Constraint Programming Approaches to Electric Vehicle and Robot Routing Problems

by

Kyle Booth

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Department of Mechanical & Industrial Engineering
University of Toronto

Abstract

Constraint Programming Approaches to Electric Vehicle and Robot Routing Problems

Kyle Booth
Doctor of Philosophy
Department of Mechanical & Industrial Engineering
University of Toronto
2021

Driven by global efforts to curb greenhouse gas emissions, there has been significant investment in electric vehicle (EV) technology in recent years, resulting in a substantial increase in EV market share. Concurrently, the demand for mobile robots, such as unmanned aerial vehicles (UAVs) and land-based robots, has also experienced rapid growth, encouraged by recent advances in the autonomy and capabilities of these systems. Common to both of these technologies is the use of electric motors for propulsion and batteries for mobile energy storage. Techniques for the coordination of electric vehicle fleets, whether human-operated or autonomous, must address a variety of unique challenges, including sparse recharging infrastructure, significant recharge durations, and limited battery capacities.

The central thesis of this dissertation is that constraint programming (CP) can be an effective and flexible paradigm for modeling and solving routing problems involving electric vehicles. While efforts on the development of mixed-integer linear programming (MILP) approaches to electric vehicle routing problems within the vehicle routing literature are expansive and consistently growing, the exploration of CP for approaching these increasingly pervasive problems has been, thus far, limited.

Throughout this dissertation, we address this thesis by developing novel CP formulations for both existing and new electric vehicle routing problem domains, leveraging modeling strategies from the MILP literature. Specifically, we propose novel CP models based on recharging path multigraphs as well as introduce a novel single resource modeling strategy that exploits symmetry in vehicle fleets for CP models involving interval, sequence, and cumulative function expression variables. These methodological contributions are then applied to four problem domains, where applicable. The first two problem domains involve electric vehicle routing with traditional logistics fleets. The third domain requires the synchronization of heterogeneous fleets involving UAVs and ground-based vehicles in a large-scale surveillance setting. Finally, the last problem domain involves the routing of social robots in a retirement home setting under complex real-world constraints, including synchronization and optional task precedence. For each of these domains, we empirically demonstrate the effectiveness of our CP approaches.

Acknowledgements

First and foremost, I would like to thank my supervisor, Chris Beck, for his guidance and support over the past six years. Thank you for introducing me to the wonderful worlds of scheduling and mathematical optimization, and for providing me with the opportunity to conduct research in these exciting fields. The valuable lessons I have learned under your tutelage will remain with me throughout my career.

Thank you to my internal committee members, Goldie Nejat, Timothy Chan, and Eric Diller, for their time and helpful feedback throughout my graduate studies. Thank you to the members of my final committee, Andre Cire and Peter van Beek, for their valuable comments during the final preparation of this dissertation and during my defense.

Thank you to the National Sciences and Engineering Research Council of Canada and the Ontario Graduate Scholarship program for providing funding for my research.

Thank you to my UTORG and TORCH friends; I am so proud of what we were able to accomplish together. Thank you to my colleagues at NASA Ames for introducing me to the wild world of quantum computing, and to my colleagues at the University of Waterloo for giving me the opportunity to teach scheduling within the Department of Management Sciences, an experience that I will always cherish.

I would like to express my sincerest thanks to my TIDEL friends: Tony, Margarita, Chang, Chiara, Eldan, Wen-Yang, Buser, Arik, Michael, Tanya, Filip, Stefana, Ranjith, and Luke. Thank you for the numerous discussions, arguments, jokes, and coffee breaks, and for making the lab feel like home.

Thank you to my friends, Paul, Steve, and Ryan, for always keeping me grounded, and to my friend Wale for painstakingly introducing me to operations research and decision variables.

I would like to express a special thank you to my family. To my parents, Judy and Fred, for providing me with an unshakeable foundation of support from which I feel like I can accomplish anything; this is an invaluable gift. To my brother, Adam, for his love and mutual interest in sports highlights, and to my creature friends Lapa, Ella, and Chance, for all of the days that they made brighter with their presence.

Finally, thank you to my best friend and wife, Meredith. This dissertation would not have been possible without your unwavering love, patience, and support. I am extremely lucky to have you in my life.

Dedicated to my parents.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Electric vehicles (EVs) are vehicles that use electric motors to propel themselves. The development of EVs has a long history, with initial models introduced in the late 1800s and the technology holding a market share advantage over combustion vehicles until roughly 1930 [189]. Demand for electric vehicles stagnated during the mid-1900s, hampered by higher development cost and poorer performance than their combustion-based counterparts. In recent years, however, EV market share and investment has risen considerably as a result of concerted global efforts by governments to curb greenhouse gas emissions [213]. In addition to not locally producing greenhouse gas emissions, EVs are attractive as they are less noisy than combustion engines and they can be powered from renewable energy sources [59]. The advent of EVs has propelled vehicle routing research to incorporate the unique objectives and constraints associated with their operation. Specifically, route-planning for EV fleets must consider sparser recharging networks, longer recharge times, and limited battery capacity [59].

Beyond the electrification of traditional vehicles, the development of battery-powered mobile robots has also seen significant recent growth, driven by the availability of relatively inexpensive and reliable robots [6]. The increased capabilities and autonomy of these systems have allowed them to substitute for humans in many fields including surveillance, transportation, and medical care [181]. Route planning for battery-powered mobile robots, such as unmanned aerial vehicles (UAVs) or ground-based systems, involves many of the same constraints as EV routing in traditional transportation systems, including limited recharging infrastructure, limited vehicle range, and extended recharge durations. As a result, models and methods developed for EV route-planning in traditional contexts are relevant for the coordination and planning of less conventional electric vehicle fleets. Indeed, the class of problems related to electric routing problems is becoming increasingly pervasive as technologies migrate to electric solutions.

The route planning of vehicle fleets using mathematical modeling and optimization has been an actively researched area since the 1930s [44]. The goal is to find a set of routes for a fleet of vehicles that optimizes some underlying objective function (e.g., minimizing fleet distance traveled) while subject to problem-specific constraints (e.g., each customer package must be delivered). Broadly, these efforts can be partitioned into two main categories: i) heuristic methods and ii) exact methods. A heuristic approach, given the particulars of a route planning problem and a run-time budget, tries to find the best possible solution while providing no guarantees on the quality of that solution. These techniques prioritize rapid feasibility and finding solutions of sufficient quality over provable optimality. An exact approach, however, is guaranteed to find the (provably) best solution to the problem given enough time.

Since these problems are often NP-Hard [139], exact approaches leverage increasingly intelligent forms of enumerative search, such as branch-and-bound [132].

The development of state-of-the-art exact approaches often involves the integration of numerous subroutines and sophisticated techniques for exploiting problem structure. As an example, the Concorde *traveling salesman problem* (TSP) solver maintains state-of-the-art performance for finding optimal TSP solutions, the largest solved (so far) having 85,900 cities [5]. Concorde integrates over 700 functions and thousands of lines of code to solve standard, symmetric instances of TSP. Minor variations in the problem definition, such as the inclusion of time windows, preclude the problem from being solved with this highly specialized software. While not to the same degree, other exact approaches, such as sophisticated branch-and-price-and-cut methods, suffer from similar complexity and inflexibility [62, 52].

In an effort to improve the accessibility of exact approaches and broaden the set of problems that can be solved with them, researchers have made ongoing contributions to general-purpose solver technology. These solvers take, as input, a model of a problem expressed by the practitioner and then invoke a series of general-purpose routines in a systematic tree search to solve the model. Using this framework, users can benefit from ongoing advances in the engine of the solver. Commercial mixed-integer linear programming (MILP) solvers, for example, have exhibited a machine-independent (i.e., algorithmic only) speed-up of over 400,000 times from the early 1990s to 2012 [22]. Additionally, models can be easily altered to incorporate new side constraints as problem characteristics change, providing a level of simplicity and flexibility not present in more adhoc or specialized algorithms.

While the development of general purpose solvers may seem to create a clean partition between modeling a problem and solving it, realizing the performance potential of these tools requires a keen understanding of both parts. For a given problem, there exist a multitude of formulations that are, in terms of the problem definition, mathematically equivalent. In practice, due to the systematic tree search algorithms used, these different formulations can have enormous impact on solver performance. In constraint programming (CP), for example, the selection of one constraint over another can significantly effect the inference that can be performed at a node in the search. Similarly, in MILP, one formulation may have a tighter LP relaxation than another. Indeed, there is a large body of literature concerned with determining which formulations perform better than others, including empirical studies in scheduling [125], vehicle routing [76], and facility location [78]. Due to the nature of tree search algorithms and the hardness of the problems being solved, the comparison of formulations is usually done computationally, though they can also be compared using metrics such as model size (e.g., number of variables and constraints) and linear relaxation strength, among others.

In this dissertation, we investigate the development of novel CP approaches for routing problems involving electric vehicles. At the time of writing, CP has not been explored as an alternative to MILP for modeling and solving these problems. Given the success of CP technology for other families of combinatorial problems in routing and scheduling [127], we investigate whether this paradigm can play a key role in the growing area of electric vehicle routing.

Thesis Statement. *The central thesis of this dissertation is that constraint programming can be an effective and flexible paradigm for modeling and solving routing problems involving electric vehicles. Leveraging modeling strategies from mixed-integer linear programming, we can formulate constraint programming models that, when coupled with general-purpose solver technology, provide high-quality solutions to instances of this important and increasingly pervasive class of problems.*

1.1 Approach

This dissertation adopts an engineering-style, application-driven approach to the investigation of our thesis. Specifically, we explore the development and application of CP and MILP approaches to four different problems. In Chapter 4 we begin with two relatively ‘pure’ electric vehicle routing problems (i.e., those with few side constraints) involving more traditional logistics fleets, where CP-based modeling strategies are developed using techniques from their MILP analogs. The problems investigated in subsequent chapters become progressively richer and our modeling efforts demonstrate the flexibility of the CP paradigm. Chapter 5 introduces a problem involving the synchronization of heterogeneous fleet types with ground-based and aerial vehicles, and Chapter 6 a problem involving synchronization constraints, partial vehicle recharges, optional task precedence, and a swath of other side constraints.

In the context of each application, our research approach consists of two primary stages: model development and empirical study. In general, model development is the most time consuming process, and efforts may cycle back-and-forth between empirical evaluation and model design as decisions are informed by observed empirical performance. For this reason, model development is both a mathematical and programming exercise. Indeed, the specific constraint solver software used is established prior to the model development phase as, particularly for CP, the software itself identifies the language of constraints that can be used. We provide an overview of the model development and empirical study stages as follows.

Model Development. In this dissertation, we make efforts to develop our formulations in a consistent, disciplined manner, aiming to produce memory-efficient models that result in strong solver performance. Given a problem domain, the first step of our model development approach is to identify key problem characteristics that can significantly impact the design of our formulations (e.g., the presence of vehicle fleet symmetry, a characteristic that can be exploited during modeling). Next, depending on the problem characteristics identified, we determine the most appropriate mathematical representations for the core aspects of the problem. While vehicle routing problems are typically defined and represented on graphs, certain problem traits can preclude the effective use of specific graph types (e.g., simple graphs or multigraphs). Finally, for each of the appropriate representations identified, we implement a set of candidate formulations in both CP and MILP, noting key model metrics such as the number of variables and constraints in each. The performance of these models is then evaluated in the empirical study.

Empirical Study. With our developed formulations in hand, we adopt an empirical approach for their evaluation, consisting of an extensive set of computational experiments. First, a benchmark set of instances for which the proposed formulations will be applied to is identified. This benchmark can be either from the existing literature, generated in this work, or some combination of the two. Next, we select the metrics that will be recorded during our experiments. In this dissertation, we are primarily concerned with metrics associated with the solver's ability to find feasible solutions and prove optimality for a given formulation and problem instance. We also report the specific quality of returned solutions and bounds on the objective function that can be used to compute optimality gaps. Finally, our computational experiments consist of running our implemented formulations with the solver, and recording the selected performance criteria. Often, this criteria is recorded in a time-series fashion to provide a better understanding of the evolution of performance over time. In this case, the application drives the selection of appropriate run-time limits. The results are then gathered, aggregated and presented in tabular or graphical format to facilitate comparisons and derive insight.

1.2 Dissertation Overview

The chapters of this dissertation support our thesis in various ways. Chapter 2 provides background information and Chapter 3 presents a review of the relevant literature. The next three chapters (Chapters 4, 5, and 6) form the main content of this dissertation, within which we investigate a variety of electric routing problems and develop novel formulations for them. Finally, Chapter 7 provides conclusions and indicates directions for future work. We provide a high-level summary of each of the chapters as follows.

Chapter 2 provides the reader with the MILP and CP background required to support an understanding of the methods used in this dissertation. For each of these paradigms, we summarize how problems, broadly, are modeled and solved. First, we detail the acceptable forms of variables and constraints in the respective modeling formalisms. We then identify, with examples, a generic implementation of the tree search used to solve problems in each method with accompanying references to both commercial and open source solvers.

Chapter 3 reviews the literature relevant to this dissertation. The survey provides a brief history of the field of vehicle routing before narrowing in on routing efforts within MILP for problems involving electric vehicles. We structure the literature according to the composition of vehicle fleets (homogeneous vs. heterogeneous) and the mobility of recharge stations (static vs. mobile), characteristics that are relevant to the problems investigated in the main chapters of this dissertation. The review then summarizes relevant work within CP, extending our scope to vehicle routing broadly as the contributions of this dissertation are among the first investigations into CP for routing electric vehicle fleets.

Chapter 4 proposes the first CP formulations for the *electric vehicle routing problem with time windows* (EVRPTW) and the *pickup and delivery problem with time windows and electric vehicles* (PDPTW-EV). Following work in the MILP literature, we present CP models based on both augmented graph and recharging path multigraph problem representations. For each of these representations, we detail an alternative resource CP model, as well as a model that leverages a novel single resource transformation to exploit the symmetry of the vehicle fleet. We conduct an empirical analysis of our formulations and compare them to MILP models from the literature. The work in this chapter extends our previously published research in the *Proceedings of the Sixteenth International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research* [23].

