A HOLISTIC APPROACH FOR WORKFORCE SCHEDULING AND ROUTING PROBLEM

KEREM CAN MANALP

Thesis for the Master's Program in Computer Engineering

Graduate School

Izmir University of Economics

Izmir

2024

A HOLISTIC APPROACH FOR WORKFORCE SCHEDULING AND ROUTING PROBLEM

KEREM CAN MANALP

THESIS ADVISOR: ASST. PROF. DR. KUTLUHAN EROL

Master's Exam Jury Members
Assoc. Prof. Dr. Kaya OĞUZ
Asst. Prof. Dr. Kutluhan EROL
Asst. Prof. Dr. Başak Esin KÖKTÜRK GÜZEL

A Master's Thesis
Submitted to
the Graduate School of Izmir University of Economics
the Department of Computer Engineering

Izmir

2024

ETHICAL DECLARATION

I hereby declare that I am the sole author of this thesis and that I have conducted my work in accordance with academic rules and ethical behaviour at every stage from the planning of the thesis to its defence. I confirm that I have cited all ideas, information and findings that are not specific to my study, as required by the code of ethical behaviour, and that all statements not cited are my own.

Name, Surname: KEREM CAN MANALP

Date:09.07.2023

Signature:

ABSTRACT

A HOLISTIC APPROACH FOR WORKFORCE SCHEDULING AND ROUTING PROBLEM

Manalp, Kerem Can

Master's Program in Computer Engineering

Advisor: Asst. Prof. Dr. Kutluhan Erol

June, 2024

This thesis investigates the Workforce Scheduling and Routing Problem (WSRP) in a business-centric manner, focusing on the development and application of a mathematical model designed to optimise task assignments for field service workers. We proposed an optimisation model that utilises a maximisation function centred around a scoring system that integrates customer satisfaction factors like Service Level Agreement (SLA) times and priorities. We adapted datasets from Solomon (1987) and Homberger and Gehring (1999) and tested our model, conducting a total of 204 experiments within the specified time limits for each algorithm implemented. Three methods were adapted and explored for the proposed model: a Mixed Integer Programming (MIP) approach using the Gurobi Solver, Tabu Search, and a Genetic Algorithm. The effectiveness of these methods was evaluated based on their runtime performance, solution quality and customer satisfaction indicators. The findings showed that although the solution qualities of all three methods are robust, there are significant differences in larger problem sizes. Additionally, results demonstrated our model's capability in assigning tasks to employees, considering both customer

satisfaction factors and operational efficiencies. This research highlights the critical role of advanced models in field service operations, enhancing service delivery quality and operational efficiency. With the insights obtained from our experiments, this study not only contributes to proposing a more business-centric model but also the ongoing efforts to improve resource allocation strategies in service industries.

Keywords: WSRP, business-centric optimization model, scheduling, task assignment, routing

ÖZET

İŞGÜCÜ PLANLAMA VE YÖNLENDİRME PROBLEMİ İÇİN BÜTÜNSEL BİR YAKLAŞIM

Manalp, Kerem Can

Bilgisayar Mühendisliği Yüksek Lisans Programı

Tez Danışmanı: Asst. Prof. Dr. Kutluhan Erol

Haziran, 2024

Bu tez, İşgücü Planlama ve Yönlendirme Problemi'ni (WSRP) iş odaklı bir yaklaşımla incelemekte olup, saha hizmet çalışanları için görev atama optimizasyonu amacıyla matematiksel bir modelin geliştirilmesi ve uygulanmasına odaklanmaktadır. Müşteri memnuniyeti faktörlerini, Servis Seviye Anlaşması (SLA) süreleri ve iş öncelikleri gibi unsurları entegre eden bir puanlama sistemi merkezli maksimizasyon fonksiyonu kullanan bir optimizasyon modeli önerdik. Solomon (1987) ve Homberger ve Gehring (1999) tarafından sağlanan veri setlerini uyarlayarak modelimizi test ettik ve her bir algoritma ile, belirlenen süre sınırları içinde toplam 204 deney gerçekleştirdik. Önerilen modele üç adet yöntem uyarlandı ve incelendi: Karışık Tamsayılı Programlama (MIP) yaklaşımı, Tabu Arama ve Genetik Algoritma. Bu yöntemler etkinliği, çalışma süreleri, çözüm kaliteleri ve müşteri memnuniyeti göstergeleri açısından değerlendirildi. Bulgular, tüm üç yöntemin de yüksek çözüm kalitesi sunduğunu göstermekle birlikte, daha büyük problem boyutlarında bazı önemli farklılıklar gözlemlendi. Ayrıca, sonuçlar modelimizin, müşteri memnuniyeti faktörlerini ve operasyonel verimlilikleri göz önünde bulundurarak çalışanlara görev

atama yeteneğini ortaya koydu. Bu araştırma, hizmet kalitesini ve operasyonel verimliliği arttırarak saha hizmet operasyonlarında ileri modellerin kritik rolünü vurgulamaktadır. Deneylerimizden elde edilen içgörülerle bu çalışma, sadece daha iş odaklı bir model önermekle kalmayıp, hizmet endüstrilerinde kaynak planlama stratejilerini geliştirme çabalarına da katkıda bulunmaktadır.

Keywords: WSRP, business-centric optimization model, scheduling, task assignment, routing

ACKNOWLEDGEMENTS

I would like to express my deepest thanks to my supervisors, Asst. Prof. Kutluhan EROL and Prof. Dr. Cem EVRENDİLEK for their support and guidance.

I would also like to thank my wife Seden MANALP, for supporting me in every way throughout the whole process

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ALNS : Adaptive Large Neighbourhood Search

FTSP : Field Technician Scheduling Problem

GA : Genetic Algorithms

GRASP : Greedy Randomised Adaptive Search Procedure

HCCSP : Home Care Crew Scheduling Problem

HHC : Home Health Care

ILP : Integer Linear Programming

ILS : Iterated Local Search

IP : Integer Programming

LP : Linear Programming

MIP : Mixed Integer Programming

OOM : Out Of Memory

RCL : Restricted Candidate List

SLA : Service Level Agreement

STRSP : Service Technician Routing and Scheduling

TS : Tabu Search

TSP : Travelling Salesperson Problem

TTSP : Technician and Task Scheduling Problem

VRP : Vehicle Routing Problem

VRPTW : Vehicle Routing Problem with Time Windows

WSRP : Workforce Scheduling and Routing Problem

CHAPTER 1: INTRODUCTION

In today's dynamic workforce landscape, the effective management of field service operations is essential to the success of organisations. As industries increasingly rely on field services to meet customer demands, optimising workforce scheduling and routing becomes a critical challenge. The complexities associated with managing an expanding pool of tasks, diverse skill sets among employees, stringent service level agreements (SLAs) coupled with the importance of customer satisfaction, emphasise the need for a sophisticated scheduling system.

The term Workforce Scheduling and Routing Problem (WSRP), coined by Castillo-Salazar, Landa-Silva, and Qu (2012), generalises the challenge of effectively assigning personnel to perform tasks at different locations while considering both worker and customer needs. For example, technicians carrying out maintenance jobs, nurses visiting patients, and carers tending to citizens at their homes, all these work scenarios can be considered within the scope of WSRPs. The WSRP can also be seen as an extension of the well-known vehicle routing problem with time windows (VRPTW), with the relaxation of the constraint that all tasks or destinations be visited (Xie, Potts, and Bektaş, 2017).

The WSRP encapsulates the multi-aspected nature of task assignment such as time windows, task priority, travelling time, skill requirements, service duration and teaming, and it introduces an optimal assignment plan. Manual scheduling becomes increasingly complex as the number of tasks and employees grows. This complexity makes automated solutions essential, not only for improving operational efficiency but also for maximising workforce potential and providing customer satisfaction.

This thesis explores the complexities of the WSRP, examining algorithms that address the challenges of workforce scheduling. We introduce an innovative optimisation model that transforms traditional multi-objective functions by integrating a customisable score generation formula with an adjustable objective function. By simplifying the objective function while maintaining its adaptability, this approach allows for the maximisation of operational efficiency by prioritising tasks not only based on their intrinsic attributes but also on logistical and strategic considerations.

The aim is to enhance our understanding of task assignment challenges and improve the efficiency of workforce operations, especially in field services.

This research offers a fresh perspective on managing field service operations by reformulating the problem as a maximisation problem centred around a singular, meticulously crafted score. It addresses the need for a system that not only meets the technical and operational demands of each task but also leverages service quality, considering customer satisfaction. The result is a model that promises enhanced efficiency, satisfaction, and adaptability in the face of evolving service demands of industries.

We will present a mathematical formulation of the WSRP, introduce our model and adapt multiple solution procedures, including mixed integer programming, a genetic algorithm, and a tabu search algorithm. The model we present, using our score generation function, captures the non-linear aspects of customer satisfaction as pre-computed coefficients that are integrated into a linear model for faster optimisation. This approach not only reduces complexity but also enhances performance when solving the problem. Additionally, it offers flexibility to incorporate new objectives into the existing objective function without the need for structural changes.

This thesis is organised as follows: Section 2 provides the necessary background information and related works to our problem and discusses their relations, differences, and contributions. Sections 3 and 4 describe the problem and introduce model formulation, constraints and solution approaches employed. Section 5 provides the experimental setup and datasets adapted. Section 6 presents and discusses the results obtained. Finally, section 7 concludes the thesis and provides future works.

CHAPTER 2: BACKGROUND

2.1. Exact Approaches

Exact solution methods guarantee finding the best possible solution while strictly adhering to all problem constraints. These methods search the entire solution space within the scope of the defined objective. Their application is limited in the context of NP-hard problems, due to the exponential growth of computational work needed with the size of the problem. The increase in the number of variables and constraints that cause the combinatorial explosion is the "curse of dimensionality", as Bellman (1966) refers to, which perfectly fits the concept of exact solution methods.

The WSRP can be considered as a combination of the personnel scheduling problem (Brucker, Qu, and Burke, 2011) and the vehicle routing problem with time windows (Lenstra and Rinnooy Kan, 1981), which are generalisations of the Traveling salesperson problem, one of the archetypes of *NP-complete* complexity class (Papadimitriou, 1977). Just like these problems, WSRP also deals with a huge number of possible solutions that grow exponentially as we add more tasks, employees and routing considerations. As the problem gets bigger, the computational work needed to solve the problem becomes more challenging, rendering exact methods less viable for obtaining solutions within a reasonable timeframe.

Some of the exact approaches are dynamic programming, network flow algorithms, and integer linear programming-based methods such as branch-and-bound, branch-and-cut, and branch-and-price (Burke and Kendall, 2014). In section 4 we present a mixed integer programming model for our WSRP problem.

