A greedy gradient-simulated annealing hyper-heuristic for a curriculum-based course timetabling problem," in *12th UK Workshop on Computational Intelligence, UKCI 2012, Edinburgh, United Kingdom, September 5-7, 2012*, IEEE, 2012, pp. 1–8.

[83] G. M. Jaradat, M. Ayob, and Z. Ahmad, "On the performance of scatter search for post-enrolment course timetabling problems," *J. Comb. Optim.*, vol. 27, no. 3, pp. 417–439, 2014.

[84] A. L. Bolaji, A. T. Khader, M. A. Al-Betar, and M. A. Awadallah, "University course timetabling using hybridized artificial bee colony with hill climbing optimizer," *J. Comput. Sci.*, vol. 5, no. 5, pp. 809–818, 2014.

[85] H. Cambazard, E. Hebrard, B. O'Sullivan, and A. Papadopoulos, "Local search and constraint programming for the post enrolment-based course timetabling problem," *Annals OR*, vol. 194, no. 1, pp. 111–135, 2012.

[86] C. Nothegger, A. Mayer, A. M. Chwatal, and G. R. Raidl, "Solving the post enrolment course timetabling problem by ant colony optimization," *Annals OR*, vol. 194, no. 1, pp. 325–339, 2012.

[87] A. Gunawan, K. M. Ng, and K.-L. Poh, "A hybridized lagrangian relaxation and simulated annealing method for the course timetabling problem," *Comput. Oper. Res.*, vol. 39, no. 12, pp. 3074–3088, 2012.

[88] M. A. Al-Betar, A. T. Khader, and M. Zaman, "University course timetabling using a hybrid harmony search metaheuristic algorithm," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 42, no. 5, pp. 664–681, 2012.

[89] R.-M. Chen and H.-F. Shih, "Solving university course timetabling problems using constriction particle swarm optimization with local search," *Algorithms*, vol. 6, no. 2, pp. 227–244, 2013.

[90] D. Qaurooni and M. R. Akbarzadeh-Totonchi, "Course timetabling using evolutionary operators," *Appl. Soft Comput.*, vol. 13, no. 5, pp. 2504–2514, 2013.

[91] C.-C. Wu, "Parallelizing a clips-based course timetabling expert system," *Expert Syst. Appl.*, vol. 38, no. 6, pp. 7517–7525, 2011.

[92] H. Asmuni, "Fuzzy methodologies for automated university timetabling solution construction and evaluation," Ph.D. dissertation, University of Nottingham, 2008.

[93] J. Henry Obit, "Developing novel meta-heuristic, hyper-heuristic and cooperative search for course timetabling problems," Ph.D. dissertation, University of Nottingham, 2010.

[94] T. A. Redl, "A study of university timetabling that blends graph coloring with the satisfaction of various essential and preferential conditions," Tech. Rep., 2004.

[95] H. Babaei, J. Karimpour, and A. Hadidi, "A survey of approaches for university course timetabling problem," *Computers & Industrial Engineering*, vol. 86, pp. 43–59, 2015.

[96] T. Müller, "Itc2007 solver description: A hybrid approach," *Annals of Operations Research*, vol. 172, no. 1, p. 429, 2009.

[97] M. Atsuta, K. Nonobe, and T. Ibaraki, "Itc-2007 track2: An approach using general csp solver," 2008.

[98] M. Clark, M. Henz, and B. Love, "Quikfix—a repair-based timetable solver," in *Proceedings of the Seventh International Conference on the Practice and Theory of Automated Timetabling,*, Citeseer, 2008.

[99] J. Xu and J. Zhang, "Exploration-exploitation tradeoffs in metaheuristics: Survey and analysis," in *Proceedings of the 33rd Chinese control conference*, IEEE, 2014, pp. 8633–8638.

[100] Y. Borenstein and R. Poli, "Information landscapes," in *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, 2005, pp. 1515–1522.

[101] B. Prieve and R. Fabry, "Vmin—an optimal variable-space page replacement algorithm," *Communications of the ACM*, vol. 19, no. 5, pp. 296–297, 1976.

[102] E. Sevinc and T. Dokeroglu, "A novel parallel local search algorithm for the maximum vertex weight clique problem in large graphs," *Soft Computing*, vol. 24, no. 5, pp. 3551–3567, 2020.

APPENDIX A

The results of the experiments with 21 benchmark problem instances using our proposed greedy algorithm, Greedy-CB-CTT

Table A.1: Summary execution results for Heur-1 1/1.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	4	220	38	3	265
comp02	8	0	0	0	0	335	404	26	773
comp03	3	0	0	0	0	225	424	21	673
comp04	0	0	0	0	0	290	308	17	615
comp05	4	0	0	0	10	225	1240	9	1488
comp06	1	0	0	0	0	370	666	35	1072
comp07	0	0	0	0	0	340	630	35	1005
comp08	1	0	0	0	0	255	432	18	706
comp09	2	0	0	0	0	220	588	27	837
comp10	0	0	0	0	0	295	624	27	946
comp11	0	0	0	0	0	155	92	4	251
comp12	0	0	0	0	4	235	1260	14	1513
comp13	0	0	0	0	0	255	540	22	817
comp14	2	0	0	0	0	210	564	25	801
comp15	3	0	0	0	0	225	424	21	673
comp16	2	0	0	0	0	330	584	25	941
comp17	1	0	0	0	0	310	588	30	929
comp18	0	0	0	0	0	295	160	8	463
comp19	2	0	0	0	0	230	410	30	672
comp20	0	0	0	0	0	365	612	42	1019
comp21	4	0	0	0	0	380	522	25	931

Table A.2: Summary execution results for Heur-1 1/2.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2665	145	34	4	2848
comp02	14	0	0	0	4513	265	498	19	5309
comp03	9	0	0	0	2885	250	442	20	3606
comp04	7	0	0	0	3368	305	364	18	4062
comp05	9	0	0	0	3400	220	1052	6	4687
comp06	13	0	0	0	5137	380	670	21	6221
comp07	16	0	0	0	4990	350	678	26	6060
comp08	0	0	0	0	4078	180	428	18	4704
comp09	7	0	0	0	2525	265	540	16	3353
comp10	11	0	0	0	4992	375	554	19	5951
comp11	0	0	0	0	1870	140	68	2	2080
comp12	8	0	0	0	2094	280	1468	17	3867
comp13	4	0	0	0	4265	315	454	19	5057
comp14	1	0	0	0	2640	205	526	22	3394
comp15	9	0	0	0	2885	250	442	20	3606
comp16	11	0	0	0	4114	385	624	21	5155
comp17	8	0	0	0	5327	285	662	21	6303
comp18	0	0	0	0	810	305	180	6	1301
comp19	15	0	0	0	3041	245	486	19	3806
comp20	11	0	0	0	6904	470	534	32	7951
comp21	8	0	0	0	3795	305	524	18	4650

Table A.3: Summary execution results for Heur-1 1/3.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	7	0	0	0	2761	185	38	7	2998
comp02	22	0	0	0	6965	320	502	16	7825
comp03	13	0	0	0	4580	225	550	17	5385
comp04	11	0	0	0	4451	285	388	12	5147
comp05	13	0	0	0	4665	195	1076	8	5957
comp06	12	0	0	0	7255	360	588	24	8239
comp07	13	0	0	0	7123	320	604	32	8092
comp08	5	0	0	0	4926	230	512	14	5687
comp09	11	0	0	0	3217	255	614	11	4108
comp10	15	0	0	0	6114	380	638	27	7174
comp11	0	0	0	0	2251	110	90	2	2453
comp12	14	0	0	0	1816	295	1414	13	3552
comp13	7	0	0	0	6172	320	528	12	7039
comp14	7	0	0	0	3091	260	524	23	3905
comp15	13	0	0	0	4580	225	550	17	5385
comp16	15	0	0	0	4964	350	696	22	6047
comp17	8	0	0	0	6333	375	588	22	7326
comp18	1	0	0	0	830	315	236	4	1386
comp19	17	0	0	0	4179	275	444	24	4939
comp20	13	0	0	0	9005	450	644	24	10136
comp21	6	0	0	0	4525	355	434	14	5334

Table A.4: Summary execution results for Heur-1 1/4.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	3186	200	32	6	3424
comp02	12	0	0	0	4045	265	586	28	4936
comp03	7	0	0	0	1677	215	444	15	2358
comp04	10	0	0	0	3388	310	350	16	4074
comp05	10	0	0	0	3070	245	944	8	4277
comp06	8	0	0	0	5435	350	632	28	6453
comp07	15	0	0	0	5872	370	612	27	6896
comp08	12	0	0	0	3579	255	484	18	4348
comp09	9	0	0	0	1970	205	528	26	2738
comp10	7	0	0	0	5602	290	594	31	6524
comp11	0	0	0	0	1980	140	80	3	2203
comp12	4	0	0	0	1570	260	1286	14	3134
comp13	14	0	0	0	4380	330	536	17	5277
comp14	1	0	0	0	806	245	530	19	1601
comp15	7	0	0	0	1677	215	444	15	2358
comp16	10	0	0	0	3087	320	590	22	4029
comp17	8	0	0	0	4035	345	486	26	4900
comp18	0	0	0	0	740	295	148	7	1190
comp19	10	0	0	0	2085	240	420	21	2776
comp20	13	0	0	0	7896	435	618	26	8988
comp21	11	0	0	0	3888	345	544	21	4809