Chapter 5 investigates the viability of range-constrained, battery-powered commercial UAVs for a multi-UAV target search problem over real-world road networks. The operation range of the UAVs is bolstered via the inclusion of a fleet of ground-based mobile recharging vehicles (MRVs) that can meet up with and recharge UAVs. Starting from real-world road network data, we propose a pipeline to represent the problem as a routing graph through a series of discretization steps. The formulations from Chapter 4 are then adapted to model and solve the mixed-fleet routing problem with synchronization constraints using both MILP and CP. We utilize the proposed optimization formulations to gauge the impact of including the MRVs in the fleet. Finally, an empirical investigation of our methods using real-world road network data from Scotland is conducted. Our findings suggest that the inclusion of MRVs can make commercially-available UAVs viable for target search over real-world road networks. The work in this chapter is based on our research published in *IEEE Robotics & Automation Letters [26].*

Chapter 6 explores the *social robot routing problem in retirement homes* (SRRP-RH), a complex, real-world multi-robot task allocation and scheduling problem that takes place in a retirement home setting. The problem involves a large number of complex side constraints, including vehicle synchronization and optional task precedence. We adapt and apply both MILP and CP augmented graph routing formulations to the SRRP-RH, leveraging modeling strategies developed in Chapters 4 and 5. We compare our formulations to those previously proposed in the literature with favorable results, and demonstrate that our proposed models capture key problem characteristics, such as user travel in the environment, that existing models do not. The research in this chapter significantly extends our work published in the *Proceedings of the Twenty-Second International Conference on Principles and Practice of Constraint Programming* [25] and represents one of the few works that link the fields of electric vehicle routing and multi-robot task allocation.

Finally, Chapter 7 concludes the dissertation and provides directions for future research.

1.3 Summary of Contributions

The contributions of this dissertation are centered on the development of new CP and MILP formulations, with emphasis on the former, for both existing and novel applications involving the routing of battery-powered vehicles. Chapter 4 includes a traditional performance evaluation of the presented formulations. Chapters 5 and 6, in addition to traditional performance evaluations, use the developed optimization models to derive insights into the real-world problems to which they are applied. For each chapter we summarize the main contributions in terms of mathematical modeling and empirical analysis.

Chapter 4: Constraint Programming for Vehicle Routing and Pickup and Delivery Problems with Electric Vehicles

1. We propose the first CP formulations for the *electric vehicle routing problem with time windows* (EVRPTW) and the *pickup and delivery problem with time windows and electric vehicles* (PDPTW-EV). Following modeling techniques from the MILP literature, our CP models are based on both augmented graph and recharging path multigraph problem representations. To our knowledge, our work is the first to propose multigraph-based CP models for vehicle routing problems.

2. For both the augmented graph and multigraph representations, we introduce a novel single resource transformation for modeling routing problems involving homogeneous vehicle fleets using optional interval, sequence, and cumulative function expression variables. Exploiting vehicle symmetry, the single resource formulation uses significantly fewer variables and constraints than the alternative resource modeling approach and, for large problems, requires orders of magnitude less memory. Our single resource transformation can be applied to other problems in scheduling and routing with homogeneous resources. In its entirety, this chapter proposes four CP models for each problem, for a total of eight novel formulations.

3. We conduct an extensive empirical analysis of our CP formulations against MILP formulations from the literature. For the augmented graph formulations of the EVRPTW, we demonstrate that our single resource CP approach outperforms the alternative resource CP approach on all experiment classes and outperforms the augmented graph MILP model from the literature for nearly all medium-to-large problem classes. We demonstrate that the single resource CP approach based on recharging path multigraphs for EVRPTW is competitive with a multigraph-based MILP model for fleet minimization problems, specifically for those involving longer planning horizons. Finally, we show that the proposed CP approaches for the PDPTW-EV exhibit strong performance and

are often superior to a published multigraph-based MILP formulation in terms of both feasibility and solution quality.

Chapter 5: Target Search on Road Networks with Range-Constrained UAVs and Ground-based Mobile Recharging Vehicles

1. We propose a novel algorithm for constructing a target search routing graph from real-world road network data, over which our optimization models are formulated. The algorithm consists of a series of discretization steps, including a shortest-path sampling of the underlying road network and the evaluation of a hitting set problem to determine the placement of UAV/MRV recharge rendezvous opportunities.

2. We propose novel MILP and CP models for range-constrained multi-UAV, multi-MRV routing with synchronized recharging rendezvous. The proposed MILP formulation uses a compact two-index augmented graph formulation that leverages the homogeneity of each vehicle type (i.e., UAV and MRV). The CP models, both alternative resource and single resource transformation, extend techniques presented in Chapter 4 to heterogeneous fleets. Synchronization constraints for UAV/MRV recharging rendezvous are formulated in each model. The modeling efforts in this chapter illuminate how mixed-fleets can be synchronized under the single resource transformation modeling strategy.

3. We conduct an empirical analysis which supports the use of our routing graph construction pipeline in small-to-medium sized real-world scenarios, but likely precluding its use for large fleets. In terms of producing synchronized target search plans, we show that the CP-based approaches significantly outperform MILP. Further, while the alternative resource CP model is superior for the real-world instances investigated, the single resource CP model offers attractive performance as problem size scales. We demonstrate that the inclusion of a fleet of MRVs can make commercially-available UAVs viable for the studied application.

Chapter 6: Social Robot Routing in Retirement Homes

1. We classify the SRRP-RH according to taxonomies from both the multi-robot task allocation (MRTA) and vehicle routing problem literature and discuss the complexity of the fleet distance minimization variant of the problem. This work represents, to the best of our knowledge, the first link between the fields of electric vehicle routing and MRTA.

2. Following the augmented graph representation as utilized in Chapters 4 and 5, we propose novel MILP and CP models for the SRRP-RH with a variety of different objective functions. These models include consideration for user travel in the environment, a problem characteristic not addressed by previous efforts. Our MILP formulation leverages vehicle symmetry via a compact formulation to significantly reduce model size compared to a previously proposed model. Further, our CP approach provides more accurate energy consumption modeling than existing efforts.

3. We compare our approaches to those previously published on a relaxed variant of SRRP-RH and demonstrate, through an empirical study, that our methods are often superior with respect to solution feasibility and quality for a variety of objective functions. Furthermore, we propose

a variable ordering heuristic for our augmented graph CP approach that yields the strongest performance across the tested methods, in terms of solution quality, by a considerable margin.

4. We demonstrate, via modeling and experimental analysis, that our proposed approaches are able to capture user travel within the environment, an important aspect of SRRP-RH that helps to ensure that produced routes are achievable in practice.

Chapter 2

Preliminaries

This chapter provides background to support an understanding of the methods used in this dissertation. We provide detailed descriptions of the mixed-integer linear programming (MILP) and constraint programming (CP) paradigms, focusing on modeling and solving problems within these frameworks. For a more comprehensive overview of these subjects beyond the basics covered in this chapter, the author refers the reader to available texts [209, 179].

2.1 Mixed-Integer Linear Programming

Mixed-integer linear programming (MILP) is a mathematical optimization approach for problems modeled as a set of decision variables taking on either continuous or integer values, constrained by linear constraints and with the goal of optimizing a linear objective function. MILP has been used widely for optimization since the 1950s, with notable application to routing and scheduling problems [197, 170].

2.1.1 Modeling

MILPs are used both to model and solve various optimization and feasibility problems.

Definition 2.1.1. *A mixed-integer linear program (MILP) is an optimization problem of the following form:*

$$\min \quad \mathbf{c^T x} \tag{2.1}$$

$$s.t. \quad \mathbf{Ax} \leq \mathbf{b} \tag{2.2}$$

$$\mathbf{x} \geq \mathbf{0} \tag{2.3}$$

$$x_i \in \mathbb{Z} \qquad \forall i \in \mathcal{I} \tag{2.4}$$

where $\mathbf{x}$ *is the vector of* n *decision variables,* $\mathbf{c^T}$ *is a cost vector with* $\mathbf{c} \in \mathbb{R}^n$*,* $\mathbf{A} \in \mathbb{R}^{m \times n}$ *is a matrix of variable coefficients and* $\mathbf{b} \in \mathbb{R}^m$ *is a column vector. The set* $\mathcal{I}$ *identifies all variables that must be integral in a solution; all other variables can take on continuous values. If* $\mathcal{I} = \emptyset$*, then the program is a linear program (LP) (i.e., all variables are continuous). If* $\dim(\mathbf{x}) = |\mathcal{I}|$*, then the program is called a pure integer linear program (ILP) (i.e., all variables must take on discrete values). If* $x_i \in \{0, 1\}, \forall i \in \mathcal{I}$*, the program is called a 0-1 (binary) linear program.*

Including integer (or binary) variables into the MILP formulation greatly increases the modeling expressiveness of the paradigm, at the expense of complexity. Integer programming is a known $\mathcal{NP}$-complete problem [113], for which no known polynomial-time algorithm exists. For a given mathematical problem definition, there is often more than one way to model the problem using MILP, particularly for large, complex problems.

Due to the way MILPs are solved, some models can be evaluated more efficiently (in practice) than others. The traveling salesman problem (TSP) [5] provides a nice illustration of the importance of modeling decisions in the context of MILP.

Definition 2.1.2. Traveling salesman problem (TSP). *Given a set of cities, $V = \{1, \dots, n\}$, with cost (usually distance) c_{ij} to travel from city $i \in V$ to $j \in V$, we seek a route that starts at a city, visits each other city exactly once, and returns to where it starts. The objective is to minimize the total travel cost.*

In the literature, the most commonly employed MILP models for the TSP are the Dantzig, Fulkerson, and Johnson formulation [46], known as the DFJ formulation, and the Miller-Tucker-Zemlin formulation [155], known as the MTZ or compact formulation.

Definition 2.1.3. DFJ TSP formulation. *The DFJ formulation uses a single binary decision variable, x_{ij}, that is 1 if the solution visits city j immediately after i and 0 otherwise. The model is expressed as follows:*

$$\min \quad \sum_{i \in V} \sum_{j \in V, i \neq j} x_{ij} c_{ij} \tag{2.5}$$

$$\sum_{i \in V, i \neq j} x_{ij} = 1 \qquad \forall j \in V \tag{2.6}$$

$$\sum_{j \in V, i \neq j} x_{ij} = 1 \qquad \forall i \in V \tag{2.7}$$

$$\sum_{i \in S} \sum_{j \in S, i \neq j} x_{ij} \leq |S| - 1 \qquad \forall S \subset V, |S| \geq 2 \tag{2.8}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in V, j \in V, i \neq j \tag{2.9}$$

Objective (2.5) *minimizes total travel cost. Constraints* (2.6) *and* (2.7) *ensure each city has exactly one predecessor and successor (these are called "degree" constraints). Constraint* (2.8) *enforces no subtours in the solution. Given a set of cities, $S \subset V$, a subtour is a sequence of visits that starts at a city, $i \in S$, visits each other city in S exactly once and returns to i without visiting any cities outside of S. Constraint* (2.9) *identifies the binary domain of the decision variable.*

In the DFJ model, Constraint (2.8) must be included for each subset $S \subset V : |S| > 2$. The number of these constraints is exponential in the number of cities. As such, it is difficult to implement this model without resorting to more sophisticated solution procedures such as branch-and-cut [66], though these more involved implementations can be quite efficient. It is also difficult to incorporate richer problem characteristics, such as time windows, into this formulation. As a result, a more compact and flexible formulation was presented for modeling and solving TSPs.

Definition 2.1.4. MTZ ("compact") TSP formulation. *The MTZ formulation uses binary variables*

x_{ij}, defined as in the DFJ model, and continuous labeling variables u_i, used to sequence the visits.

$$\min \sum_{i \in V} \sum_{j \in V, i \neq j} x_{ij} c_{ij} \tag{2.10}$$

$$\sum_{i \in V, i \neq j} x_{ij} = 1 \qquad \forall j \in V \tag{2.11}$$

$$\sum_{j \in V, i \neq j} x_{ij} = 1 \qquad \forall i \in V \tag{2.12}$$

$$u_i + 1 - n(1 - x_{ij}) \leq u_j \qquad \forall i, j \in \{2, \dots, n\}, i \neq j \tag{2.13}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in V, j \in V, i \neq j \tag{2.14}$$

$$u_1 = 0 \tag{2.15}$$

$$1 \leq u_i \leq n - 1 \qquad \forall i \in \{2, \dots, n\} \tag{2.16}$$

Objective (2.10) *minimizes total travel cost. Constraints* (2.11) *and* (2.12) *are the degree constraints. Constraint* (2.13) *ensures the visits are properly sequenced. Constraints* (2.14) *through* (2.16) *identify variable domains.*

The MTZ model uses auxiliary labeling variables and disjunctive constraints to eliminate subtours. While the formulation includes n more variables than the DFJ formulation, the number of constraints required grow quadratically with the number of cities, allowing this formulation to be implemented in solvers for problems with significantly more cities. The MTZ formulation is also flexible in that it can incorporate other problem characteristics such as time windows and precedence constraints that are common in routing applications. The primary drawback, however, is that it suffers from a weak linear relaxation, and thus weak tree search bounds, often resulting in poor solver performance. Given these various tradeoffs between model size, flexibility, and performance, modeling is a large area of focus for the operations research and mathematical programming communities [68, 125, 134].

2.1.2 Solving

The standard approach for solving MILPs is branch-and-bound tree search [132]. The key to often avoiding the worst-case exponential search is the fact that disregarding the integrality requirement on the integer variables results in a linear programming (LP) relaxation of the MILP that can be solved in polynomial time. The LP is solved at each node in the branch-and-bound tree, yielding a bound on the objective function and, once a feasible solution is found, the ability to prune sub-trees that are proven not to contain solutions better than the current incumbent. The solution to the LP relaxation is also used heuristically to determine the order in which sub-trees are explored.

Modern MILP solvers combine branch-and-bound search with techniques such as cutting planes in a branch-and-cut framework [164] in order to solve large, complex problems. Along with a description of the historical development of MILP solving, Bixby [22] shows a machine-independent (i.e., algorithmic only) speed-up of over 400,000 times in commercial MILP solvers from the early 1990s to 2012. While we do not provide details of these advanced MILP solving techniques, we identify the core processes behind MILP branch-and-bound in the following sections.