2.1.1. Linear Programming and Mixed Integer Programming

Linear programming problems (LP) seek to find the optimum values of real-valued variables in the presence of linear equality and/or inequality constraints that maximise or minimise a linear objective function. When all or some of the variables are restricted to have integer values, the resulting problems are called pure integer programming (pure IP), and mixed integer programming (MIP), respectively. More formally, an MIP

is defined as given a set of coefficients c_j, d_k, a_{ij}, g_{ik}, b_i, find optimal values for variables x_j and y_k that maximize z, specified below:

$$(MIP)\ \text{Maximize} \quad z = \sum_j c_j\, x_j + \sum_k d_k\, y_k$$

$$\text{subject to} \quad \sum_j a_{ij}\, x_j + \sum_k g_{ik}\, y_k \le b_i \quad (i = 1,2,\dots,m)$$

$$x_j \ge 0 \quad (j = 1,2,\dots,n)$$

$$y_k \in \{0,1,2,\dots\} \quad (k = 1,2,\dots,p)$$

Note that LP has polynomial-time solutions, whereas IP and ILP are in the NP-complete complexity class, for which the only known algorithms are in exponential time. Some of the exact solution techniques for solving MIPs are branch-and-bound, branch-and-cut, branch-and-price and cutting plane methods (Burke and Kendall, 2014).

2.2. Approximation Approaches

In most real-life scenarios where optimisation challenges are complex and the size of the problem is large enough, exact methods may not be practical due to the excessive computational time requirements. While exact methods solve the problem optimally and guarantee to find the best result, approximation methods aim to find high-quality solutions more quickly without an optimality guarantee. This trade-off is the essence of approximation methods, and maintaining the balance is the key to the success of these approaches.

2.2.1. Heuristic Search Methods

The fundamental challenge of heuristic search is developing a strategy that provides good enough solutions in a reasonable amount of time. These search methods explore the solution space and aim to guide the search towards better results in every iteration, using intuition and/or the knowledge/experience gained from previous solutions. While greedy construction algorithms, neighbourhood search, relaxation techniques,

and decomposition are some common heuristic search methods, there are other widely used techniques known as ‘metaheuristics’ such as tabu search (TS), genetic algorithms (GA), greedy randomised adaptive search (GRASP) and simulated annealing (SA).

2.2.1.2. Tabu Search

Tabu search (TS) is a meta-heuristic search method for solving any optimisation problems that was first coined by Glover (1986) in the same paper where the meta-heuristic term was introduced. The aim is to find the best decisions or actions (moves) to find the optimal result while taking advantage of adaptive memory. The adaptive memory helps guide the search through more promising regions by exploiting the information collected during the search process. (Glover, Laguna and Marti 2007)

A simple local search algorithm explores the solution space by iteratively making moves from one solution to a neighbouring solution until no improvement occurs. However, by consistently selecting the best neighbouring solution, we risk becoming trapped at a local optimum. Although at first glance, the tabu search algorithm may resemble a simple local search algorithm, TS addresses the limitation of the local search methods by employing adaptive memory. We can classify adaptive memory as recency-based and frequency-based memory. In other words, information collected via recency-based and frequency-based memory allows us to modify the neighbourhood strategically instead of exploring the immediate neighbourhood. This helps us both escape local optima and potentially discover better solutions that a basic local search approach would overlook.

The pseudo-code, which illustrates the general structure of the TS algorithm presented in Figure 1

Algorithm 1: Tabu Search

```
1: s = s0  // Initialize with an initial solution
2: Initialize tabuList with a maximum length of T
3: bestResult = Evaluate(s) // Evaluate the initially generated solution
4: while the stopping condition is not met do
5:   modifiedNeighbourhood = GenerateNeighbourhood(s, tabuList) // Generate neighbors excluding those in the tabu list
6:   bestNeighbour = SelectBestNeighbour(modifiedNeighbourhood) // Find the best solution in the modified neighbourhood
7:   bestNeighbourResult = Evaluate(bestNeighbour)
8:   s = bestNeighbour // Move to the best neighbouring solution
9:   if bestNeighbourResult is better than bestResult
10:     bestResult = bestNeighbourResult // Update the best known result
11:  UpdateTabuList(tabuList, bestNeighbourResult) // add current solution to the tabu list
12: end if
13: end while
14: return bestResult // Return the best result found
```

Figure 1. Pseudo code of tabu search algorithm

Recency-based memory in tabu search is managed with a tabu list, which records moves to avoid a predetermined period. This period is determined by the list's length, referred to as the tabu tenure. To optimise memory usage, short-term memory specifically logs the moves leading to new solutions rather than recording the entire solution. By restricting recent moves, we not only prevent cycles and revisit the same solutions but also maintain diversity.

Frequency-based memory tracks the information acquired throughout the search regarding the number of iterations or ratios of an attribute (move) that belongs to a good solution or how often attributes change. In other words, we can represent frequency-based memory as refining the search strategy by the frequency of the presence or absence of moves in a solution historically.

The fundamental aspect of Tabu Search's adaptive memory strategy is to maintain a balance between concentrating the search within promising areas (intensification) and broadening the search to new areas of the solution space to avoid local optima (diversification).

One additional concept in TS is the aspiration criteria. This feature makes exceptions to the tabu rules under certain conditions, such as a predefined quality threshold. Aspiration criteria override the tabu list, allowing a move that is otherwise tabu. The most common use of aspiration criteria is when a move that is tabu improves the best result acquired.

2.2.1.3. Genetic Algorithms

Genetic algorithms (GA) are one of many algorithms inspired by nature. They are based on the principles of evolution such as natural selection, mutation, and mating. The genetic algorithm is a population-based heuristic that dates back to the 1970s and was first proposed by Holland, J. (1975).

Solution space in GA is abstracted as a population of individuals, and each individual represents a solution to the problem to be solved. Further generations of populations produce iteratively through cross-over, mutation and selection processes in the concept of survival of the fittest.

In a typical genetic algorithm, the process begins by creating an initial group of solutions referred to as a population and each solution (individual) is represented by a chromosome. This representation can either be direct, where the chromosome straightforwardly maps to the solution, or indirect, requiring a decoding process to interpret the solution from the chromosome. The most common practice of chromosome representation is binary string representation (Katoch, Chauhan and Kumar, 2021) where each bit denotes a gene of the chromosome.

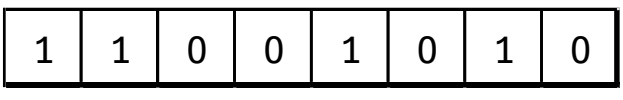

Figure 2. Chromosome

Once the initial population is generated, the fitness value, which can be calculated via an objective function in an optimisation problem, is assigned to all chromosomes. Chromosomes with higher fitness values are more likely to be selected for the subsequent cross-over process, where segments of two parent chromosomes are combined to create two distinct offspring, introducing new solutions to the population. After that, a mutation operation is applied to the population to maintain diversity and avoid premature convergence by randomly altering some genes.

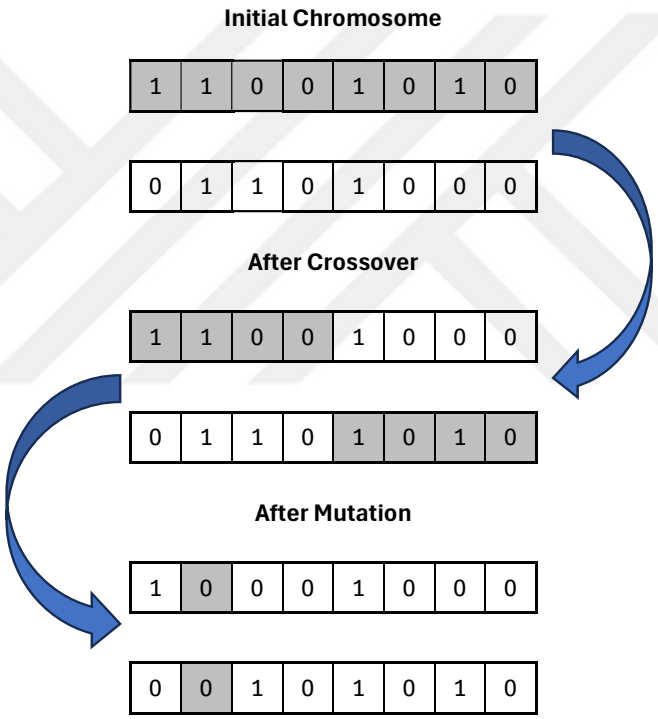

Figure 3. Genetic Operators

A specific selection mechanism that considers fitnesses, determines individuals that will survive and pass to the next generation, ensuring that the evolving population adapts towards better solutions. This cycle of selection, cross-over, mutation and evaluation is repeated with the population evolving over successive generations until

a specified termination criterion is met. A typical genetic algorithm is presented in Figure 4.

Algorithm 2: Genetic Algorithm

```
1:  population = InitializePopulation()
2:  CalculateFitnessValues(population) // Calculate and assign fitness of each individual in the population
3:  bestSolution = InitializeBestSolution() // Initialize variable to track the best solution found
4:  while the stopping condition is not met do
5:      parents = SelectParents(population) // Select parents for crossover based on fitness
6:      offspring = Crossover(parents) // Apply crossover to generate offspring
7:      population = Combine(population, offspring) // Combine parents and offspring to form a new population
8:      Mutate(population) // Apply mutation to the entire population
9:      CalculateFitnessValues(population) // Recalculate fitness for the entire population
10:     UpdateBestSolution(population, bestSolution) // Update best solution if found in the population
11:     population = SelectNewGeneration(population) // Select individuals for the next generation based on fitness
12: end while
13: return bestSolution // Return the global best solution found
```

Figure 4. Pseudo-code of genetic algorithms

2.3. Related Work

Field service operations are a crucial aspect across various service industries that demand efficient scheduling and routing to meet customer needs. Over decades, research in this area has progressed, leading to new terms that cover a broader range of topics and industries. Early research focused on Technician Routing and Scheduling Problems (Xu and Chiu, 2001) (Cordeau et al., 2010) (Kovacs et al, 2012), Home

Health Care (Cheng and Rich, 1998) (Bertels and Fahle, 2006) (Rasmussen et al, 2012) and Manpower Allocation (Lim, Rodrigues and Song, 2004) (Li, Lim and Rodrigues, 2005) (Dohn, Kolind and Clausen, 2007) to figure out how to manage tasks and people in different places. The term "Workforce Scheduling and Routing Problem (WSRP)" started being used in 2012 (Castillo-Salazar, Landa-Silva, and Qu, 2012), with a broader view that includes various service industries.

The Workforce Scheduling and Routing Problem (WSRP) represents a significant challenge due to the complex interplay between task assignments, employee skill sets, and customer satisfaction metrics. Despite the abundant research in this field, literature often presents it under various titles, leading to a diversity of terms for similar challenges. Although they appear under various names in the literature, WSRP and WSRP-like scenarios can fundamentally be considered as generalisations of the Vehicle Routing Problem with Time Windows (VRPTW), which is one of the most extensively researched topics in operations research. (Bodin, 1983) (Solomon and Desrosiers, 1988)

This thesis builds on previous studies, using WSRP to explore the complex needs of modern service operations.In this section, we review some of the problems tackled in the literature that can be considered as a variant of WSRP.

2.3.1. Technician Scheduling Problem

In this problem, the objective is to assign a set of tasks to a set of field technicians with different skills. The Technician Scheduling Problem has been examined under various names across the literature, illustrating its broad applicability and interpretation. For instance, Xu and Chiu, (2001) referred to it as the 'field technician scheduling problem,' Cordeau et al. (2010) termed it the 'technician and task scheduling problem,' and Kovacs et al. (2011) described it as the 'service technician routing and scheduling problem.' This also indicates its wide-ranging relevance and diverse interpretations in different service industries.