Table A.5: Summary execution results for Heur-1 1/5.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	4	195	30	5	234
comp02	8	0	0	0	0	240	442	28	718
comp03	1	0	0	0	0	205	424	27	657
comp04	0	0	0	0	0	300	372	20	692
comp05	4	0	0	0	5	180	1298	11	1498
comp06	2	0	0	0	0	360	618	31	1011
comp07	1	0	0	0	0	335	626	37	999
comp08	1	0	0	0	0	225	406	24	656
comp09	1	0	0	0	0	205	518	26	750
comp10	0	0	0	0	0	335	548	28	911
comp11	0	0	0	0	0	150	84	3	237
comp12	1	0	0	0	4	235	1338	18	1596
comp13	0	0	0	0	0	240	496	20	756
comp14	0	0	0	0	0	255	442	17	714
comp15	1	0	0	0	0	205	424	27	657
comp16	2	0	0	0	0	345	628	23	998
comp17	2	0	0	0	0	325	576	29	932
comp18	0	0	0	0	0	290	184	5	479
comp19	3	0	0	0	0	250	426	33	712
comp20	0	0	0	0	0	410	624	39	1073
comp21	4	0	0	0	0	360	502	28	894

Table A.6: Summary execution results for Heur-1 1/6.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	334	190	30	4	558
comp02	3	0	0	0	3120	295	430	33	3881
comp03	3	0	0	0	827	265	350	30	1475
comp04	0	0	0	0	920	285	416	20	1641
comp05	4	0	0	0	3405	225	1120	11	4765
comp06	1	0	0	0	2127	335	734	21	3218
comp07	0	0	0	0	2044	355	546	33	2978
comp08	0	0	0	0	880	280	470	28	1658
comp09	0	0	0	0	207	210	536	23	976
comp10	2	0	0	0	1473	335	564	38	2412
comp11	0	0	0	0	175	155	84	3	417
comp12	2	0	0	0	1804	245	1130	17	3198
comp13	0	0	0	0	1036	225	566	32	1859
comp14	0	0	0	0	698	190	468	28	1384
comp15	3	0	0	0	827	265	350	30	1475
comp16	3	0	0	0	1690	325	666	24	2708
comp17	1	0	0	0	1735	310	566	27	2639
comp18	0	0	0	0	0	310	172	7	489
comp19	6	0	0	0	505	235	400	29	1175
comp20	1	0	0	0	2425	320	628	34	3408
comp21	1	0	0	0	1130	360	500	27	2018

Table A.7: Summary execution results for Heur-1 1/7.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	1871	200	22	5	2098
comp02	6	0	0	0	1203	240	486	28	1963
comp03	1	0	0	0	1380	215	468	19	2083
comp04	0	0	0	0	1258	240	472	18	1988
comp05	2	0	0	0	80	195	1206	10	1493
comp06	1	0	0	0	2251	305	650	29	3236
comp07	2	0	0	0	2487	300	674	46	3509
comp08	1	0	0	0	480	225	502	28	1236
comp09	3	0	0	0	498	250	534	21	1306
comp10	1	0	0	0	1625	350	542	34	2552
comp11	0	0	0	0	1540	135	102	3	1780
comp12	5	0	0	0	1204	195	1364	17	2785
comp13	0	0	0	0	1019	300	510	25	1854
comp14	0	0	0	0	1278	200	518	28	2024
comp15	1	0	0	0	1380	215	468	19	2083
comp16	0	0	0	0	2233	310	536	25	3104
comp17	2	0	0	0	2000	300	550	27	2879
comp18	2	0	0	0	0	325	110	7	444
comp19	7	0	0	0	942	220	498	23	1690
comp20	0	0	0	0	3691	320	622	33	4666
comp21	3	0	0	0	1660	280	516	26	2485

Table A.8: Summary execution results for Heur-1 1/8.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	1228	135	40	7	1410
comp02	3	0	0	0	2183	195	490	37	2908
comp03	2	0	0	0	1223	295	422	24	1966
comp04	0	0	0	0	1105	240	440	24	1809
comp05	3	0	0	0	750	210	1202	9	2174
comp06	0	0	0	0	2133	325	674	32	3164
comp07	0	0	0	0	1940	415	588	43	2986
comp08	0	0	0	0	615	225	458	26	1324
comp09	1	0	0	0	619	220	498	21	1359
comp10	0	0	0	0	1482	345	618	35	2480
comp11	0	0	0	0	235	115	78	3	431
comp12	2	0	0	0	1733	250	1330	20	3335
comp13	0	0	0	0	1108	275	542	24	1949
comp14	0	0	0	0	604	185	502	27	1318
comp15	2	0	0	0	1223	295	422	24	1966
comp16	0	0	0	0	1771	300	700	31	2802
comp17	0	0	0	0	1956	265	524	31	2776
comp18	0	0	0	0	75	300	180	8	563
comp19	3	0	0	0	463	185	426	32	1109
comp20	2	0	0	0	2626	330	598	35	3591
comp21	3	0	0	0	1177	290	636	27	2133

Table A.9: Summary execution results for Heur-1 1/9.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	1673	150	44	9	1876
comp02	16	0	0	0	4268	270	492	26	5072
comp03	6	0	0	0	2380	225	388	10	3009
comp04	9	0	0	0	3428	305	414	18	4174
comp05	8	0	0	0	3725	250	848	8	4839
comp06	5	0	0	0	5727	355	650	34	6771
comp07	13	0	0	0	5773	370	570	38	6764
comp08	12	0	0	0	3708	250	482	17	4469
comp09	6	0	0	0	2015	200	562	18	2801
comp10	9	0	0	0	5439	290	606	38	6382
comp11	0	0	0	0	810	100	120	4	1034
comp12	8	0	0	0	1180	285	1250	15	2738
comp13	15	0	0	0	4142	295	546	15	5013
comp14	4	0	0	0	837	245	458	19	1563
comp15	6	0	0	0	2380	225	388	10	3009
comp16	13	0	0	0	2692	375	610	26	3716
comp17	11	0	0	0	4013	320	554	32	4930
comp18	0	0	0	0	750	285	170	6	1211
comp19	5	0	0	0	2534	175	424	30	3168
comp20	7	0	0	0	8041	350	690	26	9114
comp21	11	0	0	0	3329	390	580	19	4329

Table A.10: Summary execution results for Heur-1 1/10.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2260	235	34	4	2533
comp02	6	0	0	0	2652	265	562	27	3512
comp03	6	0	0	0	1213	215	500	20	1954
comp04	0	0	0	0	1012	265	392	16	1685
comp05	6	0	0	0	2135	200	1210	15	3566
comp06	8	0	0	0	3016	395	628	26	4073
comp07	8	0	0	0	1922	390	632	34	2986
comp08	3	0	0	0	692	235	530	24	1484
comp09	6	0	0	0	700	220	614	19	1559
comp10	5	0	0	0	1749	370	604	22	2750
comp11	0	0	0	0	1951	155	104	3	2213
comp12	9	0	0	0	1339	255	1454	14	3071
comp13	4	0	0	0	1071	290	484	19	1868
comp14	1	0	0	0	841	190	508	18	1558
comp15	6	0	0	0	1213	215	500	20	1954
comp16	8	0	0	0	1708	340	630	23	2709
comp17	4	0	0	0	1684	360	528	19	2595
comp18	2	0	0	0	140	290	156	4	592
comp19	7	0	0	0	1319	165	522	22	2035
comp20	10	0	0	0	2660	395	590	30	3685
comp21	8	0	0	0	882	345	468	26	1729

Table A.11: Summary execution results for Heur-1 2/1.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	3606	220	38	3	3867
comp02	8	0	0	0	13969	335	404	26	14742
comp03	3	0	0	0	11134	225	424	21	11807
comp04	0	0	0	0	8151	290	308	17	8766
comp05	4	0	0	0	10714	225	1240	9	12192
comp06	1	0	0	0	10612	370	666	35	11684
comp07	0	0	0	0	9760	340	630	35	10765
comp08	1	0	0	0	7711	255	432	18	8417
comp09	2	0	0	0	9165	220	588	27	10002
comp10	0	0	0	0	9296	295	624	27	10242
comp11	0	0	0	0	3196	155	92	4	3447
comp12	0	0	0	0	4510	235	1260	14	6019
comp13	0	0	0	0	10668	255	540	22	11485
comp14	2	0	0	0	7128	210	564	25	7929
comp15	3	0	0	0	11134	225	424	21	11807
comp16	2	0	0	0	9274	330	584	25	10215
comp17	1	0	0	0	10174	310	588	30	11103
comp18	0	0	0	0	2526	295	160	8	2989
comp19	2	0	0	0	10072	230	410	30	10744
comp20	0	0	0	0	12338	365	612	42	13357
comp21	4	0	0	0	10687	380	522	25	11618