The branch-and-bound tree search used to evaluate MILPs consists of two primary components: inference and branching. Inference is accomplished by solving the LP relaxation and bounding the tree based on its objective value, while branching partitions the search space by introducing additional inequalities in the program.

2.1.2.1 Inference

Evaluating the LP relaxation allows inferences to be made during the tree search that can rule out exploring portions of the tree.

Definition 2.1.5. *The LP relaxation of the MILP presented in Eqns.* (2.1) *through* (2.3) *is defined as follows:*

$$\min \quad \mathbf{c^T x} \tag{2.17}$$

$$s.t. \quad \mathbf{Ax} \leq \mathbf{b} \tag{2.18}$$

$$\mathbf{x} \geq \mathbf{0} \tag{2.19}$$

The program is identical to the MILP, except Constraint (2.4) *is omitted, effectively relaxing the integrality condition. We note that Constraints* (2.18) *can be converted to form* $\mathbf{Ax} = \mathbf{b}$*, so-called 'standard form', via the use of additional slack variables.*

The optimal solution to the LP relaxation provides a bound on the objective function of the MILP that can then be used to infer which areas of the search space need to be further explored, and which can be pruned. There are a number of algorithms used to evaluate LPs, the most popular of which are the simplex method [45] and the interior point method [112]. While the simplex algorithm has exponential worst-case complexity, the interior point method has been shown to run in polynomial time. In terms of empirical complexity, these algorithms have been shown to be competitive for routine applications of linear programming [152].

2.1.2.2 Branching

Branching is the mechanism used to generate new subproblems to explore in a 'divide and conquer' fashion. The standard branching strategy first looks to identify integer variables that, in the optimal solution to the LP relaxation at the current node, take on fractional values. Identifying a specific variable to branch on is known as *variable selection*, and there exist numerous heuristics used for the variable selection problem [2].

Given a variable with fractional value in the solution to the LP relaxation, $x_i = \hat{x}_i$, two children are generated from the current node using the constraints $x_i \leq \lfloor \hat{x}_i \rfloor$ and $x_i \geq \lceil \hat{x}_i \rceil$. Each child consists of the objective and set of constraints from its parent node, in addition to the noted branching constraint. Determining which child to investigate next is known as *node selection*, and there exist a variety of heuristics used for the node selection problem as well [2].

2.1.2.3 Complete Algorithm

The complete MILP branch and bound algorithm involves iterating between solving LP relaxations and branching on integral variables with fractional values. The process can be summarized as follows (assuming a minimization objective):

1. Given an MILP, initialize an upper bound, $UB = \infty$, an incumbent solution, $\mathbf{x}^* = \emptyset$, and a subproblem set, $\mathcal{S} = \{LP^{(0)}\}$, where $LP^{(0)}$ is the root node LP relaxation. Set iteration counter $i = 0$.

2. Repeat until $\mathcal{S} = \emptyset$:

   (a) Select a subproblem, $LP^{(j)}$, with index j, from set $\mathcal{S}$ and solve to optimality (e.g., using the simplex method).

   (b) If infeasible or $LP^{(j)}$ objective value $z^{(j)} \geq UB$, discard subproblem, $\mathcal{S} \leftarrow \mathcal{S} \setminus \{LP^{(j)}\}$, and return to 2.

   (c) Else if $LP^{(j)}$ solution $\mathbf{x}^{(j)}$ is feasible w.r.t. the original MILP set $UB = z^{(j)}$ and $\mathbf{x}^* = \mathbf{x}^{(j)}$. Discard subproblem $LP^{(j)}$ and return to 2.

   (d) Else, pick a fractional variable that violates MILP integrality, $x \in \mathbf{x}^{(j)}$, generate children $LP^{(i+1)} := LP^{(j)} \cup \{x \leq \lfloor \hat{x} \rfloor\}$ and $LP^{(i+2)} := LP^{(j)} \cup \{x \geq \lceil \hat{x} \rceil\}$ and add them to subproblem set, $\mathcal{S} \cup \{LP^{(i+1)}, LP^{(i+2)}\}$. Update iteration counter $i = i + 2$. Discard subproblem $LP^{(j)}$. Return to 2.

While these steps capture the essence of MILP branch and bound search, more sophisticated schemes use local lower bounds at various nodes to conduct further pruning. Upon termination of the algorithm, incumbent $\mathbf{x}^*$ will store the optimal variable values with associated objective function value UB. Note that we could pose a similar set of steps for a maximization problem.

2.1.2.4 Example

To demonstrate a standard implementation of MILP branch-and-bound, consider the 0-1 knapsack problem, modeled and solved using MILP.

Definition 2.1.6. *0-1 knapsack problem. Given a knapsack with capacity C, a set of items, $i \in N$, with weights, w_i, and values, v_i, the 0-1 knapsack problem looks to select items to place in the knapsack such that their total value is maximized and capacity constraints are satisfied. The problem can be modeled as an MILP as follows:*

$$\max \quad \sum_{i \in N} v_i x_i \tag{2.20}$$

$$s.t. \quad \sum_{i \in N} w_i x_i \leq C \tag{2.21}$$

$$x_i \in \{0, 1\} \qquad \forall i \in N \tag{2.22}$$

where $x_i = 1$ if item i is selected and 0 otherwise. Objective (2.20) *maximizes the total value of items selected, Constraint* (2.21) *ensures capacity is not violated, and Constraint* (2.22) *enforces integrality on the decision variables.*

With this MILP model, we conduct branch-and-bound search for a specific instance of the problem.

Example 2.1.1. *Consider an instance of the 0-1 knapsack problem with a knapsack capacity of 15, and five items with properties illustrated in Table 2.1.*

Item	1	2	3	4	5
Weight, w_i	12	1	4	1	2
Value, v_i	4	2	10	1	2

Table 2.1: 0-1 knapsack problem example instance.

The problem instance can be modeled as the MILP: $\{\max\ 4x_1 + 2x_2 + 10x_3 + 2x_4 + x_5 \mid 12x_1 + x_2 + 4x_3 + 2x_4 + x_5 \leq 15, x_i \in \{0,1\}, \forall i \in \{1,2,\ldots,5\}\}$*. The program can then be solved with branch and bound as follows:*

Step 1: *Evaluate the root node LP relaxation:* $LP^{(0)} := \{\max\ 4x_1 + 2x_2 + 10x_3 + 2x_4 + x_5 \mid 12x_1 + x_2 + 4x_3 + 2x_4 + x_5 \leq 15, 0 \leq \mathbf{x} \leq \mathbf{1}\}$*. This yields solution* $\mathbf{x}^{(0)} = (0.5833, 1, 1, 1, 1)$ *with cost* $z^{(0)} = 17.33$*. Update upper bound,* $UB = z^{(0)}$*.*

Step 2: *Branch on fractional variable* x_1 *by introducing the constraints* $x_1 \leq 0$*, resulting in child* $LP^{(1)}$*, and* $x_1 \geq 1$*, resulting in child* $LP^{(2)}$*.*

Step 3: *Evaluate child:* $LP^{(1)} := \{\max\ 4x_1 + 2x_2 + 10x_3 + 2x_4 + x_5 \mid 12x_1 + x_2 + 4x_3 + 2x_4 + x_5 \leq 15, x_1 \leq 0, 0 \leq \mathbf{x} \leq \mathbf{1}\}$*. This yields solution* $\mathbf{x}^{(1)} = (0, 1, 1, 1, 1)$ *with cost* $z^{(1)} = 15$*. Since all variable values satisfy integrality, we store solution as incumbent,* $\mathbf{x}^* = \mathbf{x}^{(1)}$ *with cost* $z^* = z^{(1)}$*. No children to generate.*

Step 4: *Evaluate child:* $LP^{(2)} := \{\max\ 4x_1 + 2x_2 + 10x_3 + 2x_4 + x_5 \mid 12x_1 + x_2 + 4x_3 + 2x_4 + x_5 \leq 15, x_1 \geq 1, 0 \leq \mathbf{x} \leq \mathbf{1}\}$*. This yields solution* $\mathbf{x}^{(2)} = (1, 0, 0.75, 0, 0)$ *with cost* $z^{(2)} = 11.5$*. Since* $z^{(2)} = 11.5 < 15 = z^*$*, the children of this node cannot produce a better solution than our current incumbent. We prune the node by optimality and return* $\mathbf{x}^*$ *as the optimal solution with cost* z^**.*

A depiction of the branch and bound tree search is given in Figure 2.1.

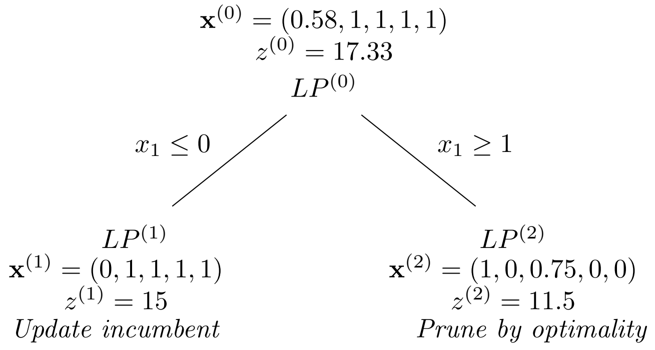


Figure 2.1: 0-1 knapsack branch and bound tree example.

In the example, the problem was solved to proven optimality by solving three LP relaxations. Note that if our node ordering strategy had instead investigated child $LP^{(2)}$ first, the resulting tree may have been different.

2.1.3 Software

There exist a number of commercial and open source software packages for both modeling and solving LPs and MILPs. The most popular and powerful commercial solvers include IBM ILOG CPLEX, Gurobi, and FICO Express Optimizer [104]. Notable open source solvers include the COIN-OR Branch & Cut (CBC) solver [69], the GNU Linear Programming Kit (GLPK), and SCIP [1]. Finally, there are a number of high-level modeling packages intended to allow efficient mathematical modeling with compatibility to a variety of solvers. Such packages include AMPL [70], PuLP [156], Pyomo [93], as well as many others.

2.2 Constraint Programming

Constraint programming (CP) is more general than MILP, allowing variable types beyond integer and continuous (e.g., interval [127] and set variables [80]), and dropping the restriction of linearity in the constraints and objective function. Problem modeling in CP focuses on combining global constraints [204] that encapsulate frequently recurring combinatorial sub-structure. Developed primarily within the artificial intelligence community, CP has also been applied to a wide range of combinatorial optimization problems with routing and scheduling being some of the most successful commercial areas [10]. In this section we present necessary background, including the notion of constraint satisfaction problems (CSPs), consistency, global constraints, filtering algorithms, and branch-and-infer tree search, all fundamental concepts in CP.

Unlike MILP, there is no standardized notation for describing CP formulations, as the description of models is strongly dependent on the specific solver and language they are implemented in. Further, the availability of certain variable types and constraints typically varies depending on the software used. In this dissertation, we formulate our models and follow the notation conventions used in previous work, with particular emphasis on the keywords, syntax and modeling objects used in the IBM ILOG CP Optimizer solver [203, 179, 127].

2.2.1 Modeling

CPs are modeled as either CSPs or constraint optimization problems (COPs).

Definition 2.2.1. Constraint satisfaction problem (CSP). *A CSP consists of a set of decision variables, $\mathcal{X} = \{x_1, \dots, x_n\}$, with domains given by $\mathcal{D} = \{D_1, \dots, D_n\}$, and a set of constraints, $\mathcal{C} = \{C_1, \dots, C_k\}$. The domain of a variable is the set of values to which the variable can possibly be assigned. Each constraint, $C \in \mathcal{C}$, acts on a subset of $\mathcal{X}$, known as the* scope *of the constraint. A solution to a CSP is an assignment of values to variables, from the domains of those variables, such that each constraint is satisfied.*

Definition 2.2.2. Constraint optimization problem (COP). *A COP is a CSP with an objective function. The optimal solution to a COP is an assignment of values to variables, from the domains of those variables, such that each constraint is satisfied and the objective function is optimized (either maximized or minimized).*

2.2.1.1 Decision Variables

In this section we identify some of the rich variable types that the CP paradigm allows, in addition to the standard continuous, binary, and integer variable types. While we focus on the variable types used in the modeling in this dissertation, we note that this is not a complete list.

Optional interval variables. Formally, optional interval variables are decision variables whose possible values are convex intervals: $\{\bot\} \cup \{[a, b) \mid a, b \in \mathbb{Z}, a \leq b\}$, where a and b are the start and end values of the interval and $\bot$ is a special value indicating the variable is not present in the solution [127]. The presence (binary), start, and length of an optional interval variable, *var*, can be expressed within a CP model using $\textsc{Pres}(var)$, $\textsc{Start}(var)$, and $\textsc{Length}(var)$, respectively. We use the notation $\textsc{intervalVar}([d_1, d_2], [s_1, s_2])$ to define these variables in our models (and $\textsc{optIntervalVar}$ if the interval variable is optional), where d_1 and d_2 dictate bounds on the length of the interval variable (if $d_1 = d_2$, the interval variable is fixed length) and where s_1 and s_2 identify the permissible start time window of the interval variable (if $s_1 = s_2$, the interval variable can only start at one specific time point). Model constraints are only enforced over present interval variables Optional interval variables are useful for modeling routing or scheduling problems. An optional interval variable is visualized in Figure 2.2.

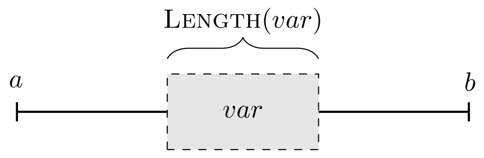


Figure 2.2: Optional interval variable. Dashed line indicates optionality of the variable (present or absent). Solid lines indicate the time window for task, $[a, b)$. Curly bracket indicates task length, $\textsc{Length}(var)$, which may itself be a variable.