Xu and Chiu, (2001) describe field technician scheduling problem (FTSP) as a typical job assignment and scheduling problem with the following constraints. Jobs need certain skills and must be performed at customer locations within the specified time

windows by qualified technicians with the requisite skills. Jobs are spread across various locations, so travelling times from one job to another must be taken into account. Finally, jobs have priorities according to their importance or urgency. Xu and Chiu, (2001) formulated the problem as a maximisation problem, and the objective is to schedule as many jobs as possible within the time windows. The authors used heuristic methods for solving the problem, namely a greedy heuristic, a local search algorithm and a greedy randomised adaptive search procedure (GRASP). Results demonstrated that GRASP produced the best solutions for all scenarios and the local search algorithm is its close competitor and requires much less computer power.

Cordeau et al. (2010) introduced the Technician and Task Scheduling Problem (TTSP), which incorporates additional complexities such as outsourcing, team collaboration, dependent activities, and varying levels of skill proficiency. The model ensures that technicians, either individually or in groups, must have the necessary skills to perform a task. Moreover, any task with predecessors requires those preceding tasks to be completed in advance. Notably, this problem does not include travelling times and time window constraints, diverging from other technician routing and scheduling problems. A two-phase construction heuristic and an adaptive large neighbourhood search heuristic were proposed for the problem, aiming to minimise the latest task's ending time weighted by priorities.

Building on the work of Cordeau et al. (2010), Kovacs et al. (2012) presented a different approach in their study on Service Technician Routing and Scheduling (STRS). They reintroduced elements like time window constraints and travel times, which were not considered by Cordeau et al (2010). Their focus was on optimising the schedule to minimise travel time and outsourcing costs, which also takes into consideration the logistical aspects of scheduling, emphasising efficiency in both time management and cost reduction. The authors have modelled and solved the STRS both with and without teaming, where teaming refers to grouping employees to perform the same task, using adaptive large neighbourhood search

2.3.2. Home Health Care

The Home Health Care Problem (HHC) is similar at its core to the Technician Scheduling Problem, though it uses different terms. Here, the workers are nurses, and the main goal is to ensure high-quality care. Nurses, who have various skills, need to be matched with clients located in different places, each requiring specific care skills. The aim is to balance the nurses' workload and keep consistent care for clients, all while considering the preferences of both nurses and clients.

Cheng and Rich (1998) explored the Home Health Care Problem, where nurses were evaluated as either full-time or part-time nurses. Similar to the concept of outsourcing in technician scheduling, part-time nurses are compensated based on the hours they work. Nurses start their day from home and visit patients within a certain time window. The study also adds lunch breaks into the schedule. The objective is to minimize the amount of overtime and part-time hours. The authors proposed a combined mixed-integer programming (MIP) and a two-phase heuristic. In the first phase, feasible tours are constructed with a randomized greedy algorithm; in the second phase, heuristic attempts to improve the solution using a form of local search procedure.

By presenting some soft constraints like preferred time windows of patients, preferred nurses by patients, and preferred patients by nurses, Bertels and Fahle (2006) aim to incorporate the satisfaction of both nurses and patients. While the hard constraints, such as time window, qualification, and working shift hours, ensure feasibility and adherence to regulations, the authors aim to achieve satisfaction by penalizing soft constraints in the objective function. The author's objective is to assign all jobs to nurses while all hard constraints are respected, such that the overall cost for that assignment is minimal and the number of preferences satisfied is maximal. They proposed both single-paradigm and hybrid approaches and showed that the hybrid approaches, especially constraint programming and tabu search, produce better results than single-paradigm approaches.

Home Care Crew Scheduling Problem (HCCSP), as the term used by Rasmussen et al. (2012) introduced the concept of temporal dependency. This refers to scenarios where one caregiver’s action depends on another’s, such as when one caregiver starts a washing machine and another later empties it, defined as 'minimum difference and maximum difference.' Moreover, temporal dependencies must be at the specified time

windows which are predefined via five temporal dependency types. Another key aspect of the paper is visit clustering. To be able to solve larger instances, clustering is included in the problem as it divides the area into different sections which are defined as clusters and restricts the caregiver's visits to a clustered area. The authors presented a branch and price algorithm to minimize the uncovered visits, maximizing carer visit preferences and minimising total travelling costs and showed that using clusters with preferred visits significantly decreased computational run times.

2.3.3. Manpower Allocation Problem

The core challenge of the manpower allocation problem lies in strategically distributing human resources across various tasks, reflecting similar concerns as those in Technician Scheduling and Home Health Care. While the concept of teaming appears in many WSRP-related problems, it's important to differentiate it from the demand for a fixed number of servicemen at specific locations, which is a distinctive aspect of manpower allocation. In essence, whereas teaming involves grouping employees based on compatibility and skillsets for tasks, manpower allocation focuses on matching a predetermined number of employees to locations, highlighting a fundamental difference between the two approaches.

Lim, Rodrigues, and Song (2004) examined the real-life manpower allocation requirements at the Port of Singapore and proposed a manpower allocation model with time windows. Here the workers are referred to as servicemen and are dispatched from a centre in required numbers to various locations. The objectives are to minimize the number of servicemen used, distance cost and times spent on travelling and waiting. The authors developed a tabu-embedded simulated annealing algorithm and a squeaky wheel optimization with local search algorithm and got better results than those already available at the port.

Li, Lim and Rodrigues, (2005) studied the manpower allocation problem with time windows and job teaming constraints. Their research introduced a worker-type notation, marking a divergence from Lim, Rodrigues, and Song’s (2004) earlier work. This notation is crucial as it accounts for jobs that require multiple workers of different types, presenting a more complex approach to allocate manpower effectively. Two approaches are proposed for the objective of minimizing total number of workers, total

travelling distance and total waiting time. One is a construction heuristic combined with simulated annealing and the other utilizes the network flow model and combines it with simulated annealing.

Dohn, Kolind and Clausen, (2007) introduced more complexity to the manpower allocation problem with the predetermined and limited number of teams and the possibility of the task's requirement of multiple teams to cooperate. The objective is to minimize the tasks that are not covered due to the limited number of teams and skill requirement of tasks. The problem is solved using the Branch-and-Price approach.

2.3.4. Workforce Scheduling and Routing Problem

Almost all studies we explored aim to effectively schedule workforces while considering the travelling times between locations which is the routing part of the problem. Since the term 'Workforce Scheduling and Routing Problem (WSRP)' was first introduced by Castillo-Salazar, Landa-Silva, and Qu (2012), the practice of using distinct terms for each industry has evolved into the adoption of a more general term that can be applied across all industries. This thesis explores the complex needs of modern services in a more customer-focused manner in the context of the WSRP.

Algethami, Pinheiro, and Landa-Silva (2016) examined the Workforce Scheduling and Routing Problem (WSRP) using a real-world Home Health Care (HHC) dataset. They identified skill requirements and maximum working hours as hard constraints while aiming to enhance both customer and worker satisfaction by including several soft constraints in the objective function. To tackle this challenge, they developed a tailored Genetic Algorithm (GA) featuring indirect chromosome encoding and the application of diverse genetic operators. Their implementation of a tailored GA, serves as a foundation for our exploration of chromosome encoding and genetic operators that will be presented in subsequent sections of this thesis.

Xie, Potts, and Bektaş (2017) primarily investigated the Workforce Scheduling and Routing Problem (WSRP) through an iterated local search (ILS) approach, comparing its effectiveness with a mixed integer programming (MIP) model and an adaptive large neighbourhood search (ALNS) algorithm, as previously explored by Kovacs et al. (2012). They formulated the WSRP as a mixed integer programming problem, aiming

to minimize outsourcing and traveling costs—a critical objective for enhancing operational efficiency in workforce management. Their findings demonstrate that ILS consistently outperforms the ALNS algorithm, particularly in larger instances, highlighting its potential for more effective workforce scheduling and routing solutions.

Building on the foundational work of Kovacs et al. (2012) and Xie, Potts, and Bektaş (2017), Gu, Zhang, and Zinder (2022) introduced a novel optimization technique for the Workforce Scheduling and Routing Problem (WSRP), termed Lagrangian Iterated Local Search (Lagrangian ILS). This method, first proposed in 2019 by the same authors, innovatively combines iterated local search with Lagrangian relaxation to enhance solution accuracy and reduce computational time. Their computational results showed better performance over the CPLEX and the ILS approach used by Xie, Potts, and Bektaş (2017).

CHAPTER 3: PROBLEM DESCRIPTION

Consider an IT company that provides field services. It manages a set of tasks, along with field service workers (employees) who will perform those tasks. Each employee is equipped with a set of skills. Each task is assigned to a specific category that determines the task's working duration and skill requirement. These categories do not impact the computational aspects of the model or the optimisation process. Instead, they help organise tasks by grouping those with similar characteristics, making it easier to manage task attributes. This approach enhances the clarity of the model, ensuring inputs are easy to handle without affecting the solution. In addition to the category-specific attributes, every task has three attributes: priority, arrival time, and SLA (service-level agreement, which is the required duration from arrival to service completion time). Finally, the travel times among all task locations, including the depot, are available.

Given such a set of tasks and a set of employees, the goal is to schedule the workforce (assignment of tasks to employees and the order in which they are to be completed) such that tasks are completed as effectively as possible in the presence of all the constraints. This may entail some tasks running late or even deferred to the following days. In order to quantify the effectiveness of a schedule, one must consider the priority of each task and adherence to the SLA. To this end, we introduce a scoring mechanism that provides control over our model's objective by allowing the contribution of each input to be adjusted through weighting. By adjusting these weights, we can shift the focus of our scheduling priorities to match specific operational goals, enhancing the model's adaptability to different scenarios. For example, increasing the weight of deadline input means prioritising the timely completion of tasks, making compliance with service level agreements the primary focus of our scheduling. With the given inputs, we aim to find a sequence of tasks that will be assigned to employees in such a way as to obtain maximum points, considering all the travel time and employee skill versus task requirement constraints. Tasks that are assigned to an employee should be completed before the end of work hours (18:00 or 480 units), and the given SLA times should be taken into consideration while planning the day.

After the scheduling process, we will output a detailed plan showing which employees will handle which tasks in which order, along with the estimated start and completion times for each task. Note that, in essence, we are doing offline scheduling, with a one-day time horizon. Nonetheless, the “any time” nature of the algorithms we use is also suitable for dynamic planning, as new jobs arrive, some jobs get cancelled, some employees do not show up, or tasks or travel times are prolonged. Furthermore, our analysis of how solution quality changes as algorithm execution times change provides insights into what algorithm to use under what circumstances and for how long.