Table A.12: Summary execution results for Heur-1 2/2.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	1350	145	34	4	1533
comp02	14	0	0	0	7977	265	498	19	8773
comp03	9	0	0	0	7029	250	442	20	7750
comp04	7	0	0	0	6871	305	364	18	7565
comp05	9	0	0	0	7854	220	1052	6	9141
comp06	13	0	0	0	5281	380	670	21	6365
comp07	16	0	0	0	4164	350	678	26	5234
comp08	0	0	0	0	5271	180	428	18	5897
comp09	7	0	0	0	6413	265	540	16	7241
comp10	11	0	0	0	3433	375	554	19	4392
comp11	0	0	0	0	1986	140	68	2	2196
comp12	8	0	0	0	2713	280	1468	17	4486
comp13	4	0	0	0	8867	315	454	19	9659
comp14	1	0	0	0	5013	205	526	22	5767
comp15	9	0	0	0	7029	250	442	20	7750
comp16	11	0	0	0	5217	385	624	21	6258
comp17	8	0	0	0	3783	285	662	21	4759
comp18	0	0	0	0	2210	305	180	6	2701
comp19	15	0	0	0	4957	245	486	19	5722
comp20	11	0	0	0	6986	470	534	32	8033
comp21	8	0	0	0	5640	305	524	18	6495

Table A.13: Summary execution results for Heur-1 2/3.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	7	0	0	0	670	185	38	7	907
comp02	22	0	0	0	3370	320	502	16	4230
comp03	13	0	0	0	3591	225	550	17	4396
comp04	11	0	0	0	4873	285	388	12	5569
comp05	13	0	0	0	6608	195	1076	8	7900
comp06	12	0	0	0	877	360	588	24	1861
comp07	13	0	0	0	371	320	604	32	1340
comp08	5	0	0	0	2980	230	512	14	3741
comp09	11	0	0	0	3500	255	614	11	4391
comp10	15	0	0	0	269	380	638	27	1329
comp11	0	0	0	0	1306	110	90	2	1508
comp12	14	0	0	0	1769	295	1414	13	3505
comp13	7	0	0	0	6204	320	528	12	7071
comp14	7	0	0	0	2802	260	524	23	3616
comp15	13	0	0	0	3591	225	550	17	4396
comp16	15	0	0	0	2239	350	696	22	3322
comp17	8	0	0	0	481	375	588	22	1474
comp18	1	0	0	0	1648	315	236	4	2204
comp19	17	0	0	0	1450	275	444	24	2210
comp20	13	0	0	0	2467	450	644	24	3598
comp21	6	0	0	0	614	355	434	14	1423

Table A.14: Summary execution results for Heur-1 2/4.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	550	200	32	6	788
comp02	12	0	0	0	9526	265	586	28	10417
comp03	7	0	0	0	6756	215	444	15	7437
comp04	10	0	0	0	6045	310	350	16	6731
comp05	10	0	0	0	7908	245	944	8	9115
comp06	8	0	0	0	5880	350	632	28	6898
comp07	15	0	0	0	3580	370	612	27	4604
comp08	12	0	0	0	4404	255	484	18	5173
comp09	9	0	0	0	5351	205	528	26	6119
comp10	7	0	0	0	3682	290	594	31	4604
comp11	0	0	0	0	1936	140	80	3	2159
comp12	4	0	0	0	3517	260	1286	14	5081
comp13	14	0	0	0	7473	330	536	17	8370
comp14	1	0	0	0	5869	245	530	19	6664
comp15	7	0	0	0	6756	215	444	15	7437
comp16	10	0	0	0	6024	320	590	22	6966
comp17	8	0	0	0	4073	345	486	26	4938
comp18	0	0	0	0	1986	295	148	7	2436
comp19	10	0	0	0	6270	240	420	21	6961
comp20	13	0	0	0	6613	435	618	26	7705
comp21	11	0	0	0	4849	345	544	21	5770

Table A.15: Summary execution results for Heur-1 2/5.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	3606	195	30	5	3836
comp02	8	0	0	0	14007	240	442	28	14725
comp03	1	0	0	0	11132	205	424	27	11789
comp04	0	0	0	0	8151	300	372	20	8843
comp05	4	0	0	0	10624	180	1298	11	12117
comp06	2	0	0	0	10612	360	618	31	11623
comp07	1	0	0	0	9760	335	626	37	10759
comp08	1	0	0	0	7711	225	406	24	8367
comp09	1	0	0	0	9257	205	518	26	10007
comp10	0	0	0	0	9296	335	548	28	10207
comp11	0	0	0	0	3196	150	84	3	3433
comp12	1	0	0	0	4510	235	1338	18	6102
comp13	0	0	0	0	10668	240	496	20	11424
comp14	0	0	0	0	7138	255	442	17	7852
comp15	1	0	0	0	11132	205	424	27	11789
comp16	2	0	0	0	9354	345	628	23	10352
comp17	2	0	0	0	10219	325	576	29	11151
comp18	0	0	0	0	2526	290	184	5	3005
comp19	3	0	0	0	10081	250	426	33	10793
comp20	0	0	0	0	12338	410	624	39	13411
comp21	4	0	0	0	10680	360	502	28	11574

Table A.16: Summary execution results for Heur-1 2/6.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	3486	190	30	4	3710
comp02	3	0	0	0	13244	295	430	33	14005
comp03	3	0	0	0	9328	265	350	30	9976
comp04	0	0	0	0	7383	285	416	20	8104
comp05	4	0	0	0	9636	225	1120	11	10996
comp06	1	0	0	0	9375	335	734	21	10466
comp07	0	0	0	0	8329	355	546	33	9263
comp08	0	0	0	0	7161	280	470	28	7939
comp09	0	0	0	0	7558	210	536	23	8327
comp10	2	0	0	0	7786	335	564	38	8725
comp11	0	0	0	0	3111	155	84	3	3353
comp12	2	0	0	0	3573	245	1130	17	4967
comp13	0	0	0	0	10027	225	566	32	10850
comp14	0	0	0	0	6290	190	468	28	6976
comp15	3	0	0	0	9328	265	350	30	9976
comp16	3	0	0	0	7976	325	666	24	8994
comp17	1	0	0	0	7318	310	566	27	8222
comp18	0	0	0	0	2396	310	172	7	2885
comp19	6	0	0	0	8407	235	400	29	9077
comp20	1	0	0	0	11170	320	628	34	12153
comp21	1	0	0	0	8531	360	500	27	9419

Table A.17: Summary execution results for Heur-1 2/7.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2065	200	22	5	2292
comp02	6	0	0	0	12663	240	486	28	13423
comp03	1	0	0	0	8903	215	468	19	9606
comp04	0	0	0	0	7196	240	472	18	7926
comp05	2	0	0	0	10834	195	1206	10	12247
comp06	1	0	0	0	8866	305	650	29	9851
comp07	2	0	0	0	8284	300	674	46	9306
comp08	1	0	0	0	7047	225	502	28	7803
comp09	3	0	0	0	7256	250	534	21	8064
comp10	1	0	0	0	7829	350	542	34	8756
comp11	0	0	0	0	2064	135	102	3	2304
comp12	5	0	0	0	3650	195	1364	17	5231
comp13	0	0	0	0	9877	300	510	25	10712
comp14	0	0	0	0	5691	200	518	28	6437
comp15	1	0	0	0	8903	215	468	19	9606
comp16	0	0	0	0	7553	310	536	25	8424
comp17	2	0	0	0	7191	300	550	27	8070
comp18	2	0	0	0	2502	325	110	7	2946
comp19	7	0	0	0	8318	220	498	23	9066
comp20	0	0	0	0	11036	320	622	33	12011
comp21	3	0	0	0	7302	280	516	26	8127

Table A.18: Summary execution results for Heur-1 2/8.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	3266	135	40	7	3448
comp02	3	0	0	0	13063	195	490	37	13788
comp03	2	0	0	0	9755	295	422	24	10498
comp04	0	0	0	0	7300	240	440	24	8004
comp05	3	0	0	0	10868	210	1202	9	12292
comp06	0	0	0	0	9135	325	674	32	10166
comp07	0	0	0	0	8207	415	588	43	9253
comp08	0	0	0	0	7195	225	458	26	7904
comp09	1	0	0	0	7363	220	498	21	8103
comp10	0	0	0	0	7844	345	618	35	8842
comp11	0	0	0	0	3111	115	78	3	3307
comp12	2	0	0	0	3518	250	1330	20	5120
comp13	0	0	0	0	9949	275	542	24	10790
comp14	0	0	0	0	6201	185	502	27	6915
comp15	2	0	0	0	9755	295	422	24	10498
comp16	0	0	0	0	7874	300	700	31	8905
comp17	0	0	0	0	7518	265	524	31	8338
comp18	0	0	0	0	2396	300	180	8	2884
comp19	3	0	0	0	9057	185	426	32	9703
comp20	2	0	0	0	11199	330	598	35	12164
comp21	3	0	0	0	8283	290	636	27	9239