Sequence variables. A sequence variable, $\pi = \textsc{sequenceVar}(A)$, defined on a set of optional interval variables, $A = \{var_1, \ldots, var_n\}$, is a rich decision variable type whose possible values are all of the permutations (i.e., orderings) of the present interval variables in A [127]. For example, if $A = \{var_1, var_2\}$, where $\textsc{Pres}(var_1) = 1$ and $\textsc{Pres}(var_2) \in \{0, 1\}$, then the domain of π consists of permutations: $\{(var_1), (var_1, var_2), (var_2, var_1)\}$. Sequence variables are useful for expressing model constraints over a permutation of optional interval variables. Given a sequence variable, π, various constraints can be expressed, including those on the interval variable previous to *var* in the sequence, $\textsc{Prev}_\pi(var)$, and temporal constraints such as the $\textsc{NoOverlap}(\pi)$ constraint (see Section 2.2.1.2), which ensures the interval variables in the sequence do not overlap. A sequence variable only reasons about the present interval variables within its scope. Sequence variables are useful for modeling the relative order of tasks to be executed, whether they be scheduling jobs or routing visits. A sequence variable is visualized in Figure 2.3.

Cumulative function expressions. A cumulative function expression is an expression whose value is the sum of individual contributions of intervals. More formally, a cumulative function expression f is defined as the sum of a set of elementary functions, f_i, such that: $f = \sum_i \epsilon_i \cdot f_i$ where $\epsilon_i \in \{-1, +1\}$. When the values f_i are fixed, f becomes a stepwise integer function [127]. In this dissertation, we use the elementary functions $\textsc{Step}(t, impact)$, which has a height of 0 before t, and *impact* after t,

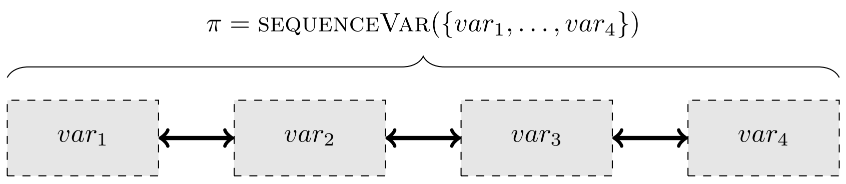


Figure 2.3: Sequence variable. The variable is defined over a set of interval variables. The domain of the variable is the possible permutations of the interval variables, represented by the black arrows.

StepAtStart$(var, impact)$, which has a height of 0 before the start of interval variable var, and $impact$ after the start of var, and Pulse$(var, impact)$, which has a height of 0 before and after var, and $impact$ during var. We can also express various constraints on cumulative function expressions, such as AlwaysIn$(f, [a, b), [min, max])$, which ensures that $min \leq f \leq max$ holds for all time points in a up until, but not including, b, and AlwaysIn$(f, var, [min, max])$, which ensures that $min \leq f \leq max$ holds during the processing of interval variable var. Cumulative expression variables are useful for representing vehicle load and energy constraints in vehicle routing problems [119, 25], as well as energy consumption/expenditure in scheduling problems. A cumulative function expression is visualized in Figure 2.4.

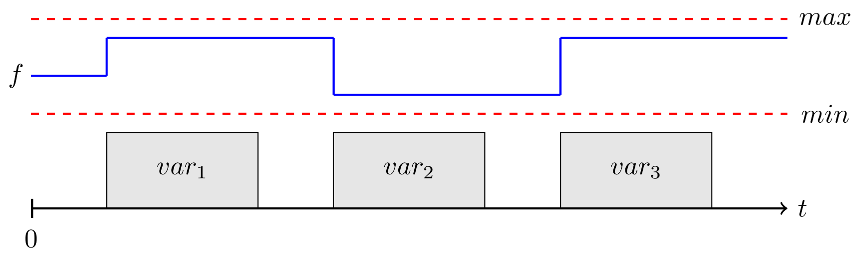


Figure 2.4: Cumulative function expression, f, in blue. Present interval variables (indicated by solid borders) have impact on the variable according to expression constraints. Minimum and maximum bounds can be enforced, as illustrated by the red dashed lines. In this case, $f = \text{StepAtStart}(var_1, impact_1) - \text{StepAtStart}(var_2, impact_2) + \text{StepAtStart}(var_3, impact_3)$.

2.2.1.2 Constraints

In this section, we identify key CP constraints used in this dissertation. We classify constraints as either simple constraints or global constraints. Once again, this is not a complete list of the constraints available in CP. A more comprehensive list can be found in the global constraint catalog [15].[1]

Simple constraints. These constraints are expressed over a single variable and, in the case of interval variables, are often included in the definition of the variable itself.

Definition 2.2.3. Interval variable presence. Pres$(var) \in \{0, 1\}$ *constrains the presence of an optional interval variable. A value of 1 indicates the interval variable, var, is present and 0 indicates it is absent.*

Definition 2.2.4. Interval variable start/end. Start$(var) \in \mathbb{Z}^+$ *and* End$(var) \in \mathbb{Z}^+$ *are used to restrict the start and end time of interval variable, var.*

[1]Online global constraint library: https://sofdem.github.io/gccat/.

Definition 2.2.5. Interval variable length. $\textsc{Length}(var) \in \mathbb{Z}^+$ *is used to constrain the length of interval variable, var.*

Global constraints. A global constraint is a constraint acting on a set of variables that represents a commonly recurring combinatorial substructure [204]. While an equivalent constraint relationship may be expressed with a conjunction of simpler constraints, global constraints can strengthen the performance of solvers by maintaining a better representation of the structure of the problem, often translating to the ability to perform more logical inference as outlined in the next section.

Definition 2.2.6. AllDifferent [203]. *The constraint* $\textsc{AllDifferent}(x_1, \ldots, x_m)$ *requires that all of the integer variables in the scope of the constraint take on different values, namely:* $x_i \neq x_j, \forall i \neq j \in \{1, \ldots, m\}$).

Definition 2.2.7. NoOverlap (or Disjunctive) [207]. *The constraint* $\textsc{NoOverlap}(x_1, \ldots, x_m)$ *requires that each of the optional interval variables in its scope do not overlap, such that:* $\textsc{Start}(x_i) \geq \textsc{End}(x_j) \vee \textsc{End}(x_i) \leq \textsc{Start}(x_j), \forall i \neq j \in \{1, \ldots, m\}$, *if variables x_i and x_j are both present. This constraint can also be posed over a sequence variable, which is in turn defined over sets of optional interval variables. For example:* $\textsc{NoOverlap}(\pi)$ *ensures the variables within sequence variable* $\pi = \textsc{sequenceVar}(\{x_1, \ldots, x_m\})$ *do not overlap. Furthermore, the specification of a pairwise transition time, t_{ij}, identifying transition times between pairs of optional interval variables, x_i and x_j, can be incorporated as follows:* $\textsc{NoOverlap}(\pi, \{t_{ij} : i \neq j \in \{1, \ldots, m\}\})$. *Here, the interval variables within π cannot overlap and must include a transition time, t_{ij}, between pairs of interval variables x_i and x_j, such that* $\textsc{Start}(x_i) \geq \textsc{End}(x_j) + t_{ji} \vee \textsc{End}(x_i) + t_{ij} \leq \textsc{Start}(x_j), \forall i \neq j \in \{1, \ldots, m\}$, *if variables x_i and x_j are both present.*

Definition 2.2.8. Cumulative [208]. *The constraint* $\textsc{Cumulative}(\{x_1, \ldots, x_m\}, z)$ *requires that the total overlapping impact of the interval variables in its scope at any point in time be no greater than z. Formally, this constraint can be expressed as $f \leq z$, where f is a cumulative function expression with elementary contributions,* $f = \sum_{i \in \{1,\ldots,m\}} \textsc{Pulse}(x_i, r_i)$, *and r_i is the impact of interval variable x_i. In the unary resource case with unit interval variable impact, $z = 1$ and $r_i = 1, \forall i \in \{1, \ldots, m\}$, the constraint is equivalent to* NoOverlap.

Definition 2.2.9. Alternative [13]. *The constraint* $\textsc{Alternative}(y, \{x_i, \ldots, x_m\})$ *ensures that if optional interval variable y is present, then exactly one optional interval variable from the set $\{x_1, \ldots, x_m\}$ is present and starts and ends with y. More formally,* $\textsc{Pres}(y) = 1 \rightarrow \Big(\sum_{i \in \{1,\ldots,m\}} \textsc{Pres}(x_i) = 1 \wedge \big(\textsc{Start}(x_i) = \textsc{Start}(y) \wedge \textsc{End}(x_i) = \textsc{End}(y) : \textsc{Pres}(x_i) = 1\big)\Big)$, *otherwise* $Pres(y) = 0 \rightarrow \sum_{i \in \{1,\ldots,m\}} \textsc{Pres}(x_i) = 0$.

Definition 2.2.10. First/last [127]. *The constraint* $\textsc{First}(\pi, x)$ *enforces that optional interval variable x, if present, is the first interval variable in sequence variable π (i.e., $\pi(1) = x$). Similarly, the constraint* $\textsc{Last}(\pi, x)$ *ensures it is the last variable in the sequence (i.e., $\pi(-1) = x$, where $\pi(-1)$ denotes the last variable in the sequence). We note that these constraints do not have direct impact on the start/end values of the intervals, they only constrain the possible permutations of the sequence variable.*

Definition 2.2.11. AlwaysIn [127]. *The constraint* $\textsc{AlwaysIn}(f, [a, b), [min, max])$ *ensures that, for a cumulative function expression variable f, $min \leq f \leq max$ holds for all time points from a up until, but* *not including, b, and a similar constraint* $\textsc{AlwaysIn}(f, var, [min, max])$ *ensures that* $min \leq f \leq max$ *holds during the interval variable* var.

Using these variables and constraints, we can model and solve a variety of combinatorial optimization problems. For example, consider the TSP.

Definition 2.2.12. CP model of the TSP. *Recall that* $V = \{1, \ldots, n\}$ *is the set of cities, and* c_{ij} *is the travel distance from city* i *to* j. *We augment the set of cities to include a dummy copy of the starting city, representing the end return visit, such that* $V' = \{1, \ldots, n, n+1\}$. *We then let* x_i *be a mandatory interval variable with zero length representing a visit to city* $i \in V'$. *We include a sequence variable,* π, *defined over these interval variables. The model as expressed as follows:*

$$\min \quad \textsc{Start}(x_{n+1}) \tag{2.23}$$
$$\textsc{NoOverlap}(\pi, \{c_{ij} : (i,j) \in V' \times V'\}) \tag{2.24}$$
$$\textsc{First}(\pi, x_1), \textsc{Last}(\pi, x_{n+1}) \tag{2.25}$$
$$x_i : \textsc{intervalVar}(0, [0, \infty]) \qquad \forall i \in V' \tag{2.26}$$
$$\pi : \textsc{sequenceVar}(\{x_1, \ldots, x_n, x_{n+1}\}) \tag{2.27}$$

Objective (2.23) *minimizes the start time of the return visit, thus minimizing the total travel distance of the route. Constraint* (2.24) *ensures visits do not overlap, including travel times. Constraint* (2.25) *enforces the first and last visits in the sequence. Finally, Constraints* (2.26) *and* (2.27) *dictate the domains of the interval and sequence variables.*

2.2.2 Solving

In CP, search effort is reduced through the use of logical inference [105]. Each global constraint has an inference algorithm that performs domain filtering: removing possible values from the domains of the variables in the scope of the constraint by proving that a value cannot satisfy the constraint itself and, therefore, cannot participate in a solution to the problem. Since variables are typically in the scope of multiple constraints, value removal by one constraint often triggers subsequent removals from the domains of other variables in neighboring constraints, resulting in the propagation of inferences through the problem representation. The overall solution process is typically a backtracking search [179] where, at each node, inference is performed for each constraint, removing values that are no longer locally consistent. The results of the propagation are then used to inform heuristics which choose the order in which the search tree is explored.

CP solvers have seen significant advances in efficiency in recent decades, transitioning from logic programming into more optimization-based paradigms and becoming an alternative to MILP-based approaches [179].

2.2.2.1 Inference

Inference is performed within CP to remove values from variable domains that cannot possibly take part in a solution. Similar to the way MILP tree search uses the LP relaxation to derive bounds and avoid large areas of the search, CP performs logical inference to reduce search effort.

Consistency. Concepts of consistency have long played a key role in CP and are fundamentally important to the performance of backtracking search algorithms [47]. Consistency conditions can be used to reduce the search space and, consequently, make the problem easier to solve. In this chapter we offer a definition for *domain consistency* (also known as *generalized arc consistency*) but recognize that there are other relevant definitions of consistency [147, 47].

Definition 2.2.13. Domain consistency [204]. *An m-ary constraint, $C(x_1, \ldots, x_m)$, with non-empty domains is called domain consistent* iff *for each variable $x_i : \forall d_i \in D_i, \forall j \in \{1, \ldots, m\} \setminus \{i\}, \exists d_j \in D_j$ such that $(d_1, \ldots, d_m)$ satisfies constraint C.*

Intuitively, the constraint is domain consistent if, after assigning any variable to any value in its domain, there exists a set of values to assign to the other variables such that the constraint is satisfied. From this, we can define a domain consistent CSP as follows:

Definition 2.2.14. Domain consistent CSP. *A CSP is domain consistent if all of its constraints are domain consistent.*

Example 2.2.1. *Consider the CSP: $\mathcal{X} = \{x_1, x_2, x_3\}$, with $D_1 = \{1, 2\}$, $D_2 = D_3 = \{2, 3\}$ and the constraints $x_1 < x_2$, $x_1 < x_3$. This CSP is domain consistent as both of the constraints are domain consistent.*

Establishing domain consistency for a given constraint can be computationally expensive. For this reason, there exist weaker forms of consistency. Domain consistency (or another specified level of consistency) for a constraint is achieved via domain filtering algorithms (as discussed in Section 2.2.2.1). Effort spent filtering domains typically results in less branching in the search tree.

As noted previously in Section 2.2.1.2, global constraints can enhance inference by maintaining a better representation of the structure of the problem. We present an example to illustrate this idea by comparing a clique of not-equals constraints to the AllDifferent global constraint.