CHAPTER 4: SOLUTION APPROACH

4.1. Model Description

In the presented mixed integer programming (MIP) model, the goal is to assign tasks to employees in such a way as to obtain maximum points feasibly. In order to achieve feasibility, some soft and hard constraints should be satisfied in the assignment process. In this model, *N* is the set of tasks, which also represents the customer locations, and *K* is the set of employees. There is also a set of skills $Q = \{0...5\}$ and a set of priorities $P = \{1, 2, 3, 4, 5\}$ where 5 is the most, and 1 is the least important priority. Task 0 denotes the central depot where all employees should depart at the start and arrive at the end of the day. Every task $i \in N$ has category-specific attributes, which are working duration d_i and skill requirement $q_i \in Q$. Based on the category of task, durations and skill requirements vary. Each task $i \in N$ also has predefined attributes such as arrival time a_i, priority p_i and SLA time l_i. Arrival time a_i refers to the moment a customer first submits the task, priority p_i indicates its level of urgency and importance, and SLA time l_i specifies the time within which a task should be completed after its arrival to meet customer expectations. Lastly, we have a variable denoted as s_{ij} representing the travelling time from task *i* to task j $i, j \in N$.

Every employee has a set of skills $R_k \subseteq Q$ and in order to assign a task to an employee, the employee should have the necessary skills. A task cannot be executed by more than one employee, and once an employee starts executing a task, it cannot be interrupted (non-preemptive). There are two constants: start of work hours and end of work hours. For convenience, the start time is arbitrarily assigned to be 0, and the end of work hours E is assigned to be 480 without loss of generality. All employees must be at the central depot 0 by the end of work hours *E*. Deadline of task $i \in N$ is denoted and computed as $\beta_i = a_i + l_i$ and a binary variable q_{ik} which indicates whether employee *k* has the necessary skill for task *i* is computed, using predefined variables q_i and R_k.

4.1.1. Score-Generation Function

Our score-generation function $f(i)$ (1) simplifies the model by reducing two adjustable inputs through weights into a single score, thereby making the model both more generic and extensible. Additionally, we capture the non-linear aspects of customer satisfaction on how well the deadlines are met as pre-computed coefficients that get incorporated into a linear model for faster optimisation. In this function, the parameters w_α, determines the contribution of priority p_i and both w_β and w_γ, determine the contribution deadline β_i to the score that will be generated. Thus, jobs with higher priority and earlier deadlines are incentivized to be completed.

4.1.2. Objective Function

The objective function (2) is a maximisation function, and we are trying to acquire the maximum points while assigning tasks to the employees with respect to the constraints and penalties. This function is decomposed into three distinct components.

The first component is the $\sum_{k\in K}\sum_{i\in N} y_{ik} f(i)$, calculates the total score from tasks assigned to employees where y_{ik} here is a binary decision variable that indicates if task $i \in N$ is assigned to employee $k \in K$ and $f(i)$ quantifies the task score. The second component, $-w_1 \sum_{k\in K}\sum_{i\in N} b_{ik}$, penalizes the tasks that are completed after their deadlines, thereby making SLAs a soft constraint and leveraging adherence to them. Lastly, the third component, $-w_2 \sum_{i\in N}\sum_{j\in N}\sum_{k\in K} x_{ijk}\, s_{ij}$, penalising the time spent travelling to tasks, which discourages choosing long paths over short paths, especially when there are enough employees to complete all tasks effortlessly. While lowering operational costs this also promotes efficient routing and scheduling. x_{ijk} in the third component is a binary decision variable that denotes employee $k \in K$ is assigned to task $j \in N$ just after completed task $i \in N$ and s_{ij} is the travel time from task $i \in N$ to task $j \in N$.

4.1.3. Constraints

Constraints (3) ensure that a task can be executed at most by one employee and that some tasks can remain unassigned. Constraints (4) and (5) guarantee employees start

from the central depot and end at the same central depot, while constraints (6) maintain flow conservation, i.e. once an employee arrives at a customer location, that employee must leave that location. Constraints (7) are to avoid revisiting the location after leaving that location. Constraints (8) and (9) guarantee that if an employee arrives at a task location from another task location, both tasks will be executed by that employee, and by no other. The finishing time of a task must be greater or equal to the previous task's finishing time plus the travelling time to the task and the working duration of the task (10). While (11) enforces a minimum finishing time for the first task of an employee, (12) enforces maximum finishing times for all tasks of an employee to be no more than working shift duration. (13) guarantees that no tasks can be assigned to employees who do not have the necessary skill requirement and (14) ensures all employees are at the central depot before the working duration ends. Constraints (15) and (16) allow us to assign a penalty if the task's deadline has passed before finishing it. Constraints (17) – (20) define the decision variables.

Table 1. Variables, parameters and sets of the problem

Sets	
N	Set of Tasks plus task 0 for the depot.
K	Set of Employees
P={1..5}	Set of Priorities, 5 being most important or urgent and 1 least
Q	Set of Skills
Input Parameters	
d_i	Working duration of task *i* based on the category of task $i \in N$
a_i	Arrival time of task $i \in N$
p_i	Priority of task $i \in N$
s_{ij}	Travelling time from task *i* to task $j \ \ \forall i, j \in N$
l_i	SLA time of task $i \in N$
q_i	Skill requirement of task *i* based on the category $i \in N$
R_k	Skill set of employee $k \in K$
Derived Variables	
$\beta_i = a_i + l_i$	Deadline of task *i* computed via arrival time and SLA
q_{ik}	Binary variable indicating whether employee *k* has the necessary skill for task *i* and derived from $q_i \in R_k$

Table 1 (Continued). Variables, parameters and sets of the problem

Tuning Parameters	
w_α	Weight of priority
w_β	Weight of deadline
w_γ	Weight of deadline
w_1	Weight of tardiness penalty
w_2	Weight of total travelling time penalty
Decision Variables	
y_{ik}	Binary variable indicating whether employee $k \in K$ is assigned to task $i \in N$
x_{ijk}	Binary variable indicating whether task $i \in N$ is visited just before task $j \in N$ by employee $k \in K$
u_{ik}	Finishing time of task $i \in N$ by employee $k \in K$
b_{ik}	tardiness variable for penalising when assigned tasks deadline passes

$$f(i) = w_\alpha p_i + w_\beta e^{-w_\gamma \beta_i} \tag{1}$$

$$max \sum_{k\in K}\sum_{i\in N} y_{ik} f(i) - w_1 \sum_{k\in K}\sum_{i\in N} b_{ik} - w_2 \sum_{i\in N}\sum_{j\in N}\sum_{k\in K} x_{ijk}\, s_{ij} \tag{2}$$

$$\sum_{k\in K} y_{ik} \leq 1 \qquad \forall i \in N \ \backslash\{0\} \tag{3}$$

$$\sum_{j\in N} x_{0jk} = 1 \qquad \forall k \in K \tag{4}$$

$$\sum_{j\in N} x_{j0k} = 1 \qquad \forall k \in K \tag{5}$$

$$\sum_{j\in N} x_{ijk} - \sum_{j\in N} x_{jik} = 0 \qquad \forall k \in K, \forall i \in N \tag{6}$$

$$x_{iik} = 0 \qquad \forall k \in K, \forall i \in N \backslash\{0\} \tag{7}$$

$$\sum_{j\in N} x_{jik} - y_{ik} = 0 \qquad \forall k \in K, \forall i \in N \tag{8}$$

$$\sum_{j\in N} x_{ijk} - y_{ik} = 0 \qquad \forall k \in K, \forall i \in N \tag{9}$$

$$E\left(x_{ijk} - 1\right) + x_{ijk}\left(s_{ij} + d_j\right) + u_{ik} - u_{jk} \leq 0 \qquad \forall i \in N, \forall j \in N\backslash\{0\}, \forall k \in K \tag{10}$$

$$y_{ik}(s_{0i} + d_i) \leq u_{ik} \qquad \forall k \in K, \forall i \in N \tag{11}$$

$$u_{ik} \leq E y_{ik} \qquad \forall k \in K, \forall i \in N \tag{12}$$

$$y_{ik} \leq q_{ik} \qquad \forall k \in K, \forall i \in N \tag{13}$$

$$\sum_{i\in N}\sum_{j\in N} x_{ijk}(s_{ij} + d_j) \leq E \qquad \forall k \in K \tag{14}$$

$$u_{ik} - y_{ik}\beta_i \leq b_{ik} \qquad \forall k \in K,\ \forall i \in N \tag{15}$$

$$b_{ik} \geq 0 \qquad \forall k \in K,\ \forall i \in N \tag{16}$$

$$x_{ijk} \in \{0,1\} \qquad \forall k \in K,\ \forall (i,j) \in N \tag{17}$$

$$y_{ik} \in \{0,1\} \qquad \forall k \in K,\ \forall i \in N \tag{18}$$

$$f_{ik} \in \mathrm{Z}^{\square} \qquad \forall k \in K,\ \forall i \in N \tag{19}$$

$$b_{ik} \in \mathrm{Z}^{\square} \qquad \forall k \in K,\ \forall i \in N \tag{20}$$

4.2. Solution Methods

This subsection introduces solution methods used in benchmarking the WSRP. One exact method and two heuristic search methods were used to tackle the problem. A mathematical solver was used with the presented MIP model, which uses a branch-and-bound algorithm as an exact method; a tabu search algorithm and a genetic algorithm were used as heuristic search methods.

4.2.1. Mixed Integer Programming Approach

In addressing the WSRP, we employed the Gurobi Solver to solve the MIP model detailed in the previous section. We selected Gurobi for its robust performance in handling large-scale integer programming challenges, which is substantial given the complexity and size of typical WSRP scenarios. Additionally, Gurobi's parallel processing capabilities are crucial for efficiently managing the computational demands of our model. The implementation was carried out using Gurobi's default parameters.

Although MIP is an exact method, Gurobi's parameter which allows us to specify a time limit, provided us flexibility using MIP like a heuristic search method. Thereby making the comparison possible with other heuristic search methods.

4.2.2. Tabu Search Algorithm

In tackling the Workforce Scheduling and Routing Problem (WSRP), we applied a Tabu Search algorithm that uses different strategies and rules to explore possible solutions effectively. First, our algorithm generates the initial solution randomly by assigning tasks to employees feasibly. Then we find new solutions in every iteration by executing feasible and non-tabu moves. Our algorithm employs four types of moves: inter-swap, intra-swap, insert, and remove, each designed to explore different aspects of the solution landscape.

Inter-Swap This move involves swapping an unassigned task with an assigned task. We generate a candidate list of feasible inter-swaps and sort them by the potential change in the objective value. If available, the best non-tabu or aspirational move is executed.

Intra-Swap Similar to the inter-swap, but it involves swapping two assigned tasks that are not necessarily within the same employee's schedule. We create a candidate list for feasible intra-swaps and again sort by the change in objective value, selecting the best non-tabu or aspirational move.

Insert This move selects an unassigned task and inserts it into an employee's schedule without creating a candidate list. Tasks are sorted by their score, and insertion is attempted for each employee provided the move is not tabu or meets the aspiration criteria.

Remove If the solution does not improve after specified iterations, a task with the lowest score from an assigned task list is removed. This move is executed if it's not tabu, aiming to explore new areas of the solution space (diversification) that might lead to an overall better configuration.

To deal with situations where no improvements occur, additional strategic responses, intensification and diversification techniques, are employed:

Reversion to Best-Known Solution (Intensification) If there is no improvement in the objective value after a specified number of iterations, the algorithm reverts to the best solution found up to that point and continues the search from there but this time with all tabu list empty.

Intensive Removal Phase (Diversification) After another specified period without objective improvement, the algorithm diversifies its search, aiming to escape from local optima by iteratively removing half of the assigned tasks. Thus, the solution landscape is significantly changed to find better solutions possibly.