Table A.19: Summary execution results for Heur-1 2/9.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2737	150	44	9	2940
comp02	16	0	0	0	9520	270	492	26	10324
comp03	6	0	0	0	7669	225	388	10	8298
comp04	9	0	0	0	6134	305	414	18	6880
comp05	8	0	0	0	8158	250	848	8	9272
comp06	5	0	0	0	5966	355	650	34	7010
comp07	13	0	0	0	3812	370	570	38	4803
comp08	12	0	0	0	4426	250	482	17	5187
comp09	6	0	0	0	5143	200	562	18	5929
comp10	9	0	0	0	3673	290	606	38	4616
comp11	0	0	0	0	2591	100	120	4	2815
comp12	8	0	0	0	3451	285	1250	15	5009
comp13	15	0	0	0	7482	295	546	15	8353
comp14	4	0	0	0	6209	245	458	19	6935
comp15	6	0	0	0	7669	225	388	10	8298
comp16	13	0	0	0	6241	375	610	26	7265
comp17	11	0	0	0	4895	320	554	32	5812
comp18	0	0	0	0	1936	285	170	6	2397
comp19	5	0	0	0	6780	175	424	30	7414
comp20	7	0	0	0	6628	350	690	26	7701
comp21	11	0	0	0	5704	390	580	19	6704

Table A.20: Summary execution results for Heur-1 2/10.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	1708	235	34	4	1981
comp02	6	0	0	0	10240	265	562	27	11100
comp03	6	0	0	0	8653	215	500	20	9394
comp04	0	0	0	0	7405	265	392	16	8078
comp05	6	0	0	0	9032	200	1210	15	10463
comp06	8	0	0	0	7480	395	628	26	8537
comp07	8	0	0	0	7809	390	632	34	8873
comp08	3	0	0	0	6836	235	530	24	7628
comp09	6	0	0	0	7550	220	614	19	8409
comp10	5	0	0	0	7088	370	604	22	8089
comp11	0	0	0	0	1959	155	104	3	2221
comp12	9	0	0	0	3562	255	1454	14	5294
comp13	4	0	0	0	9705	290	484	19	10502
comp14	1	0	0	0	5515	190	508	18	6232
comp15	6	0	0	0	8653	215	500	20	9394
comp16	8	0	0	0	7567	340	630	23	8568
comp17	4	0	0	0	8103	360	528	19	9014
comp18	2	0	0	0	2198	290	156	4	2650
comp19	7	0	0	0	7300	165	522	22	8016
comp20	10	0	0	0	9297	395	590	30	10322
comp21	8	0	0	0	8041	345	468	26	8888

Table A.21: Summary execution results for Heur-1 3/1.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2672	220	38	3	2933
comp02	8	0	0	0	9518	335	404	26	10291
comp03	3	0	0	0	6432	225	424	21	7105
comp04	0	0	0	0	5076	290	308	17	5691
comp05	4	0	0	0	6490	225	1240	9	7968
comp06	1	0	0	0	6617	370	666	35	7689
comp07	0	0	0	0	4664	340	630	35	5669
comp08	1	0	0	0	2978	255	432	18	3684
comp09	2	0	0	0	3450	220	588	27	4287
comp10	0	0	0	0	4721	295	624	27	5667
comp11	0	0	0	0	1710	155	92	4	1961
comp12	0	0	0	0	2095	235	1260	14	3604
comp13	0	0	0	0	4806	255	540	22	5623
comp14	2	0	0	0	3086	210	564	25	3887
comp15	3	0	0	0	6432	225	424	21	7105
comp16	2	0	0	0	2804	330	584	25	3745
comp17	1	0	0	0	4316	310	588	30	5245
comp18	0	0	0	0	490	295	160	8	953
comp19	2	0	0	0	5622	230	410	30	6294
comp20	0	0	0	0	8631	365	612	42	9650
comp21	4	0	0	0	4582	380	522	25	5513

Table A.22: Summary execution results for Heur-1 3/2.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2140	145	34	4	2323
comp02	14	0	0	0	4923	265	498	19	5719
comp03	9	0	0	0	1254	250	442	20	1975
comp04	7	0	0	0	5823	305	364	18	6517
comp05	9	0	0	0	4949	220	1052	6	6236
comp06	13	0	0	0	3997	380	670	21	5081
comp07	16	0	0	0	3601	350	678	26	4671
comp08	0	0	0	0	5717	180	428	18	6343
comp09	7	0	0	0	1318	265	540	16	2146
comp10	11	0	0	0	1956	375	554	19	2915
comp11	0	0	0	0	2267	140	68	2	2477
comp12	8	0	0	0	2750	280	1468	17	4523
comp13	4	0	0	0	7317	315	454	19	8109
comp14	1	0	0	0	4822	205	526	22	5576
comp15	9	0	0	0	1254	250	442	20	1975
comp16	11	0	0	0	5877	385	624	21	6918
comp17	8	0	0	0	6461	285	662	21	7437
comp18	0	0	0	0	1370	305	180	6	1861
comp19	15	0	0	0	2573	245	486	19	3338
comp20	11	0	0	0	5386	470	534	32	6433
comp21	8	0	0	0	3486	305	524	18	4341

Table A.23: Summary execution results for Heur-1 3/3.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	7	0	0	0	2783	185	38	7	3020
comp02	22	0	0	0	2787	320	502	16	3647
comp03	13	0	0	0	683	225	550	17	1488
comp04	11	0	0	0	6610	285	388	12	7306
comp05	13	0	0	0	5885	195	1076	8	7177
comp06	12	0	0	0	5201	360	588	24	6185
comp07	13	0	0	0	4511	320	604	32	5480
comp08	5	0	0	0	7179	230	512	14	7940
comp09	11	0	0	0	1811	255	614	11	2702
comp10	15	0	0	0	2135	380	638	27	3195
comp11	0	0	0	0	2917	110	90	2	3119
comp12	14	0	0	0	2670	295	1414	13	4406
comp13	7	0	0	0	9769	320	528	12	10636
comp14	7	0	0	0	6091	260	524	23	6905
comp15	13	0	0	0	683	225	550	17	1488
comp16	15	0	0	0	8050	350	696	22	9133
comp17	8	0	0	0	5546	375	588	22	6539
comp18	1	0	0	0	1652	315	236	4	2208
comp19	17	0	0	0	2111	275	444	24	2871
comp20	13	0	0	0	6717	450	644	24	7848
comp21	6	0	0	0	4824	355	434	14	5633

Table A.24: Summary execution results for Heur-1 3/4.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2680	200	32	6	2918
comp02	12	0	0	0	6226	265	586	28	7117
comp03	7	0	0	0	1779	215	444	15	2460
comp04	10	0	0	0	5861	310	350	16	6547
comp05	10	0	0	0	5895	245	944	8	7102
comp06	8	0	0	0	6894	350	632	28	7912
comp07	15	0	0	0	6509	370	612	27	7533
comp08	12	0	0	0	6001	255	484	18	6770
comp09	9	0	0	0	1170	205	528	26	1938
comp10	7	0	0	0	3516	290	594	31	4438
comp11	0	0	0	0	2448	140	80	3	2671
comp12	4	0	0	0	2482	260	1286	14	4046
comp13	14	0	0	0	7949	330	536	17	8846
comp14	1	0	0	0	5197	245	530	19	5992
comp15	7	0	0	0	1779	215	444	15	2460
comp16	10	0	0	0	6828	320	590	22	7770
comp17	8	0	0	0	3785	345	486	26	4650
comp18	0	0	0	0	1420	295	148	7	1870
comp19	10	0	0	0	2050	240	420	21	2741
comp20	13	0	0	0	8810	435	618	26	9902
comp21	11	0	0	0	2390	345	544	21	3311

Table A.25: Summary execution results for Heur-1 3/5.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2672	195	30	5	2902
comp02	8	0	0	0	9557	240	442	28	10275
comp03	1	0	0	0	6430	205	424	27	7087
comp04	0	0	0	0	5076	300	372	20	5768
comp05	4	0	0	0	6430	180	1298	11	7923
comp06	2	0	0	0	6623	360	618	31	7634
comp07	1	0	0	0	4680	335	626	37	5679
comp08	1	0	0	0	2978	225	406	24	3634
comp09	1	0	0	0	3503	205	518	26	4253
comp10	0	0	0	0	4721	335	548	28	5632
comp11	0	0	0	0	1710	150	84	3	1947
comp12	1	0	0	0	2095	235	1338	18	3687
comp13	0	0	0	0	4809	240	496	20	5565
comp14	0	0	0	0	3098	255	442	17	3812
comp15	1	0	0	0	6430	205	424	27	7087
comp16	2	0	0	0	2884	345	628	23	3882
comp17	2	0	0	0	4344	325	576	29	5276
comp18	0	0	0	0	490	290	184	5	969
comp19	3	0	0	0	5631	250	426	33	6343
comp20	0	0	0	0	8635	410	624	39	9708
comp21	4	0	0	0	4575	360	502	28	5469