Example 2.2.2. *Consider the CSP: $\mathcal{X} = \{x_1, x_2, x_3\}$, with $D_1 = D_2 = D_3 = \{1, 2\}$, and the constraints $x_1 \neq x_2$, $x_1 \neq x_3$, and $x_2 \neq x_3$. Enforcing domain consistency on each constraint does not allow us to perform* any *inference; each constraint is locally domain consistent and the variable domains remain the same. It follows that the CSP is domain consistent as posed (even though its unsatisfiability is apparent).*

Next, consider the impact of a global constraint on the same underlying CSP:

Example 2.2.3. *Consider the CSP: $\mathcal{X} = \{x_1, x_2, x_3\}$, with $D_1 = D_2 = D_3 = \{1, 2\}$, and the constraint* AllDifferent(x_1, x_2, x_3). *Enforcing domain consistency on the* AllDifferent *constraint quickly returns infeasible, as there exists no tuple that will satisfy the constraint. As such, the constraint (and, thus, the CSP) is inconsistent.*

It is often beneficial to represent a relationship of difference among a clique of variables as an AllDifferent constraint; the more global view of problem structure permits stronger inference/more filtering. In general, global constraints are designed for substructures that permit enhanced filtering, not simply to facilitate a more concise expression of CSPs.

Filtering algorithms. A filtering algorithm is a subprocess that, given a constraint with a set of variables in its scope, will prune domain values until the desired level of consistency is achieved (or a

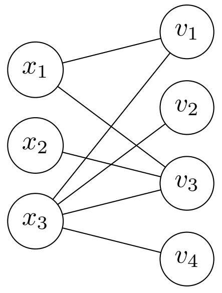


Figure 2.5: AllDifferent filtering: Example bipartite variable-value graph.

runtime/memory limit is exceeded). The process of domain filtering is conducted via *logical inference*, and the algorithms employed can take many forms. During logical inference, pruned domain values are removed when they can be proven to be inconsistent. Often, the removal of a domain value during the filtering of a constraint will trigger the subsequent removal of values from variables in neighboring constraints (i.e., those involving a common variable) resulting in the *propagation* of inferences throughout the problem representation.

The worst-case complexity of filtering algorithms can be polynomial or exponential (i.e., as intractable as the problem being solved), depending on the constraint and desired level of consistency. To mitigate this, weakened forms of consistency can be accepted, or the filtering algorithm can simply be terminated prior to achieving domain consistency. Much of the research effort in the CP community revolves around the design of algorithms that achieve a given level of consistency with lower worst-case complexity than existing methods (e.g., for problems with cumulative resource constraints [153, 208]).

As an illustrative example, we summarize the filtering algorithm for domain consistency of the AllDifferent constraint, proposed by Régin [175]. The state-of-the-art classical filtering algorithm for AllDifferent begins by constructing a bipartite variable/value graph, $G = (X, V, D)$, where V is the set of unique domain values, and D the set of edges representing possible variable/value assignments, as illustrated in Figure 2.5. The example visualized involves variables $X = \{x_1, x_2, x_3\}$ and domains $D_1 = \{v_1, v_3\}$, $D_2 = \{v_3\}$, $D_3 = \{v_1, v_2, v_3, v_4\}$, with $V = \{v_1, v_2, v_3, v_4\}$. An obvious solution to the constraint AllDifferent(x_1, x_2, x_3) would be $x_1 = v_1, x_2 = v_3, x_3 = v_2$, but there are other possibilities as well. A filtering algorithm for this constraint looks to remove all of the values that cannot participate in a solution; for this example, $x_1 = v_3$ is an assignment that will never be feasible and thus v_3 should be pruned from the domain of x_1.

Given such a bipartite graph, G, the filtering of AllDifferent proceeds as detailed in Algorithm 1. The algorithm consists of two primary subroutines: `FindMaximumMatching`, which looks to find a matching of maximum size in the bipartite variable/value graph (i.e., one with the most edges), and `RemoveEdges`, which identifies edges in the graph that can never participate in a maximum matching. The time complexity of Régin's AllDifferent filtering algorithm is $O(|X|\sqrt{|X|}|V|)$, dominated by the `FindMaximumMatching` subroutine [175].

If `FindMaximumMatching` returns a matching whose number of edges, $|M|$, is fewer than the number of variables, $|X|$, then the AllDifferent constraint cannot be satisfied and we terminate. If a maximum matching exists such that each variable can be matched with a value, we proceed to remove edges from the graph which cannot be involved in any maximum matching with the `RemoveEdges` subroutine. While not detailed here, the `RemoveEdges` subroutine searches for edges involved in directed simple paths and strongly connected components (SCCs) in a directed transformation of G, ultimately yielding a set of edges that will never participate in a maximum matching, and thus can be pruned [175]. Leveraging Tarjan's algorithm to search for SCCs [196], the complexity of `RemoveEdges` is $O(|X||V|)$, on the order of the edges in the variable/value graph.

Algorithm 1: AllDifferent filtering algorithm [175]

```
Given: $G = (X, V, D)$;
Result: False if no solution, otherwise filtered domains
$M \leftarrow$ FindMaximumMatching($G$);
if $|M| < |X|$ then
    return False;
end
return RemoveEdges($\mathcal{G}$, $M$);
```

2.2.2.2 Branching

Within CP, branching is similar to MILP in the sense that it partitions the search space by introducing new constraints. Specifically, branches are usually constraints that fix a variable to a specific value from its domain, such as $x_i = d_j$, where $d_j \in D_i$, for an integer variable, or $\text{Pres}(x_i) = 1$ to fix the presence of an optional interval variable, or $\text{Start}(x_i) = z$, to fix the start time of an optional interval variable. Generally, inequality branching (i.e., $x_i \neq d_j$) is used in conjunction with equality branching to form a two-way branching scheme. There also exists k-way branching: a selected variable produces k equality branches representing different value assignments [102].

Within CP there are a number of heuristics commonly employed for variable and value selection. Variables are often prioritized according to their impact, degree in a constraint graph, or domain size, while values are often prioritized highest-first or lowest-first [168], depending on the problem being solved. There are also other strategies, such as randomly ordering these decisions.

2.2.2.3 Complete Algorithm

At each node in the CP branch-and-infer search tree, domain filtering is initiated for each constraint if the domains of the variables in the scope of the constraint have changed. This filtering process prunes values from variable domains, resulting in fewer child nodes to be explored; if any of the variable domains is completely emptied, the subtree is declared infeasible and a backtrack is initiated. The process for solving a CSP can be summarized as follows (assuming two-way branching):

1. Given a CSP and a subproblem set, $\mathcal{S} = \{CSP^{(0)}\}$, where $CSP^{(0)}$ is the root node CSP defined by the tuple $(\mathcal{X}, \mathcal{D}, \mathcal{C})$. Set iteration counter $i = 0$.
2. Repeat until $\mathcal{S} = \emptyset$:
    (a) Select a subproblem, $CSP^{(j)}$, with index j, from set $\mathcal{S}$.
    (b) Repeat until no further propagation:
        i. For each constraint, $C \in \mathcal{C}^{(j)}$, run filtering algorithm until desired level of consistency is achieved.
        ii. Update domains, $\mathcal{D}^{(j)}$, with pruned values.

(c) If there exists an empty domain, $D_k = \emptyset : D_k \in \mathcal{D}^{(j)}$, discard subproblem, $\mathcal{S} \leftarrow \mathcal{S} \setminus \{CSP^{(j)}\}$, and return to 2.

(d) Else if each domain is a singleton, $|D_k| = 1, \forall D_k \in \mathcal{D}^{(j)}$, a feasible solution has been found. Store solution and terminate algorithm.

(e) Else select a variable $x_k \in \mathcal{X}$ with non-singleton domain and generate its children: $\mathcal{S} \leftarrow \mathcal{S} \cup \{CSP^{(i+1)}, CSP^{(i+2)}\}$, where $CSP^{(i+1)} := (\mathcal{X}^{(j)}, \mathcal{D}^{(j)}, \mathcal{C}^{(j)} \cup \{x_k = d_\ell : d_\ell \in D_k^{(j)}\})$ and $CSP^{(i+2)} := (\mathcal{X}^{(j)}, \mathcal{D}^{(j)}, \mathcal{C}^{(j)} \cup \{x_k \neq d_\ell : d_\ell \in D_k^{(j)}\})$. Update iteration counter $i = i + 2$. Discard subproblem, $\mathcal{S} \leftarrow \mathcal{S} \setminus \{CSP^{(j)}\}$, and return to 2.

This summary of CP search is quite general, and the strategy surrounding how a subproblem is selected is not specified. In practice, CP solvers employ a wide variety of search stategies, ranging from depth-first search to fail-first search, as well as dynamic approaches that alter the strategy throughout the solution process [127].

The structure of the branch-and-infer paradigm suggests a trade-off between branching and inferring operations. Successful solvers such as IBM CPLEX CPOptimizer[2] and Google OR-Tools[3] focus on finding a balance between the time spent conducting domain filtering and the time spent branching (i.e., fixing variable assignments) in the tree [127]. These solvers also employ the state-of-the-art filtering algorithms for each of the global constraints they support.

2.2.2.4 Example

To demonstrate a standard implementation of CP branch-and-infer search, consider the following CSP modeled and solved using CP.

Example 2.2.4. *Consider a CSP defined by* $\mathcal{X} = \{A, B, C\}$, $\mathcal{D} = \{D_A, D_B, D_C\}$, *with* $D_A = D_B = D_C = \{1, 2, 3\}$, *and* $\mathcal{C} = \{A > B, \textsc{AllDifferent}(A, B, C)\}$.

Step 1. *Our root node CSP is* $CSP^{(0)} = (\mathcal{X}, \mathcal{D}, \mathcal{C})$. *Run filtering on each constraint,* $c \in \mathcal{C}$.

- *Filtering* $A > B$ *yields* $A \neq 1$ *and* $B \neq 3$. *Update* $D_A \leftarrow \{2, 3\}$ *and* $D_B \leftarrow \{1, 2\}$.
- *Filtering* $\textsc{AllDifferent}(A, B, C)$ *yields no new inferences.*
- *No further propagation.*

Step 2. *Since all domains are non-empty and non-singleton, we generate children by branching on* A*:* $CSP^{(1)} = (\mathcal{X}, \mathcal{D}, \mathcal{C} \cup \{A = 2\})$ *and* $CSP^{(2)} = (\mathcal{X}, \mathcal{D}, \mathcal{C} \cup \{A \neq 2\})$.

Step 3. *We select* $CSP^{(1)}$ *from the subproblem list and run filtering on each constraint,* $c \in \mathcal{C}^{(1)}$.

- $A = 2$ *(branching constraint) results in* $D_A \leftarrow \{2\}$.
- *Filtering* $A > B$ *yields* $B = 1$. *Update* $D_B \leftarrow \{1\}$.
- *Filtering* $\textsc{AllDifferent}(A, B, C)$ *yields* $C = 3$. *Update* $D_C \leftarrow \{3\}$.

Step 4. *Since all domains are singleton, and our problem has no objective function, we have solved the CSP with solution* $A = 2$, $B = 1$, *and* $C = 3$.

[2]https://www.ibm.com/analytics/cplex-cp-optimizer.
[3]https://developers.google.com/optimization/cp/cp_solver.

2.2.3 Software

There are a variety of solvers available for modeling and solving problems in CP. On the commercial side, alongside its MILP solver, the IBM CPLEX Optimization Suite offers CPOptimizer, a rich software system that can be used to model and solve problems with state-of-the-art performance for many classes of scheduling problems [42, 126]. There is also a thriving community of open source solvers and modeling software, including GeCode [184], Choco [107], OR-Tools [165], and MiniZinc [162]. Each year the annual MiniZinc Challenge conducts a test of existing open source software on a variety of benchmarks.[4]

[4]https://www.minizinc.org/challenge.html.

Chapter 3

Literature Review

In this chapter, we review the literature relevant to this dissertation. We focus our review on mixed-integer linear programming (MILP) and constraint programming (CP) modeling for vehicle routing problems, notably those involving electric vehicles. The goal of the review in this chapter is to provide the reader with an idea for how these problems are formulated in each paradigm, and the variety of problem variants that have been investigated with them. We note that the individual chapters in this dissertation provide more focused literature reviews relevant to the specific problem studied in that chapter.

3.1 Vehicle Routing

The modern field of operations research (OR) dates back to World War II in the 1940s, where it was used, primarily in Britain, to provide a quantitative means for decision making throughout the war effort [121]. Since then, OR techniques have become a powerful tool for solving a wide variety of problems prevalent in industries such as manufacturing, transportation, and finance.

Transportation, specifically, has been one of the most successful application areas of OR [135]. Building upon earlier models for the famous *traveling salesman problem* (TSP) [43] that was initially formulated in the 1930s, George Dantzig and John Ramser first proposed the VRP in the late 1950s, under the name "the truck dispatching problem"[44]. The problem looked to find optimal routes for a fleet of delivery trucks between a terminal (depot) and set of service stations. The solution assigned trucks to service stations to minimize the distance traveled by the fleet while satisfying station demands. This problem became commonly known as the vehicle routing problem (VRP).

Definition 3.1.1. Vehicle routing problem (VRP) [134]. *Given a set of vertices (often representing customers), $V = \{v_0, \ldots, v_N\}$, where vertex v_0 is a special vertex representing the vehicle depot, with cost (usually distance) c_{ij} to travel from vertex $v_i \in V$ to vertex $v_j \in V$, we seek a set of least-cost vehicle routes for a fleet of m vehicles such that:*

- *Each vehicle route starts and ends at the depot;*
- *Each vertex in $V \setminus \{v_0\}$ is visited exactly once by exactly one vehicle;*
- *Any applicable side constraints are satisfied.*

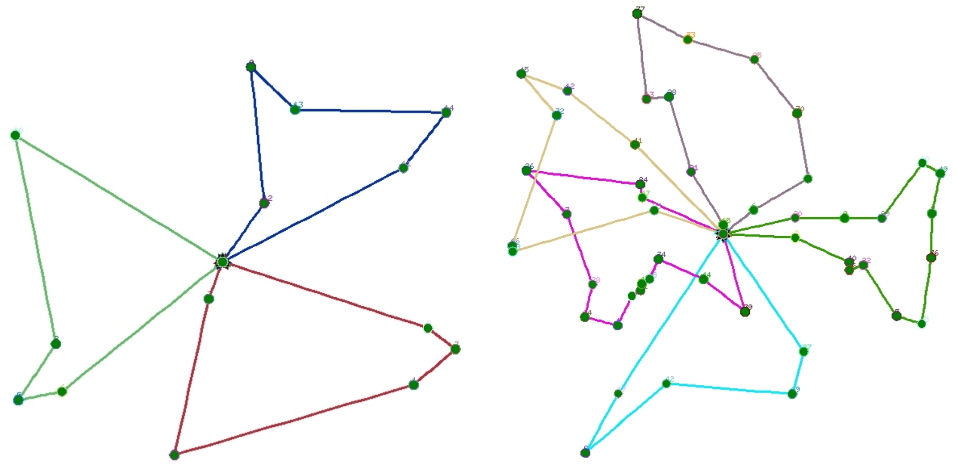


Figure 3.1: Vehicle routing problem (VRP) solutions. *Left:* Solution with three vehicles and 14 customer visits. *Right:* Solution with five vehicles and fifty customer visits.