For every move type, we defined a distinct tabu list with a defined tenure to avoid cycling back to previously visited solutions and encouraging the exploration of new areas in the solution space. By sometimes allowing moves that make the score worse, the algorithm can escape local optima and explore more broadly for potentially better solutions.

4.2.3. Genetic Algorithm

Our Genetic Algorithm (GA) begins by randomly generating the initial population by assigning tasks to employees feasibly. To maintain diversity, parent selection for crossover is performed using the roulette wheel strategy. Roulette wheel selection chooses parents by assigning probabilities proportional to their fitness values. Parents are then selected based on those probabilities, ensuring that individuals with higher fitness have a greater chance of being chosen. The crossover operation is designed to respect feasibility constraints by randomly selecting either odd or even-numbered employees, retaining their task sequences while recombining them into new offspring. After the recombining phase, a repair operation is applied to the chromosomes that violate feasibility by swapping duplicate tasks with unassigned feasible tasks.

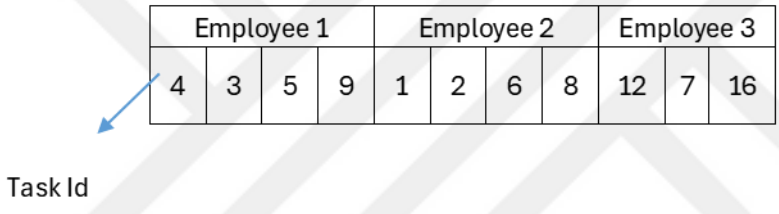

Figure 5. Chromosome represantation

Following the crossover operation, mutation is applied across the combined pool of parents and offspring to enhance solution variability. The mutation operation is performed by randomly selecting employees from the population and reassigning tasks feasibly. Unassigned tasks are sorted by their score and reassigned using a restricted candidate list (RCL), a concept taken from the greedy randomised adaptive search procedure (GRASP). Instead of selecting the highest-scoring tasks, tasks are chosen randomly from the top specified number of highest-scoring tasks in the RCL to maintain solution diversity.

The selection of the next generation is carried out through an elitist strategy, where the top *n* solutions from the current population based on fitness are carried forward to the next generation. Here, *n* is the initial population count, and the current population is the combined pool of parents and offspring. This method ensures that the best solutions

are retained, providing a steady and fast convergence towards the optimal solution while trading off some diversity.

CHAPTER 5: EXPERIMENTAL SETUP

In this chapter, we will present the problem set used for our experiments and computational studies. It's worth noting that during the literature review, we discovered that the majority of the problem set used in WSRP and WSRP-related topics is the dataset generated for the vehicle routing problem with time windows (VRPTW) by Solomon (1987) (Castillo-Salazar, Landa-Silva, and Qu, 2012), (Zhang and Li, 2024), (Kumar and Panneerselvam, 2015). Therefore, to maintain consistency and comparison simplicity, our computational experiments involved a combination of Solomon's dataset and the dataset generated by Homberger and Gehring (1999) according to the procedure reported by Solomon (1987).

In order to align these datasets with the focus of our thesis while preserving their original features, we made some modifications based on the industry consultations we performed. Insights from IT professionals revealed that tasks requiring higher levels of skill occur less frequently than those requiring basic skills. Additionally, senior employees who handle complex tasks tend to be fewer compared to junior employees who handle simpler tasks. These insights guided our modifications. We solved these problems on a computer with Windows 11 OS, Intel(R) Core(TM) i7-10700K CPU, 32 GB 3200 MHz RAM. We compared the computational performance and solution quality of the approaches described in the previous section.

In Solomon's dataset, there are 56 instances, each including 100 visits, categorised into six groups, R100, R200, C100, C200, RC100, and RC200, based on the planning horizon's length and the geographic distribution of visits. The geographical locations and customer demands in these datasets are based on the benchmark problems presented by Christofides, Mingozzi, and Toth (1979). These datasets are categorised based on their distribution patterns as randomly distributed (R1 and R2), clustered (C1 and C2), and semi-clustered (RC1 and RC2), where the semi-clustered datasets are a combination of random and clustered data. Shorter planning horizons of 230 to 240 minutes are characteristic of the R100 and RC100 groups, whereas the RC200, R200, C100, and C200 groups feature longer horizons exceeding 900 minutes. The vehicle capacity constraints restrict more than a few customers from being visited by the same vehicle for the R1, C1, and RC1 groups; in contrast, groups R2, C2, and RC2 with

large vehicle capacities allow many more customers to be visited by the same vehicle in a scheduling horizon. Additionally, the service times vary, with 10-minute durations in the R and RC groups and 90-minute durations in the C groups. The objective often includes minimising the fleet size necessary to cover all visits, with distance and travel times being identical and represented through a symmetrical Euclidean matrix.

The second dataset we used, developed by Homberger and Gehring, consists of five distinct sets of routing and scheduling problems. Five groups labelled G02, G04, G06, G08, and G10 comprised of 200, 400, 600, 800, and 1000 customers, respectively. These expanded groups aim to offer a more realistic range of problem sizes and are divided into 60 distinct instances grouped across six classes: C1, C2, R1, R2, RC1, and RC2, mirroring Solomon's original classification. These classes were developed following the methods Solomon reported, ensuring consistency with the original set. The coordinates of customers and depots are specified in a cartesian system with integer values.

As we tailored Solomon's as well as Hoberger and Gehring's datasets, we assigned categories to each task, which corresponds to the customers in the original datasets, numbered from 0 to 10. The assignment follows a left-skewed probability distribution, making lower categories more likely and reflecting the higher occurrence of simpler, low-level tasks. As detailed in the problem definition section, these categories determine each task's required skill level and working duration. Higher category numbers indicate tasks that require greater skills and more time, justifying our use of a left-skewed distribution to simulate the realistic distribution of task complexity in IT services. Additionally, these consultations we performed with industry professionals influenced our approach while generating employees with skill sets which correspond to vehicles in original datasets. We generated 20% of employees with senior skill sets, 30% of employees with mid-level skill sets, and 50% of employees with junior skill sets.

As previously mentioned, our deadlines are determined by adding the SLA to the task's arrival time. In order to keep the modification minimum, instead of generating arrival times and SLA values, we used the due time of the original dataset with a slight modification to adapt our one-day horizon plan. Due to the longer due date ranges, especially in the R2, C2, and RC2 groups, we normalised the due dates to make the

average of our deadlines to be 400, which is slightly smaller than our end-of-work hours (480).

Lastly, task priorities are assigned using a uniform distribution from 1 to 5, with 5 being the most urgent/important. After all the features that are required for the score generation function are generated, scores for each task are calculated as previously detailed in the problem definition section.

After the modifications to the data sets, the weights are assigned both in the score generation function and the objective function, which is crucial for ensuring the model's alignment with the operational goals. In the score generation function, we applied three weights to adjust the trade-off between task priority and compliance with the SLA, $w_\alpha = 4$ controls priority while $w_\beta = 0.003$ and $w_\gamma = 90$ adjust for adherence to SLA.

Similarly, in the objective function, we use two distinct weights: $w_1 = 0.06$ penalises the violations of assigned tasks deadlines (tardiness), helping maintain adherence to SLAs and the $w_2 = 0.1$ aims to reduce travelling time, enhancing route efficiency.

The parameters for our computational experiments, such as the size of the population in genetic algorithms, cross-over rate and mutation rate or the tenures of the tabu lists in tabu search, were carefully chosen based on a series of preliminary runs for different problem sizes. These runs helped establish suitable settings that balance exploration and exploitation.

With the modified datasets, we generated 108 instances from Solomon's dataset and 96 from the Homberger and Gehring dataset, using various combinations of task and employee counts. These combinations are 25x2, 25x3, 50x3, 50x4, 50x6, 100x4, 100x6, 100x8, 100x12, 200x8, 200x12, 200x16, 200x24, 400x16, 400x24, 400x32, and 400x48. Here, the first term represents task count, and the second term represents employee count. For each combination, we conducted experiments on 12 distinct instances, totalling 204 experiments per method. Different timeboxes were chosen based on a series of preliminary tests for different problem sizes, such as 5 minutes for 25x2 and 25x3, 10 minutes for 50x3 and 50x4, 15 minutes for 50x6 and 100x4, 20 minutes for 100x6, 30 minutes for 100x8, 45 minutes for 100x12 and 60 minutes for 200x8, 200x12, 200x16, 200x24, 400x16, 400x24, 400x32, 400x48. With the information acquired from the IT professionals for accurately simulating real life, a

maximum 60-minute time limit was chosen, as planning takes place in the morning before dispatching the employees.

CHAPTER 6: EXPERIMENTAL RESULTS AND DISCUSSION

In our computational experiments, we analysed the performance of three methods, genetic algorithm (GA), Mixed Integer Programming (MIP), and tabu search (TS), across different problem sizes ranging from 25 to 400 tasks. The results show that while MIP often found the best objective values, especially in smaller problem sizes, the differences in solution quality between the methods were minimal, showing their effectiveness in finding near-optimal solutions.

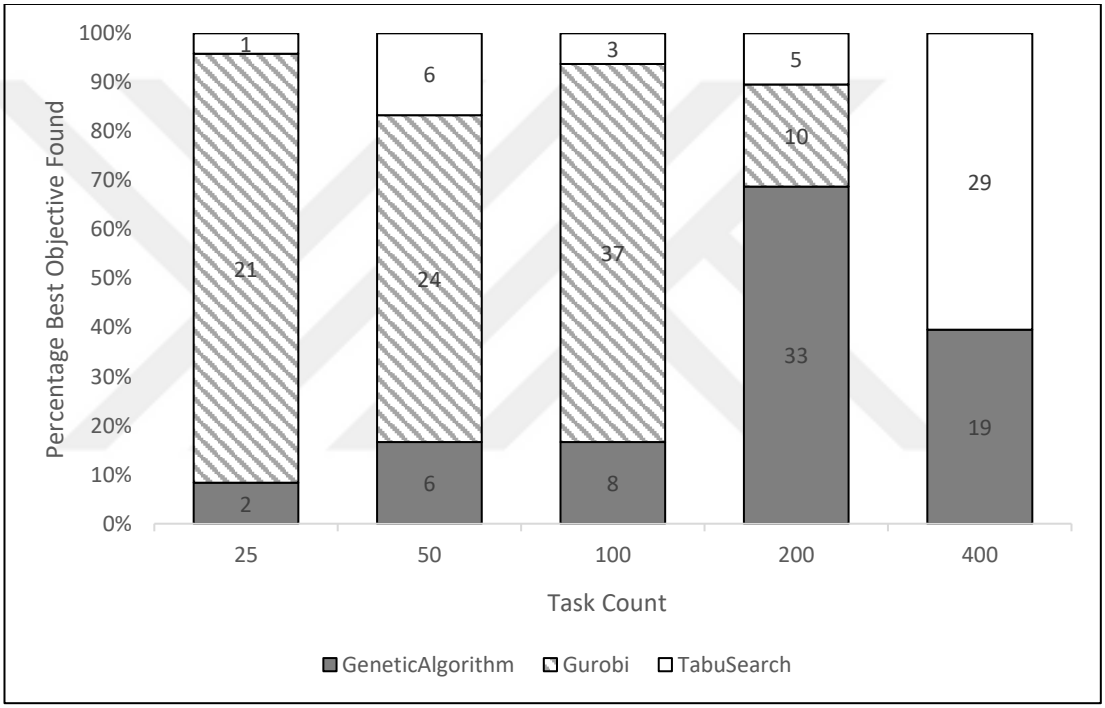

Figure 6. Methods that found the best objective by task count

MIP found the optimal solution in 10 out of 204 instances and those optimal results found were obtained from 25-task scenarios. Despite MIP's strong performance in these cases, the TS and GA were also effective, closely approaching the best results. For instance, in the 25x3 problem size, TS and GA both reached 98.35% of the best objective values found by MIP as we can see in Table 3. This shows that in smaller problem sizes, while MIP often finds the best results, the other two methods are very close. Also, as we presented in both Figure 5 and Table 2, MIP achieved the best solutions in 21 out of 24 instances in the 25-task scenarios, 24 out of 36 in the 50-task

scenarios and 37 out of 48 in the 100-task scenarios; yet again, the GA and TS were close competitors, reaching objective values within a minimum 98% of the best-found solutions in the 25 to 100-task scenarios. While in the 200-task scenarios, GA obtained the best objective values in 33 out of 48, TS obtained the best objective values in 29 out of 48 in the 400-task scenarios. Although this close performance continued across larger problem sizes, MIP ran out of memory (OoM) in the 400x32 and 400x48 scenarios.