Table A.26: Summary execution results for Heur-1 3/6.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2410	190	30	4	2634
comp02	3	0	0	0	5422	295	430	33	6183
comp03	3	0	0	0	4017	265	350	30	4665
comp04	0	0	0	0	5969	285	416	20	6690
comp05	4	0	0	0	7010	225	1120	11	8370
comp06	1	0	0	0	8739	335	734	21	9830
comp07	0	0	0	0	7193	355	546	33	8127
comp08	0	0	0	0	3957	280	470	28	4735
comp09	0	0	0	0	1974	210	536	23	2743
comp10	2	0	0	0	6503	335	564	38	7442
comp11	0	0	0	0	1105	155	84	3	1347
comp12	2	0	0	0	2693	245	1130	17	4087
comp13	0	0	0	0	7020	225	566	32	7843
comp14	0	0	0	0	4185	190	468	28	4871
comp15	3	0	0	0	4017	265	350	30	4665
comp16	3	0	0	0	5684	325	666	24	6702
comp17	1	0	0	0	3566	310	566	27	4470
comp18	0	0	0	0	665	310	172	7	1154
comp19	6	0	0	0	2798	235	400	29	3468
comp20	1	0	0	0	9777	320	628	34	10760
comp21	1	0	0	0	3774	360	500	27	4662

Table A.27: Summary execution results for Heur-1 3/7.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2232	200	22	5	2459
comp02	6	0	0	0	7247	240	486	28	8007
comp03	1	0	0	0	3611	215	468	19	4314
comp04	0	0	0	0	6407	240	472	18	7137
comp05	2	0	0	0	6690	195	1206	10	8103
comp06	1	0	0	0	7203	305	650	29	8188
comp07	2	0	0	0	6228	300	674	46	7250
comp08	1	0	0	0	4241	225	502	28	4997
comp09	3	0	0	0	2755	250	534	21	3563
comp10	1	0	0	0	5606	350	542	34	6533
comp11	0	0	0	0	2217	135	102	3	2457
comp12	5	0	0	0	2421	195	1364	17	4002
comp13	0	0	0	0	7168	300	510	25	8003
comp14	0	0	0	0	4942	200	518	28	5688
comp15	1	0	0	0	3611	215	468	19	4314
comp16	0	0	0	0	5634	310	536	25	6505
comp17	2	0	0	0	3871	300	550	27	4750
comp18	2	0	0	0	450	325	110	7	894
comp19	7	0	0	0	3748	220	498	23	4496
comp20	0	0	0	0	9469	320	622	33	10444
comp21	3	0	0	0	4796	280	516	26	5621

Table A.28: Summary execution results for Heur-1 3/8.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2141	135	40	7	2323
comp02	3	0	0	0	6793	195	490	37	7518
comp03	2	0	0	0	4310	295	422	24	5053
comp04	0	0	0	0	6147	240	440	24	6851
comp05	3	0	0	0	6960	210	1202	9	8384
comp06	0	0	0	0	8301	325	674	32	9332
comp07	0	0	0	0	6786	415	588	43	7832
comp08	0	0	0	0	4059	225	458	26	4768
comp09	1	0	0	0	2285	220	498	21	3025
comp10	0	0	0	0	5921	345	618	35	6919
comp11	0	0	0	0	1161	115	78	3	1357
comp12	2	0	0	0	2791	250	1330	20	4393
comp13	0	0	0	0	6938	275	542	24	7779
comp14	0	0	0	0	4678	185	502	27	5392
comp15	2	0	0	0	4310	295	422	24	5053
comp16	0	0	0	0	5769	300	700	31	6800
comp17	0	0	0	0	3668	265	524	31	4488
comp18	0	0	0	0	690	300	180	8	1178
comp19	3	0	0	0	4046	185	426	32	4692
comp20	2	0	0	0	9410	330	598	35	10375
comp21	3	0	0	0	3527	290	636	27	4483

Table A.29: Summary execution results for Heur-1 3/9.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2744	150	44	9	2947
comp02	16	0	0	0	6507	270	492	26	7311
comp03	6	0	0	0	2860	225	388	10	3489
comp04	9	0	0	0	6280	305	414	18	7026
comp05	8	0	0	0	6225	250	848	8	7339
comp06	5	0	0	0	7031	355	650	34	8075
comp07	13	0	0	0	6131	370	570	38	7122
comp08	12	0	0	0	6495	250	482	17	7256
comp09	6	0	0	0	1180	200	562	18	1966
comp10	9	0	0	0	3388	290	606	38	4331
comp11	0	0	0	0	2243	100	120	4	2467
comp12	8	0	0	0	2216	285	1250	15	3774
comp13	15	0	0	0	8603	295	546	15	9474
comp14	4	0	0	0	4584	245	458	19	5310
comp15	6	0	0	0	2860	225	388	10	3489
comp16	13	0	0	0	7057	375	610	26	8081
comp17	11	0	0	0	4112	320	554	32	5029
comp18	0	0	0	0	1468	285	170	6	1929
comp19	5	0	0	0	2990	175	424	30	3624
comp20	7	0	0	0	8591	350	690	26	9664
comp21	11	0	0	0	3006	390	580	19	4006

Table A.30: Summary execution results for Heur-1 3/10.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	2805	235	34	4	3078
comp02	6	0	0	0	6073	265	562	27	6933
comp03	6	0	0	0	3453	215	500	20	4194
comp04	0	0	0	0	6190	265	392	16	6863
comp05	6	0	0	0	6382	200	1210	15	7813
comp06	8	0	0	0	7029	395	628	26	8086
comp07	8	0	0	0	5134	390	632	34	6198
comp08	3	0	0	0	4208	235	530	24	5000
comp09	6	0	0	0	1954	220	614	19	2813
comp10	5	0	0	0	4523	370	604	22	5524
comp11	0	0	0	0	2738	155	104	3	3000
comp12	9	0	0	0	2438	255	1454	14	4170
comp13	4	0	0	0	6945	290	484	19	7742
comp14	1	0	0	0	5229	190	508	18	5946
comp15	6	0	0	0	3453	215	500	20	4194
comp16	8	0	0	0	5323	340	630	23	6324
comp17	4	0	0	0	4646	360	528	19	5557
comp18	2	0	0	0	1002	290	156	4	1454
comp19	7	0	0	0	2652	165	522	22	3368
comp20	10	0	0	0	7634	395	590	30	8659
comp21	8	0	0	0	3777	345	468	26	4624

Table A.31: Summary execution results for Heur-1 4/1.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	4	220	28	5	257
comp02	8	0	0	0	0	275	530	27	840
comp03	2	0	0	0	0	255	338	18	613
comp04	0	0	0	0	0	300	332	17	649
comp05	3	0	0	0	5	195	1304	4	1511
comp06	0	0	0	0	0	370	684	23	1077
comp07	1	0	0	0	0	345	616	36	998
comp08	0	0	0	0	0	250	464	23	737
comp09	4	0	0	0	0	265	510	19	798
comp10	0	0	0	0	0	315	636	25	976
comp11	0	0	0	0	0	145	66	1	212
comp12	2	0	0	0	4	245	1298	23	1572
comp13	0	0	0	0	0	285	486	20	791
comp14	1	0	0	0	0	270	464	17	752
comp15	2	0	0	0	0	255	338	18	613
comp16	0	0	0	0	0	310	556	33	899
comp17	1	0	0	0	0	350	616	23	990
comp18	0	0	0	0	0	295	144	2	441
comp19	3	0	0	0	0	245	450	22	720
comp20	0	0	0	0	0	445	616	30	1091
comp21	8	0	0	0	0	370	518	22	918

Table A.32: Summary execution results for Heur-1 4/2.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	2	0	0	0	348	215	36	6	607
comp02	12	0	0	0	0	275	570	18	875
comp03	11	0	0	0	0	255	386	12	664
comp04	1	0	0	0	0	230	470	21	722
comp05	9	0	0	0	200	200	1098	1	1508
comp06	10	0	0	0	0	365	684	22	1081
comp07	12	0	0	0	0	410	612	28	1062
comp08	0	0	0	0	0	245	492	22	759
comp09	7	0	0	0	0	230	520	16	773
comp10	8	0	0	0	0	335	588	26	957
comp11	0	0	0	0	0	125	90	1	216
comp12	10	0	0	0	0	310	1266	10	1596
comp13	2	0	0	0	0	310	454	13	779
comp14	1	0	0	0	0	240	580	20	841
comp15	11	0	0	0	0	255	386	12	664
comp16	12	0	0	0	0	370	666	24	1072
comp17	10	0	0	0	0	300	628	27	965
comp18	0	0	0	0	0	310	140	3	453
comp19	14	0	0	0	0	255	460	20	749
comp20	8	0	0	0	89	435	660	28	1220
comp21	9	0	0	0	0	350	528	22	909

Table A.33: Summary execution results for Heur-1 4/3.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	5	0	0	0	287	205	32	7	536
comp02	21	0	0	0	0	300	492	15	828
comp03	15	0	0	0	0	245	486	10	756
comp04	10	0	0	0	0	275	408	19	712
comp05	8	0	0	0	0	180	1328	3	1519
comp06	13	0	0	0	0	400	660	21	1094
comp07	12	0	0	0	0	340	632	31	1015
comp08	5	0	0	0	0	200	500	18	723
comp09	10	0	0	0	0	245	578	16	849
comp10	14	0	0	0	0	385	684	23	1106
comp11	0	0	0	0	0	115	74	1	190
comp12	11	0	0	0	0	295	1410	8	1724
comp13	7	0	0	0	0	335	542	13	897
comp14	2	0	0	0	0	235	542	22	801
comp15	15	0	0	0	0	245	486	10	756
comp16	18	0	0	0	0	365	658	18	1059
comp17	8	0	0	0	0	370	560	20	958
comp18	0	0	0	0	0	335	136	3	474
comp19	14	0	0	0	0	250	438	19	721
comp20	13	0	0	0	0	520	560	30	1123
comp21	6	0	0	0	0	380	422	12	820