Example solutions to two instances of VRP without side constraints are visualized in Figure 3.1. Typically, the objective function minimizes the total distance traveled by the vehicle fleet. Common side constraints include delivery demands on the customer visits, capacity restrictions on the vehicles, precedence requirements, and time windows on customer visits.

As posed, this initial definition of the VRP was very simple, lacking complex characteristics inherent in real-world problems.[1] Indeed, over the following sixty years, the VRP was studied intensively, attracting researchers due to both its theoretical value and its practical significance. While we focus on a specific family of VRPs in this dissertation, the interested reader is referred to comprehensive reviews of the vehicle routing problem literature [135, 198].

The field of constraint programming (CP) began to progressively establish itself as an alternative to OR-based approaches for solving hard combinatorial problems in the 1990s [179]. CP found success as a methodology for approaching both satisfaction and optimization problems in a number of application areas, most notably scheduling [10] and vehicle routing [188]. This paradigm focused on combining powerful, expressive, and flexible modeling with programmable search algorithms beyond those typically found in OR-based approaches. Early efforts in CP involved the development of local search-based constraint solvers for approaching vehicle routing problems, touting greater flexibility for integrating complex, real-world side constraints [48, 188].

Until the last twenty or so years, efforts in vehicle routing were primarily focused on minimizing fleet costs associated with travel distance subject to cargo capacity and permissible service time windows. A solution with minimal travel resulted in less wear-and-tear on the vehicles, and a lower consumption of fuel. The underlying prevalence of fossil-fuel refueling stations across logistics networks worldwide meant that a truck could, with high likelihood, be assigned a travel-minimized route without worrying about the details of refuel operations along the way. However, during the 1990s and early 2000s, an increased awareness of the effects of human-caused emissions culminated in efforts to introduce cleaner fuel alternatives, such as ethanol, hydrogen, biofuels, and electricity [56]. With the development of these technologies, the landscape of logistics began to change as fleets incorporated "green" vehicles. These additions brought new challenges to vehicle routing solutions, such as reduced vehicle range and

[1]Without side constraints the problem definition is equivalent to the *multiple traveling salesman problem* (mTSP).

significantly sparser refueling networks.

Studies addressing such restrictions have seen a flurry of activity in the literature in the past decade, with efforts made towards the development of more realistic problem definitions and increasingly efficient solution techniques. Early work on the *recharging vehicle routing problem* (RVRP) investigated the opportunity to allow range-constrained vehicles to recharge at customer sites [39]. The introduction of the *green vehicle routing problem* (GVRP) [60] provided an initial investigation of the challenges associated with alternative fuel-constrained routing problems, and was followed up shortly after by the *electric vehicle routing problem with time windows* (EVRPTW) [183]. Each of these seminal works formulate their respective routing problems as mixed-integer linear programs (MILPs), and propose heuristic approaches for rapidly producing feasible solutions at the expense of optimality. These initial works provided the foundation for the vast swath of studies that soon followed, including the development of more complex problem definitions, sophisticated models, and increasingly efficient algorithms [52, 76, 82]. Indeed, the study of strategies for routing and coordinating electric vehicles has rapidly grown beyond the context of traditional logistics fleets involving truck and trailer. Modern-day research tackles a variety of problem domains, such as those involving the coordination and routing of unmanned aerial vehicles (UAVs) or ground-based mobile robots.

3.2 MILP for Vehicle Routing

In this section we review the main MILP modeling techniques for basic VRPs with time window side constraints. We then use fleet composition and recharge station characteristics to design a classification scheme that allows us to identify the MILP modeling work on electric routing problems that is most relevant to this dissertation.

3.2.1 MILP Models for Basic VRP

MILP models for VRPs are typically defined on a routing graph, $G = (V, A)$, where $V = \{v_1, \dots, v_N\}$ is the set of customer visit vertices and A the set of arcs connecting the vertices [134, 197]. The set of vehicles available for routing is K. The cost for traversing arc $(i, j) \in A$ is given by c_{ij}. The time it takes for vehicle $k \in K$ to travel arc $(i, j) \in A$ is given by t_{ij}^k. The duration of a visit to vertex v_i is given by s_i. The arrival time window for each vertex is given by $[e_i, l_i]$. For modeling purposes, the set V is extended to include instances of the depot vertex, v_0 and v_{N+1} representing the start and end instance of the vehicle depot, respectively. We then define the sets $V_{N+1} = V \cup \{v_{N+1}\}$, $V_0 = V \cup \{v_0\}$, and $V_{0,N+1} = V \cup \{v_0, v_{N+1}\}$. The arc set is defined such that $A = \{(i, j) : v_i \neq v_j \in V_{0,N+1} \times V_{0,N+1}\}$.

3.2.1.1 Three-Index Vehicle Flow Formulations

Initial MILP models for VRP used binary routing decision variables x_{ij}^k which were assigned a value of 1 if vehicle $k \in K$ traveled arc $(i, j) \in A$, and 0 otherwise [84, 67]. Formulations that use these variables are known as three-index vehicle flow formulations [134], and are ideally used when the underlying vehicle fleet is completely heterogeneous (i.e., each vehicle is a unique type with unique characteristics, speed, capacity, etc.). A MILP model of the VRP with time window constraints using this decision variable [134] is provided as follows.

$$
\begin{aligned}
\min \quad & \sum_{k \in K} \sum_{i \in V_0} \sum_{j \in V_{N+1}, i \neq j} c_{ij} x_{ij}^k && && (3.1)\\
\text{s.t.} \quad & \sum_{k \in K} \sum_{j \in V_{N+1}, i \neq j} x_{ij}^k = 1 && \forall i \in V, && (3.2)\\
& \sum_{i \in V_{N+1}, i \neq j} x_{ji}^k - \sum_{i \in V_0, i \neq j} x_{ij}^k = 0 && \forall j \in V, k \in K, && (3.3)\\
& \tau_i + (t_{ij}^k + s_i) x_{ij}^k - l_{N+1}(1 - x_{ij}^k) \leq \tau_j && \forall i \in V_0, j \in V_{N+1}, i \neq j, k \in K, && (3.4)\\
& e_i \leq \tau_i \leq l_i && \forall i \in V_{N+1}, && (3.5)\\
& x_{ij}^k \in \{0,1\} && \forall i \in V_0, j \in V'_{N+1}, i \neq j, k \in K, && (3.6)\\
& \tau_0 = 0. && && (3.7)
\end{aligned}
$$

In addition to the x_{ij}^k variables, this formulation uses continuous variables τ_i which represent the arrival time of a vehicle at vertex v_i. Since each vertex is visited exactly once, the τ_i variables do not need to be indexed by k.[2] The fleet cost (often distance) minimization objective function is given by Eqn. (3.1). Constraint (3.2) ensures that each customer is visited exactly once across the vehicle fleet and Constraint (3.3) enforces that if a vehicle visits a particular customer, it must also leave that customer: these constraints are often called flow conservation constraints. Constraint (3.4) is a labeling constraint that prevents subtours among the visits for a given vehicle by labeling the arrival time of each visit in the route for that vehicle. In this case, the parameter l_{N+1} is a disjunctive constant that represents the routing horizon (i.e., the latest time a vehicle can return to the depot). Finally, Constraints (3.5) through (3.7) detail the domains of the decision variables, where Constraint (3.5) expresses the time window constraint for each vertex. The model has $|K| \cdot |V_{0,N+1}|^2$ binary variables and $|V_{0,N+1}|$ continuous variables.

DFJ Subtour Elimination An alternative modeling strategy for VRP is based on DFJ-style subtour elimination constraints [134], similar to those presented for the TSP in Section 2.1.1. These can be expressed as follows.

$$
\sum_{i \in S} \sum_{j \in S, i \neq j} x_{ij}^k \leq |S| - 1, \quad \forall S \subset V_{0,N+1}, |S| \geq 2, k \in K. \tag{3.8}
$$

Recall that these constraints cannot be implemented as-is in off-the-shelf solvers due to expontentially many subsets S and, as such, need to be evaluated with more sophisticated techniques such as branch-and-cut [66, 136]. Additionally, it is difficult to express certain classes of constraints, such as time windows, using these formulations. For the purposes of this dissertation, we focus on so-called ‘compact’ modeling strategies that can be readily implemented in MILP branch-and-bound solvers.

3.2.1.2 Two-Index Vehicle Flow Formulations

The composition of vehicle fleets, as illustrated in Figure 3.2, has significant impact on the design of VRP models. For electric routing problems, in particular, diverse fleet compositions are becoming more prevalent as older internal combustion engine (ICE) vehicles are deployed in mixed fleets alongside

[2]Indexing would be required for variants where multiple vehicles visit the same vertex.

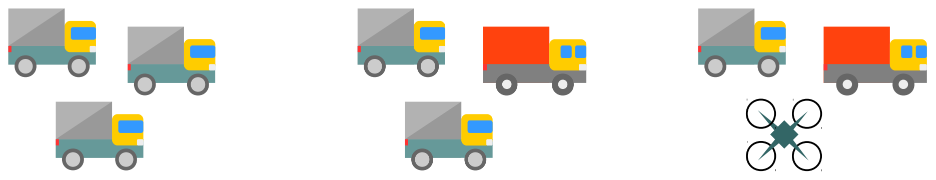


Figure 3.2: Examples of different vehicle fleet compositions. *Left:* Homogeneous fleet. *Middle:* Partially heterogeneous fleet. *Right:* Completely heterogeneous fleet.

electric and hybrid vehicles [98]. Beyond traditional logistics fleets, heterogeneous vehicle types are often seen in joint truck/UAV routing problems [161] as well as those involving the coordination of ground-based mobile robots of different types [79, 124]. Indeed, the exploitation of vehicle fleet symmetry is a primary theme in this dissertation.

Modeling efforts for homogeneous vehicle fleets were developed using a more compact two-index vehicle flow formulation [134, 53]. The general idea here, assuming the characteristics of the vehicles are identical, is to drop the index k from the variables and, instead, use variable x_{ij} which takes on a value of 1 if a vehicle travels from v_i to v_j, and 0 otherwise. Since the vehicles are homogeneous, the travel time on a given arc is given by t_{ij} for all vehicles. A MILP formulation equivalent to the one presented by constraints (3.1) through (3.7) for homogeneous fleets using the more compact two-index vehicle flow formulation [53] is as follows.

$$\min \sum_{i \in V_0} \sum_{j \in V_{N+1}, i \neq j} c_{ij} x_{ij} \tag{3.9}$$

$$\text{s.t.} \sum_{j \in V_{N+1}, i \neq j} x_{ij} = 1 \qquad \forall i \in V, \tag{3.10}$$

$$\sum_{i \in V_{N+1}, i \neq j} x_{ji} - \sum_{i \in V_0, i \neq j} x_{ij} = 0 \qquad \forall j \in V, \tag{3.11}$$

$$\tau_i + (t_{ij} + s_i) x_{ij} - l_{N+1}(1 - x_{ij}) \leq \tau_j \qquad \forall i \in V_0, j \in V_{N+1}, i \neq j, \tag{3.12}$$

$$e_i \leq \tau_i \leq l_i \qquad \forall i \in V_{N+1}, \tag{3.13}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in V_0, j \in V'_{N+1}, i \neq j, \tag{3.14}$$

$$\tau_0 = 0. \tag{3.15}$$

The formulation is very similar to the three-index flow formulation. The fleet distance minimization objective function is given by Eqn. (3.9). Constraints (3.10) and (3.11) enforce the necessary flow through the graph. Constraint (3.12) labels the vertices to prevent subtours, and Constraints (3.13) through (3.14) detail the domains of the decision variables. To bound the number of vehicles in the fleet by m, the number of arcs leaving the depot can be constrained according to $\sum_{j \in V_{N+1}} x_{0j} \leq m$. The model has $|V_{0,N+1}|^2$ binary variables and $|V_{0,N+1}|$ continuous variables, a factor of $|K|$ fewer binary variables than the three-index flow formulation.

It is important to note that the two-index vehicle flow modeling technique no longer tracks each of the vehicle routes explicitly. We provide solutions to a simple homogeneous VRP instance in Figure 3.3 to elucidate this concept. Based on the structure of the solution for the two-index formulation, it is clear that a post-processing step is needed to determine the routes assigned to each of the vehicles. This post-processing step can be accomplished efficiently. When the three-index vehicle flow formulation is used, however, these assignments can be obtained directly from the x_{ij}^k variable values in the solution as indicated in the figure.

Three-index solution:

$k = 1 : x_{0,2}^1 = 1, x_{2,1}^1 = 1, x_{1,0}^1 = 1$

$k = 2 : x_{0,3}^2 = 1, x_{3,0}^2 = 1$

$\tau_0 = 0, \tau_2 = \tau_3 = 1, \tau_1 = 2$

Two-index solution:

$x_{0,2} = 1, x_{2,1} = 1, x_{1,0} = 1, x_{0,3} = 1, x_{3,0} = 1$

$\tau_0 = 0, \tau_2 = \tau_3 = 1, \tau_1 = 2$

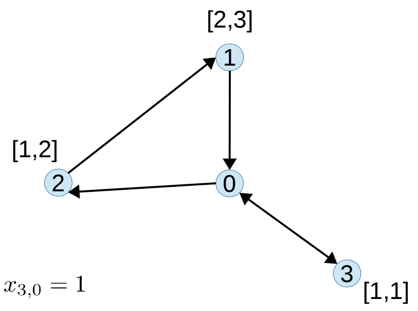


Figure 3.3: Solutions for simple VRP with time windows instance using three-index and two-index vehicle flow formulations. Terms in brackets indicate arrival time windows. There is a unit travel time between all pairs of vertices.