Table 2. Count of best objective found

Best Objective Found				
Problem Size (Task x Employee)	**Genetic**	**MIP**	**Tabu**	**Total**
25x2	1	10	1	12
25x3	1	11		12
50x3	1	9	2	12
50x4	5	4	3	12
50x6		11	1	12
100x4		12		12
100x6	2	7	3	12
100x8	2	10		12
100x12	4	8		12
200x8	5	7		12
200x12	6	3	3	12
200x16	12			12
200x24	10		2	12
400x16	12			12
400x24	5		7	12
400x32	2	OOM	10	12
400x48		OOM	12	12
Total	**68**	**92**	**44**	**204**

Table 3. Average solution quality as a percentage of best-found solution

Problem Size (Task x Employee)	Tabu	MIP	Genetic
25x3	98.35%	100.00%	98.35%
50x6	98.50%	99.90%	98.73%
100x12	98.94%	99.78%	99.43%
200x24	97.57%	94.32%	98.83%
400x48	100.00%	OOM	97.40%

Additionally, we analysed the time efficiency of these methods across various combinations of tasks and employees, maintaining a constant employee-to-task ratio in our samples. Specifically, we examined configurations such as 25 tasks with 3 employees (25x3), 50 tasks with 6 employees (50x6), 100 tasks with 12 employees (100x12), 200 tasks, 24 employees (200x24) and 400 tasks with 48 employees (400x48). The reported results in Table 4 represent the average run time required for each method to achieve solution quality within certain percentiles of the best-known objective value for that instance.

For example, the 95th percentile indicates the time an algorithm takes to reach a solution that is within 95% of the best-found solution at that instance. We can observe from Table 4 that as the problem size increases, significant differences occur in performance among the methods. In smaller problem sizes (e.g., 25x3 and 50x6), all three method's performances were decent, with quick convergence times even at higher percentiles. As the problem size grew (100x12 and 200x24), MIP's runtime increased gradually, especially noticeable at the 95th percentile, where it took over 3425 seconds for the 200x24 problem size. As mentioned previously, at the largest problem size (400x48), MIP couldn't produce any solution, resulting in out-of-memory errors (OOM), while GAs convergence times significantly increased, reaching up to 1117 seconds at the 95th percentile. Tabu Search, on the other hand, maintained its stability with 64.55 seconds at the 95th percentile.

Table 4. Solution quality (%) versus run time (sec) of each method for various problem sizes (* best result)

25x3 (Task x Employee)					50x6 (Task x Employee)			
Solution Quality	**Tabu**	**MIP**	**Genetic**		**Solution Quality**	**Tabu**	**MIP**	**Genetic**
	Run Time (sec)					**Run Time (sec)**		
75%	**0.01***	0.22	0.02		**75%**	**0.01***	2.37	0.04
80%	**0.01***	0.30	0.02		**80%**	**0.02***	3.32	0.05
85%	**0.02***	0.35	0.03		**85%**	**0.03***	5.16	0.07
90%	0.41	0.57	**0.14***		**90%**	**0.18***	7.67	0.31
95%	14.55	1.39	**1.17***		**95%**	13.50	24.98	**1.54***
100x12 (Task x Employee)					**200x24 (Task x Employee)**			
Solution Quality	**Tabu**	**MIP**	**Genetic**		**Solution Quality**	**Tabu**	**MIP**	**Genetic**
	Run Time (sec)					**Run Time (sec)**		
75%	**0.02***	11.99	0.07		**75%**	**0.24***	245.54	1.34
80%	**0.04***	16.27	0.11		**80%**	**0.33***	551.59	3.17
85%	**0.08***	50.79	0.45		**85%**	**0.48***	858.94	12.07
90%	**0.27***	124.13	2.47		**90%**	**0.81***	1678.76	53.26
95%	**5.04***	285.05	10.22		**95%**	441.79	3425.72	**289.45***
400x48 (Task x Employee)								
Solution Quality	**Tabu**	**MIP**	**Genetic**					
	Run Time (sec)							
75%	**3.16***	OoM	8.89					
80%	**4.26***	OoM	19.33					
85%	**5.84***	OoM	46.29					
90%	**8.24***	OoM	187.98					
95%	**64.55***	OoM	1117.07					

This comparative analysis shows that each method, especially in the smaller problem sizes (25, 50, and 100 task scenarios), can easily find good enough solutions in a reasonable time. As the problem size grows, MIP's computational times exponentially increase, resulting in an out-of-memory at the problem size greater than 400x24. For solution qualities, while differences aren't significant until 400-task scenarios we observed that MIP is working best at smaller instances (25x2, 25x3, 50x3, 50x4, 50x6, 100x4, 100x6, 100x8, 100x12, 200x8), GA is best at mid-sized instances (200x12, 200x16, 200x24, 400x16) and TS is in its peak performance at largest instances (400x24, 400x32, 400x48). If we were to evaluate in terms of time efficiency, although in the smaller problem sizes like 25 to 100-task scenarios, the differences can be tolerated, the superiority of TS is easily observable in all instances.

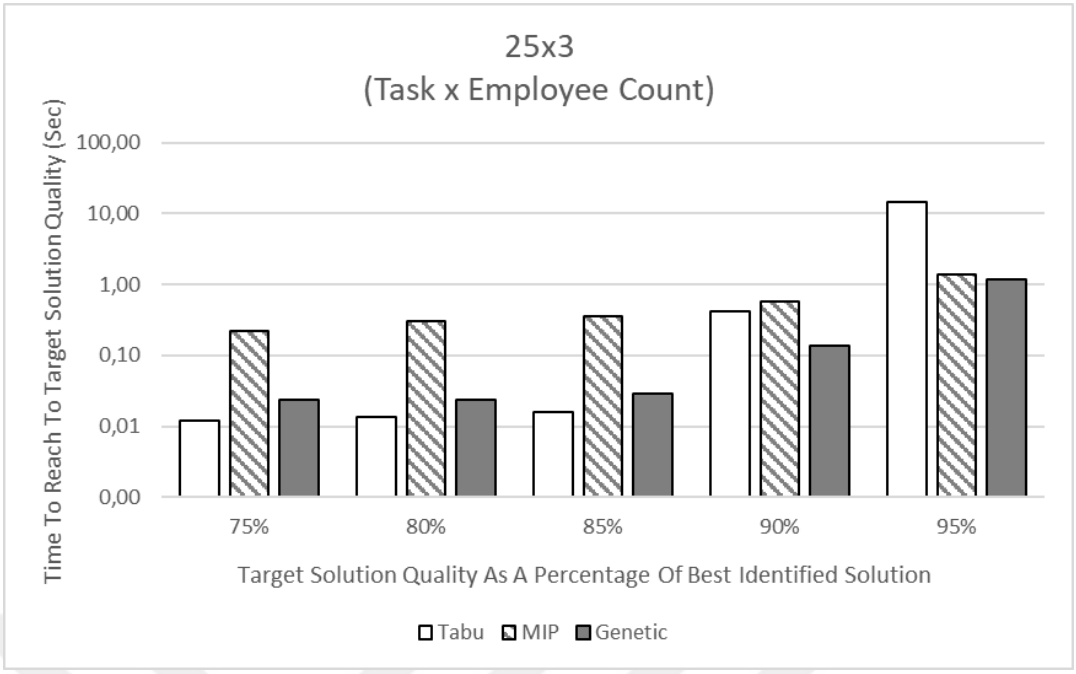

Figure 7. Convergence times of methods in a logarithmic scale to target solution quality for 25-task and 3-employee scenario

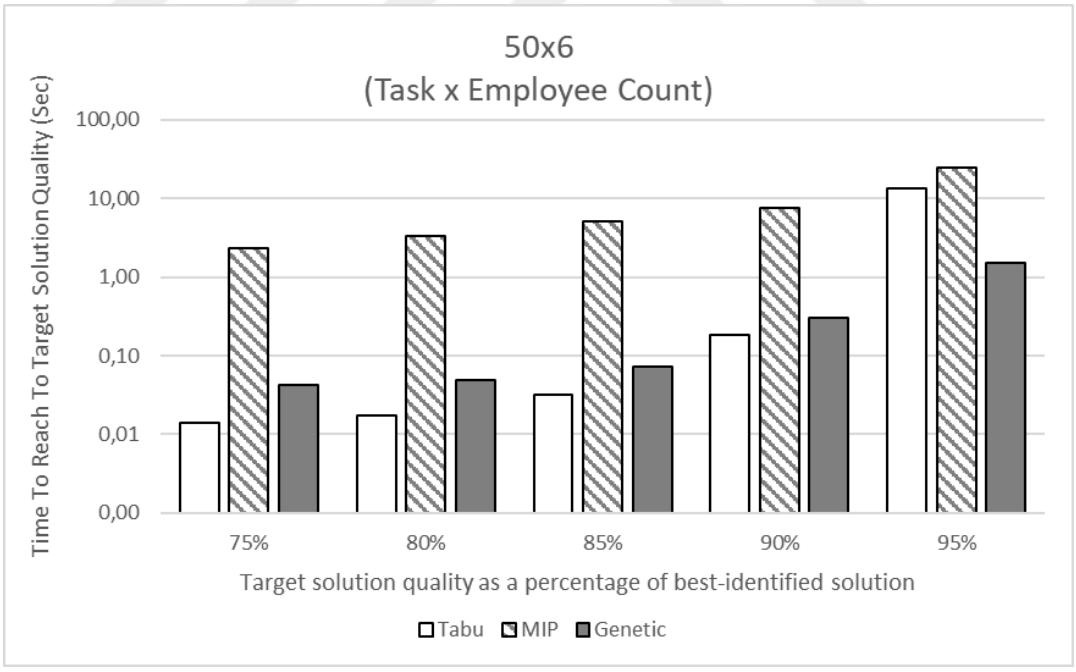

Figure 8. Convergence times of methods in a logarithmic scale to target solution quality for 50-task and 6-employee scenario