Table A.34: Summary execution results for Heur-1 4/4.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	1	0	0	0	300	190	30	6	527
comp02	17	0	0	0	0	240	538	15	810
comp03	8	0	0	0	0	260	480	14	762
comp04	3	0	0	0	0	265	402	13	683
comp05	5	0	0	0	265	205	1222	3	1700
comp06	10	0	0	0	0	410	774	19	1213
comp07	13	0	0	0	0	380	656	26	1075
comp08	9	0	0	0	0	275	502	13	799
comp09	4	0	0	0	0	210	540	12	766
comp10	4	0	0	0	0	310	548	27	889
comp11	0	0	0	0	0	150	80	1	231
comp12	3	0	0	0	0	245	1308	10	1566
comp13	7	0	0	0	0	235	570	12	824
comp14	6	0	0	0	0	265	426	17	714
comp15	8	0	0	0	0	260	480	14	762
comp16	8	0	0	0	0	365	582	19	974
comp17	7	0	0	0	0	320	560	23	910
comp18	0	0	0	0	0	290	230	4	524
comp19	9	0	0	0	0	230	376	20	635
comp20	8	0	0	0	0	360	680	19	1067
comp21	9	0	0	0	0	370	548	24	951

Table A.35: Summary execution results for Heur-1 4/5.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	4	215	34	6	259
comp02	8	0	0	0	0	285	550	19	862
comp03	1	0	0	0	0	195	430	19	645
comp04	0	0	0	0	0	280	396	21	697
comp05	3	0	0	0	5	210	1162	5	1385
comp06	0	0	0	0	0	355	704	29	1088
comp07	1	0	0	0	0	315	574	40	930
comp08	0	0	0	0	0	240	486	23	749
comp09	0	0	0	0	0	205	568	15	788
comp10	1	0	0	0	0	370	604	26	1001
comp11	0	0	0	0	0	165	84	1	250
comp12	2	0	0	0	4	250	1272	21	1549
comp13	1	0	0	0	0	320	538	15	874
comp14	0	0	0	0	0	265	522	19	806
comp15	1	0	0	0	0	195	430	19	645
comp16	1	0	0	0	0	355	540	28	924
comp17	1	0	0	0	0	330	584	33	948
comp18	0	0	0	0	0	300	164	2	466
comp19	2	0	0	0	0	210	422	25	659
comp20	0	0	0	0	0	410	646	30	1086
comp21	4	0	0	0	0	365	530	21	920

Table A.36: Summary execution results for Heur-1 4/6.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	76	160	48	7	291
comp02	5	0	0	0	0	250	504	23	782
comp03	5	0	0	0	0	245	476	14	740
comp04	0	0	0	0	0	250	446	11	707
comp05	0	0	0	0	30	205	1076	5	1316
comp06	0	0	0	0	0	385	628	18	1031
comp07	0	0	0	0	0	365	628	29	1022
comp08	0	0	0	0	0	220	444	21	685
comp09	0	0	0	0	0	200	558	22	780
comp10	3	0	0	0	0	380	548	24	955
comp11	0	0	0	0	0	145	92	1	238
comp12	1	0	0	0	0	290	1094	13	1398
comp13	0	0	0	0	0	255	514	20	789
comp14	2	0	0	0	0	235	530	20	787
comp15	5	0	0	0	0	245	476	14	740
comp16	0	0	0	0	0	385	582	19	986
comp17	3	0	0	0	0	270	538	27	838
comp18	0	0	0	0	0	320	126	1	447
comp19	2	0	0	0	0	185	522	25	734
comp20	1	0	0	0	0	360	634	24	1019
comp21	3	0	0	0	0	335	518	18	874

Table A.37: Summary execution results for Heur-1 4/7.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	170	205	24	4	403
comp02	6	0	0	0	0	335	504	18	863
comp03	0	0	0	0	0	225	448	16	689
comp04	0	0	0	0	0	255	398	18	671
comp05	3	0	0	0	35	225	1264	5	1532
comp06	1	0	0	0	0	350	708	23	1082
comp07	0	0	0	0	0	285	712	36	1033
comp08	0	0	0	0	0	235	494	17	746
comp09	2	0	0	0	0	270	534	18	824
comp10	0	0	0	0	0	400	626	25	1051
comp11	0	0	0	0	0	150	90	1	241
comp12	3	0	0	0	0	270	1126	10	1409
comp13	0	0	0	0	0	335	512	16	863
comp14	0	0	0	0	0	265	500	17	782
comp15	0	0	0	0	0	225	448	16	689
comp16	1	0	0	0	0	355	578	25	959
comp17	1	0	0	0	0	275	566	29	871
comp18	0	0	0	0	0	335	112	1	448
comp19	6	0	0	0	0	235	464	15	720
comp20	0	0	0	0	0	355	572	26	953
comp21	4	0	0	0	0	345	552	23	924

Table A.38: Summary execution results for Heur-1 4/8.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	100	225	26	5	356
comp02	5	0	0	0	0	290	490	19	804
comp03	0	0	0	0	0	215	450	16	681
comp04	0	0	0	0	0	285	424	12	721
comp05	0	0	0	0	0	235	1048	4	1287
comp06	0	0	0	0	0	305	638	25	968
comp07	0	0	0	0	0	360	654	31	1045
comp08	0	0	0	0	0	275	486	17	778
comp09	3	0	0	0	0	260	482	15	760
comp10	0	0	0	0	0	310	594	24	928
comp11	0	0	0	0	0	155	72	1	228
comp12	0	0	0	0	0	245	1294	17	1556
comp13	0	0	0	0	0	275	558	19	852
comp14	0	0	0	0	0	225	504	25	754
comp15	0	0	0	0	0	215	450	16	681
comp16	2	0	0	0	0	360	656	28	1046
comp17	1	0	0	0	0	320	602	24	947
comp18	0	0	0	0	0	315	128	1	444
comp19	2	0	0	0	0	180	430	29	641
comp20	1	0	0	0	0	345	620	23	989
comp21	3	0	0	0	0	330	562	18	913

Table A.39: Summary execution results for Heur-1 4/9.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	0	0	0	0	106	205	26	2	339
comp02	17	0	0	0	0	270	596	17	900
comp03	7	0	0	0	0	245	432	12	696
comp04	2	0	0	0	0	280	448	16	746
comp05	6	0	0	0	235	215	1096	3	1555
comp06	10	0	0	0	0	330	716	23	1079
comp07	17	0	0	0	0	380	642	22	1061
comp08	4	0	0	0	0	230	536	16	786
comp09	10	0	0	0	0	245	614	15	884
comp10	1	0	0	0	0	295	522	23	841
comp11	0	0	0	0	0	135	96	1	232
comp12	6	0	0	0	0	260	1156	18	1440
comp13	8	0	0	0	0	315	490	10	823
comp14	3	0	0	0	0	245	508	23	779
comp15	7	0	0	0	0	245	432	12	696
comp16	9	0	0	0	0	355	582	18	964
comp17	8	0	0	0	0	400	542	22	972
comp18	0	0	0	0	0	310	152	4	466
comp19	4	0	0	0	0	265	428	20	717
comp20	10	0	0	0	0	470	606	20	1106
comp21	9	0	0	0	0	370	500	23	902

Table A.40: Summary execution results for Heur-1 4/10.