In the case of *partially* heterogeneous fleets, compact formulations use three-index vehicle flow variables, x_{ij}^{λ}, where instead of indexing these variables with each explicit vehicle, $k \in K$, these variables exploit symmetries by indexing based on vehicle *type*, $\lambda \in \Lambda$ [98]. In Figure 3.2, middle, the fleet has two vehicle types such that $|\Lambda| = 2$, one less than the number of vehicles in the fleet. The travel time on a given arc in this case would be a function of vehicle type as well, represented by t_{ij}^{λ}. The labeling variables, τ_i, remain unchanged. To bound the number of vehicles in the fleet, the number of arcs leaving the depot can be constrained according to $\sum_{j \in V_{N+1}} x_{0j}^{\lambda} \leq m_{\lambda}, \forall \lambda \in \Lambda$, where m_{λ} is the number of vehicles in the fleet of type λ. In the case where each vehicle is a unique type, $|\Lambda| = |K|$ and no symmetry in the flow variables can be exploited.

In general, these efforts to exploit vehicle symmetry ensure that branch-and-bound solvers are not overloaded with too many variables, and that problem characteristics that can be efficiently inferred from solutions (e.g., the explicit routes of the vehicles) are not enumerated within the search. In this dissertation, we endeavour to apply this same exploitation of vehicle symmetry to VRP modeling within CP as well. Specifically, Chapter 4 provides techniques for completely homogeneous fleets, while Chapters 5 and 6 propose models for increasingly heterogeneous fleets.

DFJ Subtour Elimination As with the three-index vehicle flow formulation, DFJ-style subtour elimination constraints can be used for the two-index vehicle flow formulation as well by simply dropping index k.

3.2.1.3 Set Partitioning Formulation

Finally, VRP can be modeled with the set partitioning formulation [8]. These formulations use binary decision variable x_j which takes on a value of 1 if route $j \in J$ is used in the optimal solution, and 0 otherwise. In this case, J is the set of all feasible routes. The coefficient a_{ij} is equal to 1 if vertex v_i appears on route j. Finally, c_j is the optimal cost of route j. The formulation is as follows [134].

$$\min \quad \sum_{j \in J} c_j x_j \tag{3.16}$$

$$\text{s.t.} \quad \sum_{j \in J} a_{ij} x_j = 1 \qquad \forall i \in V \tag{3.17}$$

$$x_j \in \{0, 1\} \qquad \forall j \in J \tag{3.18}$$

Objective (3.16) minimizes the cost of the vehicle routes. Constraint (3.17) ensures that each customer vertex is assigned to one route and Constraint (3.18) identifies the domain of the x_j variables.

As the set of feasible routes, $j \in J$, is typically enormous, even for small problems, the set partitioning formulation cannot be directly implemented in MILP solvers. While the DFJ formulations include exponentially many constraints (rows), set partitioning formulations have exponentially many variables (columns). For this reason, approaches based on this formulation are implemented using column generation [54, 62], which is often then embedded within a branch-and-bound algorithm in what is known as branch-and-price. While these approaches are state-of-the-art for many routing problems, they are much less flexible and their implementation is often a difficult task [62].

3.2.2 MILP for Electric Routing Problems

In this section we present a classification scheme for organizing the electric routing literature most relevant to this dissertation. We then review the literature within each of the classes, noting how the contributions of this dissertation fit within each one.

3.2.2.1 Classification Scheme

The core problem characteristics investigated by this dissertation include the composition of the vehicle fleet (i.e., homogeneous vs. heterogeneous) and the recharging mode of the vehicles. We consider two primary classes of recharging stations: those with fixed location and those that are mobile. Additionally, we assume that all recharging station resources are known *a priori*, though we note that there is a large body of literature concerned with using optimization methods for optimal placement and sizing of electric vehicle recharging stations [103, 122]. We define the primary axes of our classification scheme to be these two characteristics. The classification scheme we follow is illustrated by Figure 3.4. Mobile recharging stations are typically associated with heterogeneous vehicle fleets (i.e., one vehicle type services the customer and another provides the mobile recharging). The figure also identifies the primary works within each of the classes, as well as where the chapters of this dissertation belong.

We recognize that the literature body associated with electric and green vehicle routing efforts is extremely large and growing every day. In this section we endeavour to capture only the work most relevant to the contributions of this thesis. The interested reader is invited to examine more thorough reviews on this field [59, 157].

3.2.2.2 Homogeneous Fleets with Fixed Location Recharging

In this section we summarize MILP modeling work on electric routing with homogeneous fleets and fixed location recharging stations. In Table 3.1 we detail key aspects of the primary efforts in this class,

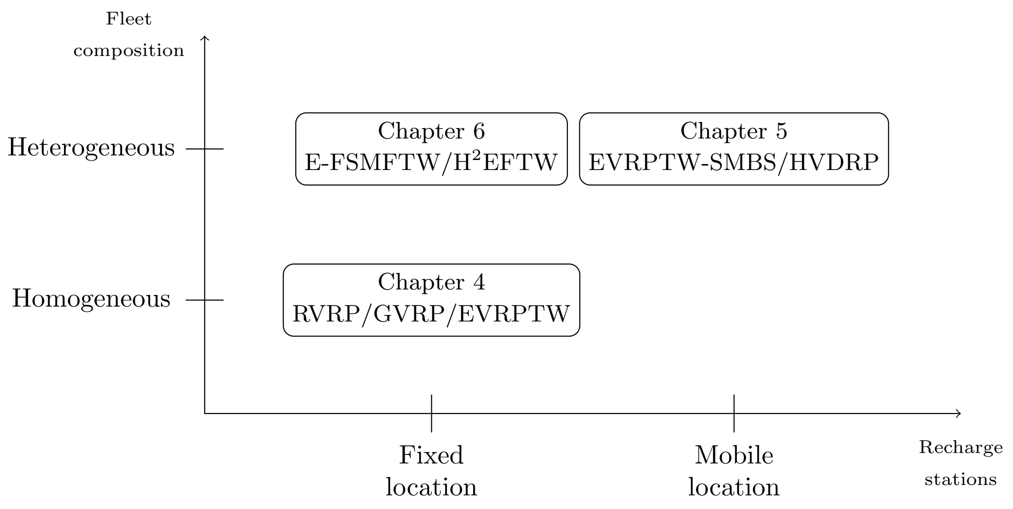


Figure 3.4: Classification scheme for MILP literature on electric routing problems.

including the underlying type of routing formulation used (based on those presented in Section 3.2.1) and the mechanism for modeling vehicle recharges.

The *recharging vehicle routing problem* (RVRP), as proposed by Conrad and Figliozzi, is the first work with these problem characteristics, extending VRP to allow for range-constrained vehicles to recharge at customer locations [39]. Their problem definition also considers vehicle capacity, customer demands, and time windows. The authors presented a three-index vehicle flow formulation with an additional binary decision variable representing whether a vehicle recharges at a given customer site or not. They express various objective functions in their MILP model (e.g., minimizing fleet size, travel distance, and vehicle recharging costs). We note that, since they consider a homogeneous fleet, their model could be reformulated to use two-index flow variables instead, with the appropriate modeling adjustments for cargo tracking. The authors also use an iterative construction heuristic to produce solutions to instances of their RVRP.

At roughly the same time, Erdoğan and Miller-Hooks introduced the *green vehicle routing problem* (GVRP), based on fleets of vehicles that use alternative fuel sources, although they note that their techniques are applicable to any fuel choice [60]. The GVRP introduces the notion of uncapacitated refueling stations, locations that a truck can visit to refuel independent of customer locations. This work models refueling opportunities via the inclusion of additional vertices in the routing graph, producing an *augmented routing graph.* The vast majority of models in electric vehicle routing adopt this particular technique (as described in more detail in Chapter 4). They proposed a compact two-index MILP model as well as two heuristics: one based on the Clarke and Wright savings algorithm [38] and another based on vertex clustering, concluding that the heuristics produce solutions sufficiently close to the MILP-based exact approach and generalize better to larger problems. A number of years later, Bruglieri et al. proposed an alternate formulation for GVRP with fewer variables [32]. This technique pre-processes the network to remove arcs between customer and recharging station vertices that can be proved not to occur in any optimal solution.

Work on the GVRP was extended by Schneider et al. in the context of electric vehicles with the addition of time windows and customer deliveries, resulting in the *electric vehicle routing problem with time windows* (EVRPTW) [183]. The problem as proposed assumes that vehicles recharge fully when

Problem	Year	Formulation	Deliveries	Time Windows	Recharging Capacity	Recharging Amount
RVRP [39]	2011	Three-index	Y	Y	Uncap.	Partial
GVRP [60]	2012	Two-index	N	N	Uncap.	Partial
EVRPTW [183]	2014	Two-index	Y	Y	Uncap.	Full
EVRPTW-PR [34, 33]	2015	Two-index	Y	Y	Uncap.	Partial
EVRPTW [52]	2016	Set partitioning	Y	Y	Uncap.	Partial
EVRP-NL [159]	2017	Two-index	N	N	Uncap.	Partial
EVRPTW-FC [115]	2018	Two-index	Y	Y	Uncap.	Partial
EVRP-NL [76]	2019	Two-index, Multigraph	N	N	Uncap.	Partial
EVRP-NL-C [73]	2019	Two-index, Multigraph	N	N	Capac.	Partial

Table 3.1: Literature summary (selected works). Electric vehicle routing with homogeneous fleets and fixed location recharging stations. ‘Capacity’ refers to whether recharging stations are capacitated or uncapacitated. ‘Amount’ refers to the level of charge vehicles can recuperate at recharge stations.

visiting a recharging station. The authors proposed a two-index flow MILP formulation, similar to that of the GVRP formulation, that uses an augmented routing graph with duplicated recharge station vertices. They also propose a hybrid variable neighborhood search/tabu search heuristic for solving larger instances. They apply their heuristic to both EVRPTW and GVRP instances with favorable results.

Indeed, GVRP and EVRPTW are two very closely related works, each kicking off a large subsequent body of literature. The primary difference between them being that refueling times in the GVRP are assumed constant, and relatively fast, while recharging operations in EVRPTW are variable and may take a significant amount of time. Additionally, EVRPTW included vehicle capacity, customer demands, time windows and instance parameters reflective of the underlying power source (e.g., electricity consumption/recharging rates versus those for biofuels). Given that the contributions of this dissertation revolve around the routing of battery-powered vehicles, we focus our efforts on work stemming from the latter.

After the proposal of the EVRPTW, a number of more sophisticated approaches followed. Bruglieri et al. [34] proposed a MILP model for the EVRPTW while allowing partial recharges (i.e., a vehicle can leave the station before recharging fully); this variant is termed the EVRPTW-PR. Their MILP models are very similar to those of Schneider et al., using two-index flow variables and an augmented routing graph for recharging. They also presented a variable neighborhood search branching approach [34, 33], combining local search and MILP branch-and-bound within a local branching scheme [65]. Their numerical results suggested that a partial recharging strategy is beneficial, when compared to the full recharge MILP model of Schneider et al. [183], and that their variable neighborhood search branching technique is promising for the class of problems. In light of these results, most of the subsequent work in the space favors models that permit the flexibility of partial recharge strategies.

The work of Desaulniers et al. presents a set partitioning formulation for the EVRPTW that contains a huge number of variables precluding the ability to use standard MILP solvers [52]. As such, the authors present exact branch-price-and-cut algorithms for both partial and full recharge variants of

EVRPTW [52]. These approaches employ sophisticated column generation, vertex labeling schemes, and cutting planes, yielding significant improvement over monolithic MILP approaches to the problem. They investigate the impact of allowing partial recharges vs. enforcing full recharges, noting that partial recharges typically result in more recharge instances per vehicle, but lower overall routing costs due to the enhanced routing flexibility.

Within a similar timeframe, the work of Keskin and Çatay investigated partial vs. full recharge policies for EVRPTW with MILP and an adapative large neighborhood search heuristic [114]. Broadly, the works of Bruglieri et al. [34, 33], Desaulniers et al. [52], and Keskin and Çatay [114] all agree that permitting partial recharges can be beneficial to routing costs. Keskin and Çatay later proposed a variant of EVRPTW using fast chargers known as EVRPTW-FC [115]. They propose two MILP formulations, as well as a matheuristic[3] to solve the problem. The first MILP formulation uses two-index flow variables and additional binary variables to determine which charger is used at recharging station visits, while the second formulation duplicates recharge station vertices to represent each recharger type. They evaluate the two formulations and conclude the former provides superior performance.

Thus far, the large majority of summarized work assumes vehicle recharging follows a linear process. Work by Montoya et al. was among the first to investigate electric vehicle routing with nonlinear recharging functions [159]. In this work, they model charging stations with various modes (slow, moderate, and fast) and represent nonlinear recharging functions using piecewise linear functions. They call this variant *electric vehicle routing problem with nonlinear recharging* (EVRP-NL). They propose a MILP formulation that uses two-index flow variables and additional variables for their piecewise linear energy function approximation. The homogeneous fleet is then routed using a hybrid metaheuristic.

Froger et al. followed this work by introducing two new formulations for EVRP-NL [76]. Their first two-index flow formulation modifies the model proposed by Montoya et al. [159] to use arc-based tracking constraints for time and state-of-charge, replacing the node-based tracking constraints used by the former work. Their second formulation leverages a so-called *recharging path multigraph*. The general idea here is to pre-process the routing graph such that, for any pair of customer vertices, only energy-feasible paths are included. This modeling technique builds on previous work for the GVRP by Andelmin and Bartolini [4] and is related to the previously mentioned work of Bruglieri et al. [32]. While the technique requires some additional programming sophistication for graph pre-processing, the resulting solver performance can be significantly more efficient. In follow-up research, Froger et al. also investigate a variant of EVRP-NL where recharging stations are capacitated (i.e., only a set number of vehicles can recharge at a time) called EVRP-NL-C [73]. They propose similar models for this problem and, in contrast to the previous experiments, find that the dominance of the recharging path multigraph model is diminished in the presence of capacitated stations. Their experiments indicate that more typical augmented graph formulations that replicate recharging stations can provide superior performance in this case. In Chapter 4 of this dissertation we provide more details for recharging path multigraph formulations.