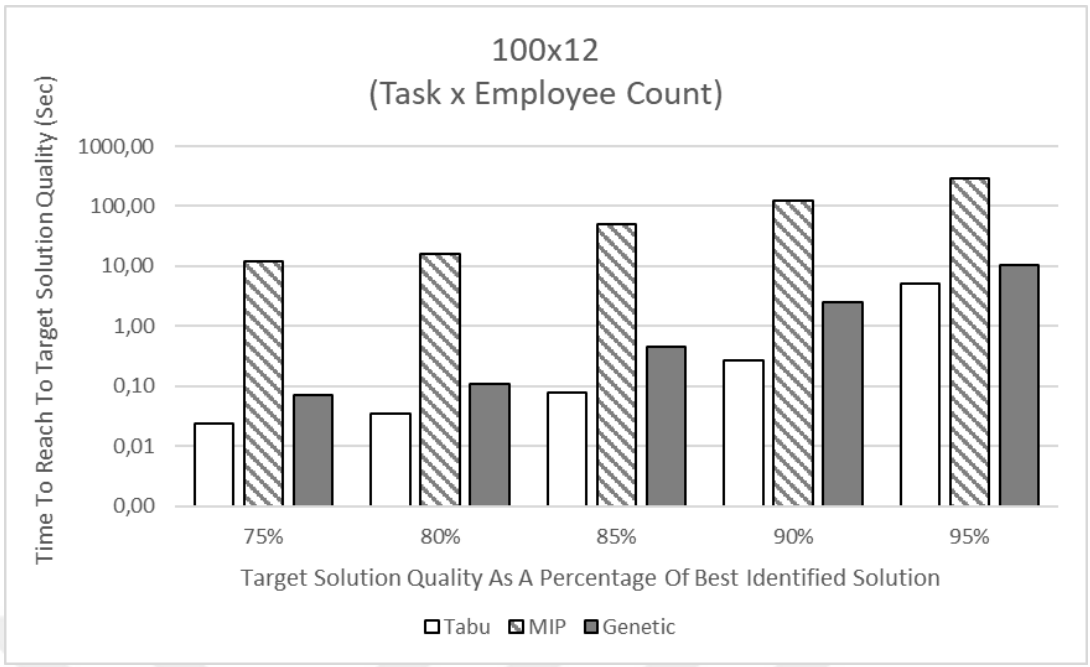

Figure 9. Convergence times of methods in a logarithmic scale to target solution quality for 100-task and 12-employee scenario

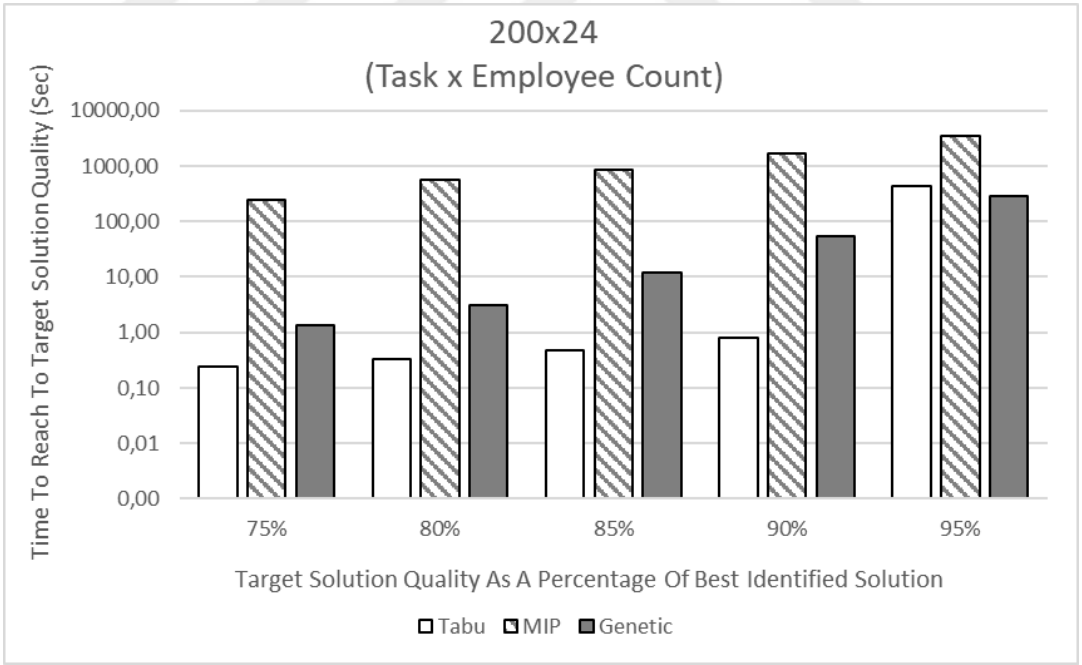

Figure 10. Convergence times of methods in a logarithmic scale to target solution quality for 200-task and 24-employee scenario

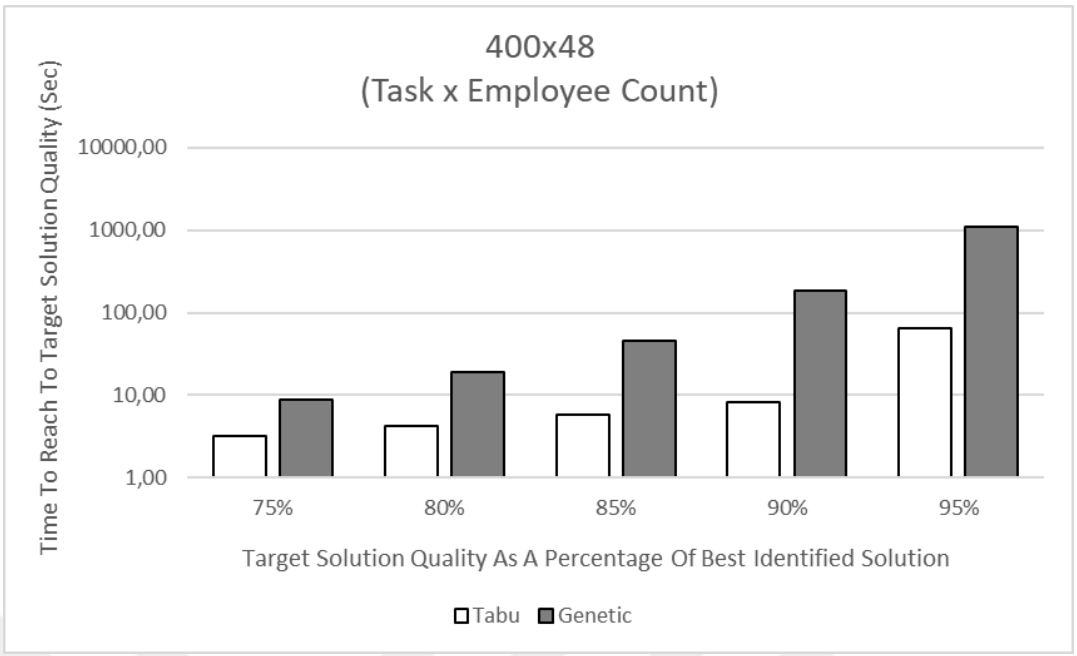

Figure 11. Convergence times of methods in a logarithmic scale to target solution quality for 400-task and 48-employee scenario

Additionally, Figures 6-10 show that, although TS is superior in terms of time efficiency, after convergence to %90 solution quality, the time required to further improve the solution exponentially increased.

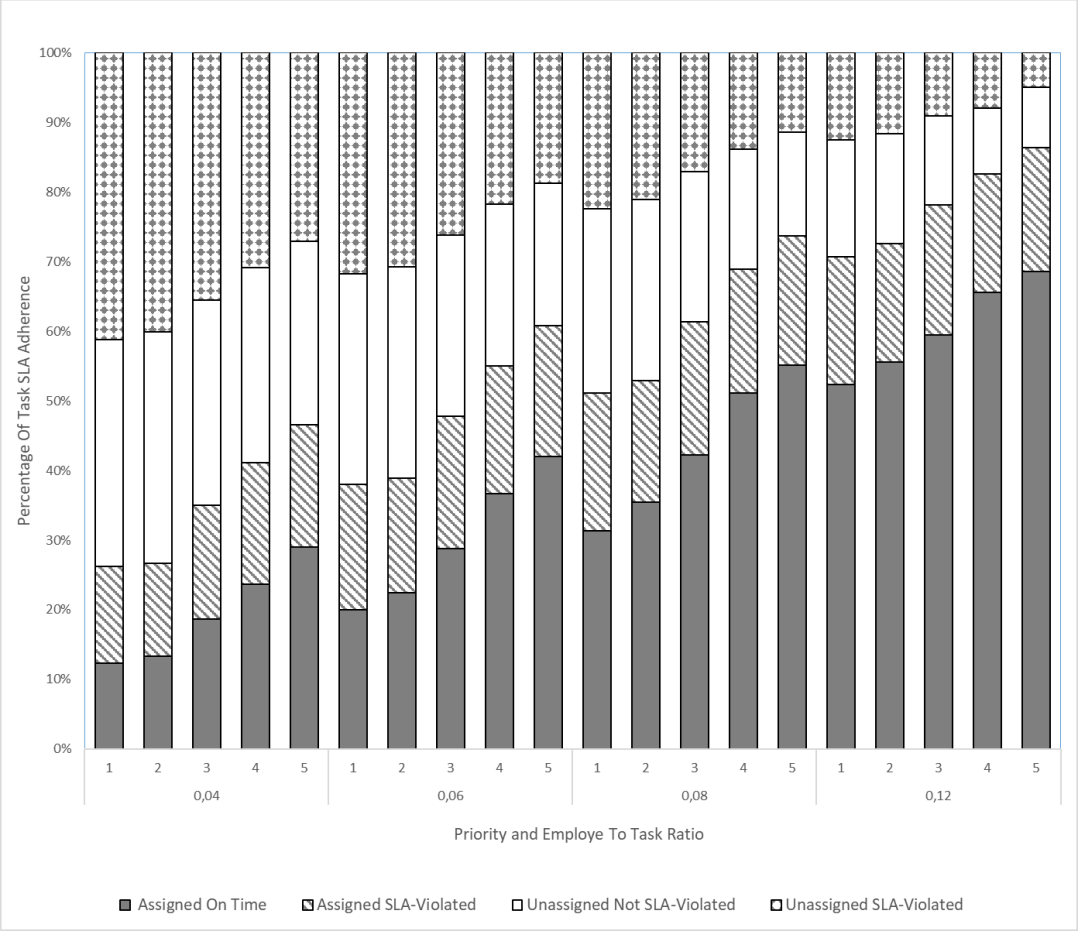

Figure 12. Percentage of average assigned and unassigned task's SLA adherence by priority and employee-to-task ratio

Building on our previous analysis and observations of algorithm performances in terms of runtime and solution quality, we analysed aspects that directly affect customer satisfaction, such as task assignment rates and SLA adherence, across various priorities. Higher priority tasks completed on time and fewer SLA violations can be directly related to better customer satisfaction. The chart in Figure 11 can be beneficial in cases where efficient resource allocation is crucial, especially in scenarios with limited resources, to maximise the completion of high-priority tasks and minimise critical SLA violations. Here, we can easily observe that the higher the grey bar and the lower the diagonal striped bar and diamond grid bar, the better customer satisfaction will be. Additionally, we can safely use the chart to imply the necessary employee-to-task ratio for customer satisfaction needs for business.