Test Data	H1	H2	H3	H4	S1	S2	S3	S4	Total Penalty
comp01	1	0	0	0	71	230	40	3	345
comp02	6	0	0	0	0	260	550	18	834
comp03	2	0	0	0	0	230	374	15	621
comp04	0	0	0	0	0	285	468	14	767
comp05	7	0	0	0	0	205	1094	7	1313
comp06	3	0	0	0	0	315	636	20	974
comp07	5	0	0	0	0	390	576	33	1004
comp08	2	0	0	0	0	245	536	17	800
comp09	6	0	0	0	0	255	554	11	826
comp10	6	0	0	0	0	320	564	26	916
comp11	0	0	0	0	0	145	84	1	230
comp12	16	0	0	0	4	330	1020	14	1384
comp13	3	0	0	0	0	280	540	14	837
comp14	4	0	0	0	0	270	436	16	726
comp15	2	0	0	0	0	230	374	15	621
comp16	3	0	0	0	0	365	688	23	1079
comp17	2	0	0	0	0	325	626	24	977
comp18	0	0	0	0	0	325	134	2	461
comp19	10	0	0	0	0	200	460	21	691
comp20	6	0	0	0	100	425	686	20	1237
comp21	10	0	0	0	0	355	516	13	894

APPENDIX B

Output files for the solutions of comp01, comp12, and comp18 with Heur-1: 1/5, Heur-1: 1/1, and Heur-2: 4/6 algorithms respectively

The output file of Heur-1: 1/5 algorithm for comp01 problem. (c0001 B 3 0 states that a lecture of c0001 takes place on Thursday (i.e.,3) in the first period (i.e., 0) in room B.)

c0001 B 0 0 c0015 C 0 0 c0070 E 0 0 c0033 F 0 0 c0063 G 0 0 c0062 S 0 0 c0001 B 0 1 c0016 C 0 1 c0064 E 0 1 c0033 F 0 1 c0061 G 0 1 c0068 S 0 1 c0001 B 0 2 c0016 C 0 2 c0057 E 0 2 c0033 F 0 2 c0061 G 0 2 c0068 S 0 2 c0001 B 0 3 c0016 C 0 3 c0057 E 0 3 c0033 F 0 3 c0061 G 0 3 c0068 S 0 3 c0001 B 0 4 c0016 C 0 4 c0057 E 0 4 c0033 F 0 4 c0061 G 0 4 c0068 S 0 4 c0001 B 0 5 c0016 C 0 5 c0057 E 0 5 c0033 F 0 5 c0061 G 0 5 c0068 S 0 5 c0004 B 1 0 c0016 C 1 0 c0057 E 1 0 c0030 F 1 0 c0061 G 1 0 c0068 S 1 0 c0004 B 1 1 c0016 C 1 1 c0058 E 1 1 c0030 F 1 1 c0069 G 1 1 c0072 S 1 1 c0004 B 1 2 c0017 C 1 2 c0058 E 1 2 c0030 F 1 2 c0064 G 1 2 c0072 S 1 2 c0004 B 1 3 c0017 C 1 3 c0066 F 1 3 c0058 G 1 3 c0059 S 1 3 c0004 B 1 4 c0025 C 1 4 c0058 E 1 4 c0066 F 1 4 c0069 G 1 4 c0059 S 1 4 c0004 B 1 5 c0025 C 1 5 c0066 F 1 5 c0069 G 1 5 c0059 S 1 5 c0004 B 2 0 c0025 C 2 0 c0066 F 2 0 c0069 G 2 0 c0059 S 2 0 c0002 B 2 1 c0025 C 2 1 c0066 F 2 1 c0069 G 2 1 c0059 S 2 1 c0002 B 2 2 c0024 C 2 2 c0031 F 2 2 c0064 G 2 2 c0072 S 2 2 c0002 B 2 3 c0024 C 2 3 c0031 F 2 3 c0064 G 2 3 c0072 S 2 3 c0002 B 2 4 c0024 C 2 4 c0031 F 2 4 c0064 G 2 4 c0072 S 2 4 c0002 B 2 5 c0024 C 2 5 c0031 F 2 5 c0064 G 2 5 c0072 S 2 5 c0002 B 3 0 c0078 C 3 0 c0066 F 3 0 c0058 G 3 0 c0059 S 3 0 c0005 B 3 1 c0078 C 3 1 c0031 F 3 1 c0065 G 3 1 c0063 S 3 1 c0005 B 3 2 c0078 C 3 2 c0062 F 3 2 c0065 G 3 2 c0063 S 3 2 c0005 B 3 3 c0078 C 3 3 c0071 F 3 3 c0065 G 3 3 c0067 S 3 3 c0014 B 3 4 c0078 C 3 4 c0071 F 3 4 c0065 G 3 4 c0067 S 3 4 c0015 B 3 5 c0032 C 3 5 c0071 F 3 5 c0065 G 3 5 c0067 S 3 5 c0015 B 4 0 c0025 C 4 0 c0062 F 4 0 c0065 G 4 0 c0063 S 4 0 c0015 B 4 1 c0025 C 4 1 c0062 F 4 1 c0070 G 4 1 c0063 S 4 1 c0015 B 4 2 c0025 C 4 2 c0062 F 4 2 c0070

G 4 2 c0063 S 4 2 c0015 B 4 3 c0025 C 4 3 c0071 F 4 3 c0070 G 4 3 c0067 S 4 3 c0015 B 4 4 c0030 C 4 4 c0071 F 4 4 c0070 G 4 4 c0067 S 4 4 c0015 B 4 5 c0030 C 4 5 c0071 F 4 5 c0070 G 4 5 c0069 S 4 5

The output file of Heur-1: 1/1 algorithm for comp012 problem.

LinLatB 10 0 0 TedUma 15 0 0 AntCul H 0 0 LogFilMat L 0 0 SocArt M 0 0 MetRicStoArt DTM 0 0 StoLibStaCSM2 HTM 0 0 LinLatB 10 0 1 ProMedCS 15 0 1 AntCul H 0 1 PalLat L 0 1 SocArt M 0 1 Codico DTM 0 1 StoTecArtCS HTM 0 1 LinLatB 10 0 2 ProMedCS 15 0 2 Sociol1 H 0 2 LetCriAnt L 0 2 MetRicStoArt M 0 2 LinLetGre1 DTM 0 2 StoLibStaCSM2 HTM 0 2 ArcStoArtMus 10 0 3 CodicoCS 15 0 3 Sociol1 H 0 3 MatCarCosResEdiSto L 0 3 TopRilMonAnt M 0 3 Paleoa DTM 0 3 StoLibSta HTM 0 3 Esteti 10 0 4 StoBib 15 0 4 Sociol1 H 0 4 StoCriArt L 0 4 TopRilMonAnt M 0 4 StoFilCS DTM 0 4 PapiroCS HTM 0 4 StoArt 10 0 5 LinLetLatCS 15 0 5 Geo1 H 0 5 MatCarCos-ResEdiSto L 0 5 TopRilMonAnt M 0 5 StoAntStaItaCS DTM 0 5 StoDisGra HTM 0 5 StoArt 10 1 0 Geo1 H 1 0 MatCarCosResEdiSto L 1 0 LinLetLat2 M 1 0 StoAntStaItaCS DTM 1 0 PapiroCS HTM 1 0 StoArt 10 1 1 CodicoCS 15 1 1 Geo1 H 1 1 StoCriArt L 1 1 LinLetLat2 M 1 1 StoFilCS DTM 1 1 LinLetGre1 HTM 1 1 ItaScr 10 1 2 StoBib 15 1 2 StoColIstMus H 1 2 StoTecArtM12 L 1 2 LinLetLat2 M 1 2 StoConCS DTM 1 2 StoLibStaCSM1 HTM 1 2 ItaScr 10 1 3 StoBib 15 1 3 StoColIstMus H 1 3 StoTecArtM12 L 1 3 EtrAntIta M 1 3 EpiLat DTM 1 3 StoLibStaCSM1 HTM 1 3 StoRom 10 1 4 LinLetLatCS 15 1 4 StoTecFot H 1 4 StoRes L 1 4 Paleoa M 1 4 StoModCS DTM 1 4 Codico HTM 1 4 StoRom 10 1 5 ArcMedCS 15 1 5 StoTecFot H 1 5 FonSoc L 1 5 Paleoa M 1 5 StoDisGra DTM 1 5 Codico HTM 1 5 Esteti 10 2 0 ChiResCS 15 2 0 InfDoc H 2 0 StoTecArtM12 L 2 0 StoFriCS M 2 0 StoTecArtM3 DTM 2 0 BioarcCS HTM 2 0 StoArcCon 10 2 1 StoMed2 H 2 1 PalLat L 2 1 StoGre M 2 1 StoArc DTM 2 1 BioarcCS HTM 2 1 StoArcCon 10 2 2 StoMed2 H 2 2 PalLat L 2 2 EtrAntIta M 2 2 EleInfSciCatBenCulM3 DTM 2 2 StoArtCon2CS HTM 2 2 Esteti 10 2 3 ArcVicOriAnt 15 2 3 InfDoc H 2 3 StoProArtCulMatMed L 2 3 LinIta M 2 3 EleInfSciCatBenCulM3 DTM 2 3 StoFilMedCS HTM 2 3 ArcStoArtMus 10 2 4 OrientCS 15 2 4 InfDoc H 2 4 LogFilMat L 2 4 LinIta M 2 4 Glotto DTM 2 4 StoArtMod2CS HTM 2 4 LinGen 10 2 5 StoMed2 H 2 5 StoProArtCulMatMed L 2 5 EtrAntIta M 2 5 StoFilMedCS DTM 2 5 StoArtMod2CS HTM 2 5 LinGen 10 3 0 ArcMedCS 15 3 0 StoMod2 H 3 0 Archiv L 3 0 SocArt M 3 0 StoTecArtM3 DTM 3 0 StoArtCon2CS HTM 3 0 LinGen 10 3 1 StoMod2 H 3 1 Archiv L 3 1 ArcCriMed M 3 1 StoRomCS DTM 3 1 LabScr HTM 3 1 LinLatA 10 3 2 StoMod2 H 3 2 Archiv L 3 2 EleInfSciCatBenCulM3 M 3 2 StoRomCS DTM 3 2 Cartog HTM 3 2 LinLatA 10 3 3 OrientCS 15 3 3 LegBenCul2 H 3 3 StoArcMed L 3 3 StoLinItaCS M 3 3 StoArc DTM 3 3 ArcVicOriAnt HTM 3 3 StoCriCin 10 3 4 NumismCS 15 3 4 LegBenCul2 H 3 4 FilTeo L