Table 5. Change in task assignments and SLA violations by methods and priorities

Methods By Priority	Assigned Task Ratio (%)	SLA Violated Task Ratio (%)	SLA Violation Per Task (Min)
Genetic Algorithm	**63%**	**34%**	**105**
1	52%	39%	122
2	56%	37%	119
3	64%	34%	99
4	69%	29%	88
5	73%	28%	96
MIP	**64%**	**33%**	**118**
1	56%	35%	133
2	57%	34%	135
3	64%	35%	114
4	68%	30%	102
5	72%	31%	107
Tabu Search	**63%**	**34%**	**116**
1	50%	38%	134
2	54%	37%	128
3	63%	36%	110
4	69%	30%	102
5	75%	29%	107

Table 6. Change in task assignments and SLA violations by methods and employee-to-task ratios

Methods By Employee Task Ratio	Assigned Task Ratio (%)	SLA Violated Task Ratio (%)	SLA Violation Per Task (Min)
Genetic Algorithm	**62%**	**34%**	**106**
0,04	37%	50%	139
0,06	52%	41%	115
0,08	66%	31%	105
0,12	81%	22%	78
MIP	**63%**	**34%**	**120**
0,04	39%	48%	162
0,06	53%	40%	140
0,08	70%	30%	111
0,12	84%	20%	76
Tabu Search	**62%**	**35%**	**118**
0,04	35%	50%	145
0,06	51%	41%	131
0,08	66%	32%	117
0,12	82%	23%	92
Grand Total	**62%**	**34%**	**114**

In Table 5, one can immediately observe that in all three methods, tasks with higher priority tended to be assigned more and had fewer SLA violations. For instance, in the highest priority (Priority 5), tasks were assigned more frequently and with the least SLA violations. Therefore, we can assume our model is capable of prioritising urgent and important tasks while minimising SLA violations. This prioritisation is crucial to customer satisfaction in real-world settings.

Furthermore, Table 6 shows that as the employee-to-task ratio increases, the percentage of tasks assigned increases, while the percentage of tasks violating SLAs and the SLA violation per task decreases. This indicates that our model minimises not only the SLA-violating task count but also the SLA-violation extent.

Especially in the limited-resourced nature of the business world, where it's not possible to schedule all tasks within a workday. The correct choice of which task to perform and which task to defer is key to success. Results showed that with its adjustable aspect for business needs, our model is capable of assigning tasks to employees while considering SLA adherence, the importance of tasks, and time spent travelling for operational efficiency.

Finally, we conclude the discussion with algorithm selection for various contexts. In scenarios like dynamic scheduling where time is very limited, the Tabu Search algorithm excels due to its quick convergence times. For small businesses or contexts where problem sizes are relatively small and solution quality is critical, Mixed Integer Programming is the suitable choice. Additionally, for mid-sized problems where solution quality is important, but time is less of a constraint, Genetic Algorithms (GA) become prominent.

CHAPTER 7: CONCLUSION AND FUTURE WORK

In this thesis, we explored the Workforce Scheduling and Routing Problem (WSRP), aiming to present a more business-centric model that can be generically adjusted for the ever-changing nature of business. In order to address the changing business needs and customer satisfaction challenges, we introduce a score-generation function. This score-generation function quantifies customer satisfaction, which is an abstract concept, by integrating some customer-related task attributes controlled with weights. Those customer attributes are the priority of the task and Service Level Agreement (SLA) times, which define the expected time frame within which a task should be completed after its arrival. With the help of this function, we computed a score and assigned it to each task. In our model, we opted for a maximisation function to align with the operational goals of completing the maximum number of tasks while minimising travel times naturally. Furthermore, by integrating customer satisfaction factors such as priority and SLA adherence into the scoring system, the maximisation function also aims to optimise customer satisfaction. This approach ensures that, especially when resources are limited, the objective function not only prioritises short routes that allow for the completion of more tasks, thereby collecting higher scores but also addresses the qualitative aspects of service delivery.

We used one exact and two heuristic search methods adapted for the proposed model and conducted 204 experiments with each method using datasets adapted from Solomon (1987) and Homberger and Gehring (1999). Then we evaluated the results qualitatively and quantitatively by first comparing the runtime and solution quality of the methods and then analysing the aspects that directly affect customer satisfaction. Results show that the solution quality of all three methods was decent enough with the fixed timebox setup until the 400-task scenarios. In the 400-task scenarios, MIP's solution quality was significantly reduced and starting from the 400x32 problem size, MIP couldn't produce any solution, resulting in out-of-memory errors. Also, in terms of time efficiency, the tabu search algorithm's performance was superior to both the genetic algorithm and the mixed integer programming.

Additionally, reported results demonstrated that our model is perfectly capable of assigning tasks to employees, prioritising critical tasks while considering customer satisfaction and minimising travelling times spent.

We see future research potential for integrating machine learning to further refine the selection and configuration of algorithms based on problem characteristics and changing business requirements as we discussed in the previous section. We hypothesize that solution quality can be predicted based on available computation time and resources along with problem size parameters for each algorithm. This would allow business users to decide which algorithm to run for how long to best meet their needs.

REFERENCES

Bellman, R. (1966). *Dynamic programming*. Science, Vol. 153(3731), pp.34–37.

Algethami, H., Pinheiro, R. L. and Landa-Silva, D. (2016). *A genetic algorithm for a workforce scheduling and routing problem*. In 2016 IEEE Congress on Evolutionary Computation (CEC), pp.927-934. IEEE.

Bertels, S. and Fahle, T. (2006). *A hybrid setup for a hybrid scenario: combining heuristics for the home health care problem.* Computers & operations research, Vol. 33(10), pp. 2866-2890.

Bodin, L. (1983). *Routing and scheduling of vehicles and crews.* Computer & Operations Research, Vol. 10(2), pp.69-211.

Brucker, P., Qu, R. and Burke, E. (2011). *Personnel scheduling: Model and complexity*. European Journal of Operational Research, Vol. 210(3): pp.467-473.

Burke, E. K. and Kendall, G. (2014). *Search methodologies: introductory tutorials in optimization and decision support techniques*. 2nd Edition, Springer; 2014.

Castillo-Salazar, A., Landa-Silva, D. and Qu, R. (2012). *A survey of workforce scheduling and routing.* In Proceedings of the 9th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2012), pp.283–302.

Castillo-Salazar, J.A., Landa-Silva, D. and Qu, R. (2016). *Workforce scheduling and routing problems: literature survey and computational study.* Annals of Operations Research, Vol. 239, pp.39-67.

Cheng, E. and Rich, J. (1998). *A Home Health Care Routing and Scheduling Problem*. Technical report CAAM TR98-04. Rice University.

Christofides, N., Mingozzi, A., and Toth, P. (1979). *The vehicle routing problem*. In N. Christofides, A Mingozzi, P. Toth, and C. Sandi (Eds.), Combinatorial Optimization. Chichester: Wiley.

Cordeau, J. F., Laporte, G., Pasin, F. and Ropke, S. (2010). *Scheduling technicians and tasks in a telecommunications company*. Journal of Scheduling, Vol. 13, pp.393-409.

Glover, F. (1986). *Future paths for integer programming and links to artificial intelligence*. Computers & operations research, Vol. 13(5), pp.533-549.

Glover, F., Laguna, M. and Marti, R. (2007). *Principles of tabu search*. Approximation algorithms and metaheuristics, Vol. 23, pp.1-12.

Gu, H., Zhang, Y. and Zinder, Y. (2022). *An efficient optimisation procedure for the workforce scheduling and routing problem: Lagrangian relaxation and iterated local search*. Computers & Operations Research, Vol. 144, p.105829.

Gu, H., Zhang, Y. and Zinder, Y. (2019). *Lagrangian relaxation in iterated local search for the workforce scheduling and routing problem*. In Analysis of Experimental Algorithms: Special Event, SEA² 2019, Kalamata, Greece, June 24-29, 2019, Revised Selected Papers (pp. 527-540). Springer International Publishing.

Holland, J. (1975). *adaptation in natural and artificial systems*. University of Michigan Press google schola, Vol. 2, pp.29-41.

Homberger, J., and Gehring, H. (1999). *Two evolutionary metaheuristics for the vehicle routing problem with time windows.* INFOR: Information Systems and Operational Research, Vol. 37(3), pp.297-318.

Johnson, E. L., Nemhauser, G. L. and Savelsbergh, M. W. (2000). *Progress in linear programming-based algorithms for integer programming: An exposition*. Informs journal on computing, Vol. 12(1), pp.2-23.

Katoch, S., Chauhan, S.S. and Kumar, V. (2021). *A review on genetic algorithm: past, present, and future.* Multimedia tools and applications, Vol. 80, pp.8091-8126.

Kovacs, A. A., Parragh, S. N., Doerner, K. F. and Hartl, R. F. (2012). *Adaptive large neighborhood search for service technician routing and scheduling problems*. Journal of scheduling, Vol. 15, pp.579-600.

Kumar, S.N. and Panneerselvam, R. (2015). *A time-dependent vehicle routing problem with time windows for e-commerce supplier site pickups using genetic algorithm*. Intelligent Information Management, Vol. 7(4), pp.181-194.

Lim, A., Rodrigues, B. and Song, L. (2004). *Manpower allocation with time windows.* Journal of the Operational Research Society, Vol. 55(11), pp.1178-1186.

Li, Y., Lim, A. and Rodrigues, B. (2005). *Manpower allocation with time windows and job-teaming constraints*. Naval Research Logistics (NRL), Vol. 52(4), pp.302-311.

Papadimitriou, C. H. (1977). *The Euclidean travelling salesman problem is NP-complete.* Theoretical computer science, Vol. 4(3), pp.237-244.

Raff, S. (1983). *Routing and scheduling of vehicles and crews: The state of the art*. Computers & Operations Research, Vol. 10(2), pp.63-211.

Rasmussen, M. S., Justesen, T., Dohn, A. and Larsen, J. (2012). *The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies*. European journal of operational research, Vol. 219(3), pp.598-610.

Solomon, M. M. (1987). *Algorithms for the vehicle routing and scheduling problems with time window constraints.* Operations research, Vol. 35(2), pp.254-265.

Solomon, M. M. and Desrosiers, J. (1988). *Survey paper—time window constrained routing and scheduling problems.* Transportation science, Vol. 22(1), pp.1-13.

Xie, F., Potts, C.N. and Bektaş, T. (2017). *Iterated local search for workforce scheduling and routing problems*. Journal of Heuristics, Vol. 23, pp.471-500.

Xu, J. and Chiu, S. Y. (2001). *Effective heuristic procedures for a field technician scheduling problem.* Journal of Heuristics, Vol. 7(5), pp.495-509.

Zhang, Y. and Li, J. (2024). *A Hybrid Heuristic Harmony Search Algorithm for the Vehicle Routing Problem With Time Windows*. IEEE Access, Vol. 12, pp.42083-42095.