3 4 Glotto M 3 4 StoArc DTM 3 4 Cartog HTM 3 4 StoCriCin 10 3 5 ArcClaCS 15 3 5 PsiTur H 3 5 FilTeo L 3 5 ArcCriMed M 3 5 FilCla2 DTM 3 5 ArchivCS HTM 3 5 StoMus 10 4 0 ArcClaCS 15 4 0 GeoTur H 4 0 ArcCla1 L 4 0 StoLinItaCS M 4 0 FilCla2 DTM 4 0 ChiResCS HTM 4 0 LetCriAnt 10 4 1 PsiTur H 4 1 StoArcMed L 4 1 ArcCla2 M 4 1 LinLetGre1 DTM 4 1 Cartog HTM 4 1 LetCriAnt 10 4 2 PsiTur H 4 2 LinIta L 4 2 ArcCla2 M 4 2 StoModCS DTM 4 2 ProMedCS HTM 4 2 StoMus 10 4 3 NumismCS 15 4 3 GeoTur H 4 3 FilTeo L 4 3 EleInfSciCatBenCulM2 M 4 3 FilCla2 DTM 4 3 ArchivCS HTM 4 3 StoRes 10 4 4 GeoTur H 4 4 ArcCla1 L 4 4 EpiLat M 4 4 StoConCS DTM 4 4 ArcVicOriAnt HTM 4 4 StoRes 10 4 5 EcoAzi H 4 5 ArcCla1 L 4 5 EpiLat M 4 5 MatTecAppArcM3 DTM 4 5 ArchivCS HTM 4 5 FonSoc 10 5 0 EcoAzi H 5 0 LinGreA L 5 0 EleInfSciCatBenCulM2 M 5 0 MatTecAppArcM3 DTM 5 0 TedUma HTM 5 0 FonSoc 10 5 1 EcoAzi H 5 1 LinGreA L 5 1 Glotto M 5 1 StoTecArtCS DTM 5 1 TedUma HTM 5 1

The output file of Heur-2: 4/6 algorithm for comp018 problem.

LET-GEO-Geo1 r1 0 0 LET-ITA-LetIta2 r10 0 0 LET-STO-StoMed2 rC1 0 0 CBC-ART-StoArcMed rC2 0 0 CBC-LIB-InfDoc rL 0 0 CBC-ART-StoArcCon rM 0 0 CBC-ARC-TopRilMonAnt rN 0 0 CBC-LIB-ArcSpe rO 0 0 CBC-ART-StoRes rVL 0 0 LET-STO-StoMed1 r1 0 1 CBC-ARC-Palantro r10 0 1 CBC-ARC-EtrAntIta rC1 0 1 LET-CST-EpiLat rC2 0 1 CBC-ART-StoTecArt rL 0 1 LET-ITA-LinIta rM 0 1 LET-CLE-LinGre rN 0 1 CBC-ART-SocArt rO 0 1 CBC-ART-EleInfSciCatBenCul rVL 0 1 CBC-LIB-StoBib r1 0 2 LET-CLE-LinLat r10 0 2 CBC-ARC-ArcCriMed rC1 0 2 CBC-ART-StoTecFot rC2 0 2 CBC-ARC-LinLat rL 0 2 CBC-LIB-PalLat rM 0 2 LET-CLE-Glo rN 0 2 LET-GEO-Geo2 rO 0 2 CBC-ART-StoCriCin rVL 0 2 CBC-ART-StoDisGra r1 0 3 LET-ITA-EleLinGenFil r10 0 3 CBC-ART-StoProArtCulMatMed rC1 0 3 LET-CLE-FilCla rC2 0 3 CBC-LIB-Cod rL 0 3 CBC-ART-MatCarCosResEdiSto rM 0 3 CBC-ARC-MatTecAppArc rN 0 3 CBC-LIB-Arc rO 0 3 CBC-ART-ArcStoArtMusA rVL 0 3 LET-CLE-LinLetGre2 rC2 0 4 CBC-ARC-Geo1 rM 0 4 CBC-LIB-StoLibSta rN 0 4 CBC-ARC-ArcCla2 rVL 0 4 CBC-LIB-StoBib r1 1 0 LET-ITA-EleLinGenFil r10 1 0 LET-GEO-Geo2 rC1 1 0 LET-CLE-FilCla rC2 1 0 CBC-LIB-Cod rL 1 0 CBC-ART-MatCarCosResEdiSto rM 1 0 CBC-ARC-MatTecAppArc rN 1 0 CBC-LIB-Arc rO 1 0 CBC-ART-StoArcMed rVL 1 0 LET-STO-StoMed1 r1 1 1 CBC-ARC-EtrAntIta r10 1 1 CBC-ART-StoRes rC1 1 1 LET-CLE-LinLetGre2 rC2 1 1 CBC-ART-StoArcCon rL 1 1 LET-ITA-LinIta rM 1 1 LET-CLE-LinGre rN 1 1 CBC-LIB-ArcSpe rO 1 1 LET-CST-EpiLat rVL 1 1 LET-GEO-Geo1 r1 1 2 LET-ITA-LetIta2 r10 1 2 CBC-ART-StoTecArt rC1 1 2 CBC-ART-EleInfSciCatBenCul rC2 1 2 CBC-ARC-LinLat rL 1 2 CBC-LIB-InfDoc rM 1 2 CBC-ARC-TopRilMonAnt rN 1 2 CBC-ART-SocArt rO 1 2 LET-STO-StoMed2 rVL 1 2 CBC-

ARC-ArcCla2 r1 1 3 LET-CLE-LinLat r10 1 3 CBC-ARC-ArcCriMed rC1 1 3 CBC-ART-StoTecFot rC2 1 3 CBC-ARC-Geo1 rL 1 3 CBC-LIB-PalLat rM 1 3 LET-CLE-Glo rN 1 3 CBC-ART-StoDisGra rO 1 3 CBC-ART-StoCriCin rVL 1 3 CBC-ARC-Palantro rC1 1 4 CBC-ART-ArcStoArtMusA rC2 1 4 CBC-LIB-StoLibSta rN 1 4 CBC-ART-StoProArtCulMatMed rVL 1 4 CBC-LIB-StoBib r1 2 0 LET-ITA-EleLinGenFil r10 2 0 LET-GEO-Geo2 rC1 2 0 LET-CLE-FilCla rC2 2 0 CBC-LIB-Cod rL 2 0 CBC-ART-MatCarCosResEdiSto rM 2 0 CBC-ARC-MatTecAppArc rN 2 0 CBC-LIB-Arc rO 2 0 CBC-ART-StoArcMed rVL 2 0 LET-STO-StoMed1 r1 2 1 LET-STO-StoMed2 r10 2 1 CBC-ARC-EtrAntIta rC1 2 1 LET-CLE-LinLetGre2 rC2 2 1 CBC-ART-StoArcCon rL 2 1 LET-ITA-LinIta rM 2 1 CBC-ARC-TopRilMonAnt rN 2 1 CBC-LIB-ArcSpe rO 2 1 CBC-ART-StoRes rVL 2 1 LET-GEO-Geo1 r1 2 2 LET-ITA-LetIta2 r10 2 2 CBC-ARC-ArcCriMed rC1 2 2 LET-CST-EpiLat rC2 2 2 CBC-ARC-Geo1 rL 2 2 CBC-LIB-InfDoc rM 2 2 CBC-ART-StoTecArt rN 2 2 CBC-ART-SocArt rO 2 2 CBC-ART-EleInfSciCatBenCul rVL 2 2 CBC-LIB-PalLat r10 2 3 CBC-ART-StoProArtCulMatMed rC1 2 3 CBC-ART-StoCriArt rC2 2 3 LET-FIL-Est rL 2 3 LET-CST-StoRom rM 2 3 LET-CLE-Glo rN 2 3 CBC-ARC-Palantro rO 2 3 LET-GEO-StoCriCin rVL 2 3 CBC-ARC-ArcCla2 r10 2 4 CBC-ART-ArcStoArtMusA rC1 2 4 CBC-ART-StoTecFot rC2 2 4 CBC-LIB-StoLibSta rL 2 4 LET-CST-StoGre rM 2 4 LET-FIL-StoSci rN 2 4 CBC-ART-StoDisGra rO 2 4 CBC-ART-StoCriCin rVL 2 4 CBC-ART-StoMus rN 2 5 CBC-ART-StoCriArt rC2 3 0 LET-FIL-Est rL 3 0 LET-CST-StoRom rM 3 0 LET-GEO-StoCriCin rVL 3 0 LET-CST-StoGre rM 3 1 LET-FIL-StoSci rN 3 1 CBC-ART-StoMus rN 3 2 CBC-ART-StoCriArt rC2 4 0 LET-FIL-Est rL 4 0 LET-CST-StoRom rM 4 0 LET-GEO-StoCriCin rVL 4 0 LET-CST-StoGre rM 4 1 LET-FIL-StoSci rN 4 1 CBC-ART-StoMus rN 4 2