At roughly the same time, Zuo et al. [214] developed new linearization techniques for approximating nonlinear recharging, using fewer variables than existing approaches and resulting in a model that outperformed the initial approach of Montoya et al. [159]. At the time of writing, it is unclear how the models of Zuo et al. [214] compare to those of Froger et al. [76] for EVRP-NL. More recently, the work of Lee uses an exact branch-and-price approach to solve EVRP-NL without any of the approximations

[3]Matheuristics are techniques that involve both metaheuristics and mathematical programming.

Problem	Year	Formulation	Vehicles	Time Windows	Recharging
EVRPTW-MF [83]	2015	Three-index (non-linear)	EV, ICV	Y	Partial
FSMVRPTW-EV [137]	2015	Three-index	Assorted	Y	Partial
E-FSMFTW [99]	2016	Three-index, Set partitioning	EV	Y	Full
EVS-FMC [178]	2018	Three-index (non-linear)	EV (buses)	N	Full
H^2EFTW [98]	2019	N/A	EV, ICV, PHEV	Y	Partial
2E-EVRP-BSS [106]	2019	Three-index	EV	N	Full
SRRP-RH (Ch. 6)	2020	Three-index	Social robots	Y	Partial

Table 3.2: Literature summary (selected works). Electric vehicle routing with heterogeneous fleets and fixed location recharging stations.

involved in previous works [138]. The proposed approach makes use of an extended charging stations network, combined with column generation, to address nonlinear charging directly.

The work of Keskin et al. built on the ideas of capacitated recharging stations in previous work by integrating queuing theoretic modeling of waiting times at recharging stations within their models [116]. They assume different recharging station vehicle arrival rates at different times of the day and propose a matheuristic that integrates adaptive large neighborhood search. They compare their approach against results of a MILP formulation, with favorable findings. Setak and Karimpour follow a similar line of research, incorporating queuing time estimates into their multigraph-based MILP model, as well as a simulated annealing approach [186].

Beyond incremental extensions to the original EVRPTW problem definition, there are other bodies of work relevant to the routing of homogeneous electric fleets with fixed location recharging stations. For example, the work of Goeke proposes a variant of the pickup and delivery problem with time windows that uses electric vehicles [82]. They propose a recharging path multigraph-based MILP formulation in addition to a tabu search heuristic. This work is particularly relevant to the comparative analysis performed in Chapter 4 of this dissertation. Moving away from traditional truck-based fleets, the work of Dorling et al. provides a two-index MILP formulation for the routing of a homogeneous fleet of battery-powered UAVs that need to visit a set of locations [57]. The problem designates the depot as a recharge location, allowing UAVs to return to it to retrieve a fresh battery. The authors also provide a simulated annealing algorithm for quickly producing suboptimal solutions to their problem.

The collection of problems and MILP formulations for routing fleets of homogeneous electric vehicles with fixed recharging stations is extensive. In Chapter 4 we adapt some of these modeling techniques to produce novel constraint programming models.

3.2.2.3 Heterogeneous Fleets with Fixed Location Recharging

In this section we summarize related MILP modeling work on electric routing with heterogeneous fleets and fixed location recharging stations. In Table 3.2 we detail key aspects of the primary efforts in this class, including the types of vehicles considered in the fleet.

As research progressed on the routing of homogeneous electric vehicle fleets, separate efforts began to look at heterogeneous (or ‘mixed’) fleet vehicle routing. Initial work on mixed fleets by Goeke and

Schneider optimized the routing of electric vehicles and conventional internal combustion vehicles (ICVs) [83]. This problem variant, coined the *electric vehicle routing problem with time windows and mixed fleet* (EVRPTW-MF), was motivated by the observation that most companies do not operate strictly electric vehicle fleets, but, instead, are gradually introducing them to replace existing vehicles. They provide a mixed-integer non-linear programming formulation that uses three-index vehicle flow variables to model partially-heterogeneous fleets, x_{ij}^{λ}, where λ represents vehicle type and is either electric or conventional combustion (as discussed in Section 3.2.1). Their formulation also incorporates energy and fuel consumption dependent on the type of cargo being carried by the vehicle. As the formulation provided is non-linear, they also offer a neighborhood search heuristic to produce solutions.

Sassi et al. investigated the routing of a mixed fleet of conventional and heterogeneous electric vehicles, denoted VRP-MFHEV [182], and proposed a neighborhood search heuristic to solve the problem. In the work of Lebeau et al., they consider the routing of fleets consisting of multiple types of both conventional combustion and electric vehicles [137]. They name their problem the *fleet size and mix vehicle routing problem with time windows for electric vehicles* (FSMVRPTW-EV), and propose a completely heterogeneous three-index MILP formulation, reflecting the diverse mix of vehicles considered (e.g., quadricycles, small vans, large vans, trucks).

Hiermann et al. looked at routing a mixed electric fleet that differs with respect to transport capacity, battery size, and acquisition cost [99]. The authors introduce the *electric fleet size and mix vehicle routing problem with time windows* (EFSMFTW), which uses linear recharging profiles and uncapacitated charging stations with full recharges. They propose a three-index vehicle flow MILP model for the problem, with indices for each vehicle type. They also present a set partitioning formulation that is solved with branch-and-price and hybrid heuristic approaches. Subsequent work by Hiermann et al., in a very similar vein to the work by Lebeau et al. [137], investigated an EVRP variant that routed ICVs, plug-in hybrid (PHEVs), and electric vehicle fleets [98]. The problem is termed the *hybrid heterogeneous electric fleet routing problem with time windows and recharging stations* (H^{2}EFTW). Recharging profiles are simple (linear) and stations are uncapacitated, however, the authors permit partial recharge strategies in contrast to their original work on EFSMFTW [99]. They propose a sophisticated metaheuristic which combines a genetic algorithm with large neighborhood search to solve the problem, deducing that a careful mix of vehicle types can significantly reduce operating costs when compared to homogeneous fleets.

Driven by the real-world adoption of battery electric buses, Rogge et al. investigate the route optimization of heterogeneous electric bus fleets in the *electric vehicle scheduling fleet size and mix problem with optimization of charging infrastructure* (EVS-FMC) [178]. They propose a solution approach that combines a genetic algorithm with a three-index vehicle flow mixed-integer non-linear program.

Following recent research trends in multi-echelon vehicle routing [41], where logistics routing problems are posed at different levels of the supply chain, Jie et al. investigate the *two-echelon capacitated electric vehicle routing problem with battery swapping stations* (2E-EVRP-BSS) [106]. In this work the authors propose methods for routing two classes of electric vehicles: the first level having a fleet of larger battery and cargo capacity electric vehicles, while the second level having a fleet of smaller electric vehicles. They present a three-index vehicle flow MILP formulation alongside a hybrid approach that utilizes column generation and adaptive large neighborhood search.

While work on routing hybrid truck-UAV fleets usually involves mobile recharging stations (as discussed in the next section), there exists research in the context of ground-based mobile robots with fixed location recharging stations. The *social robot routing problem in retirement homes* (SRRP-RH) as detailed in Chapter 6 of this dissertation involves the routing of heterogeneous fleets of battery-powered social robots. These robots can recharge their batteries at geospatially distributed, and capacitated, recharging stations. The MILP formulation proposed includes a three-index vehicle flow formulation, following previous work on mixed-fleet EVRP, while including a variety of complex side constraints including visit synchronization with human users. This work is related to previously published work on social robot routing in retirement homes [25, 199].

3.2.2.4 Heterogeneous Fleets with Mobile Recharging

In terms of ground-based logistics fleets, the literature covering mobile recharging stations is considerably more sparse than for fixed location problem variants. In these works, typically two types of vehicles exist: the first responsible for making customer deliveries and the second responsible for providing recharge services. Since there is some element of synchronization between the fleets, this category of work is highly related to the subfield of routing with multiple synchronization constraints [58] as well as mothership/drone routing variants [172]. In Table 3.3 we detail key aspects of the primary MILP formulation efforts in this class.

Recent work on the topic provided initial justification for such studies by investigating the use of a mobile battery swapping service for electric vehicles via a battery swapping van [187]. Motivated by this study, Raessi and Zografos introduced the *electric vehicle routing problem with time windows and synchronized mobile battery swapping* (EVRPTW-SMBS) [174]. The work investigates the joint routing of EV and battery-swapping van fleets, where the battery swapping vans extend the autonomy of the EVs. The authors propose a MILP formulation that employs two-index flow variables for each of the vehicle types and an integer variable representing the remaining number of recharged batteries on the battery-swapping vans. Interestingly, in this work the battery-swapping vans are also battery-powered and can run out of charge. The authors demonstrate the value of their proposed problem variant over problems with stationary recharge stations via empirical analyses.

Another type of heterogeneous fleets are those that consist of trucks and unmanned aerial vehicles (UAVs). Such logistics fleets have become more popular in recent years, with the introduction of the *flying sidekick traveling salesman problem* (FSTSP) initiating a large body of research within OR [161]. An example of work that involves such fleets with mobile recharging stations is the *hybrid vehicle-drone routing problem* (HVDRP) proposed by Karak and Abdelghany [111]. In this work, the authors propose a MILP formulation that uses three-index flow variables for the UAV fleet, permitting completely heterogeneous UAV characteristics and two-index flow variables for the ground vehicles. Further, their formulation models the possibility of the UAV traversing arcs while being carried by the ground-based vehicle.

As suggested by the relatively recent proposal of the EVRPTW-SMBS, work on electric routing with mobile recharging stations has been primarily advanced by the multi-UAV literature. In Chapter 5 of this dissertation we investigate the joint routing of range constrained UAVs and ground-based mobile recharging vehicles for target search operations (RC-ST-MRV) [26]. This work is related to existing works in the UAV routing literature. For a more detailed review of works related to this specific application, please see the review provided in Chapter 5.

Problem	Year	Formulation	Vehicles	Time Windows	Recharging
HVDRP [111]	2019	Three-index/two-index	ICVs, UAVs	N	Full
EVRPTW-SMBS [174]	2020	Synch. two-index	EVs, BSVs	Y	Full
RC-ST-MRV (Ch. 5)	2020	Synch. two-index	ICVs, UAVs	Y	Full

Table 3.3: Literature summary (selected works). Electric vehicle routing with heterogeneous fleets and mobile recharging stations.

3.2.3 Other Characteristics

The vast majority of the routing work presented in this section has assumed simple vehicle energy consumption modeling. As such, we note that a rich literature body investigating more complex modeling strategies within the context of electric routing has emerged.

To date, one of more comprehensive assessments of energy consumption modeling in the context of homogeneous electric vehicle routing is the work of Barco et al. [11]. In addition to the integration of a complex power consumption model in their routing approaches, they also propose a battery degradation model that estimates the cost of lithium-ion battery degradation as a function of temperature and other factors. The authors propose a MILP formulation which integrates both power consumption and battery degradation models. They apply their methods to a case study at El Dorado airport in Colombia, noting that the presence of battery degradation modeling can modify recharge patterns that otherwise do not account for this characteristic. The work of Basso et al. introduces the so-called *two-state electric vehicle routing problem* (2sEVRP), integrating dynamics associated with the driving cycle into their energy consumption modeling [12]. Such dynamics include acceleration and braking due to traffic lights and intersections.

3.2.4 Summary

Our review of the relevant MILP literature indicates that homogeneous fleet routing problems have received the most attention. For these problems, the use of two-index formulations is ubiquitous. The use of augmented routing graphs is prevalent throughout, however, recently proposed recharging path multigraph models have shown strong performance for certain classes of problems. Modeling efforts, having started with full recharge and uncapacitated recharge station assumptions, have since progressed to investigate the value of permitting partial recharge strategies and explore the presence of capacity restrictions on recharge stations.

For heterogeneous fleets, three-index formulations are widely used. In this context, the above detailed literature shows applications involving a wide variety of vehicle types, including electric trucks, buses, and robots. While some works use variables indexing each individual vehicle, other works index based on vehicle type, and exploit the symmetry present within a subset of vehicles exhibiting the same characteristics. The literature surrounding heterogeneous fleets with fixed-location recharging stations has been more thoroughly explored, though there exists new research on problems involving mobile recharging stations and the synchronization between fleet vehicles of different types. The benefits of mobile recharging have been demonstrated for both purely land-based vehicle fleets, as well as fleets mixing aerial and land-based vehicles.

3.3 CP for Vehicle Routing

In this section we review the main CP modeling techniques for VRPs, and summarize the relevant literature. The CP literature for vehicle routing problems is considerably smaller than that for MILP approaches, and the body of work in CP relevant to electric routing, outside of the contributions of this dissertation, is limited.

3.3.1 CP Models for Basic VRP

Recall the notation from Section 3.2.1 for the VRP problem definition on a graph. The most common formulations for VRP in CP are *successor variable models* and *interval variable sequencing models*. In this section, we detail these CP formulations for VRP with time windows.

3.3.1.1 Successor Variable Models

These CP models for VRP use successor variables [17, 130]. The primary variable in the formulation is the variable $next_i$ which identifies the direct successor of vertex v_i. The secondary variables are the arrival time at vertex v_i, represented by variable arr_i, and the vehicle that visits vertex v_i, denoted veh_i. Further, the vertex set is augmented such that a copy of the depot visits, v_0 and v_{N+1}, is made for each vehicle. We denote the start and end copies associated with vehicle $k \in K$ as $start(k)$ and $end(k)$, respectively. The set of all vertices is then given by $V' = V \cup \{start(k) : k \in K\} \cup \{end(k) : k \in K\}$. The formulation is detailed by Constraints (3.19) through (3.28).

$$\min \sum_{i \in V \cup \{start(k) : k \in K\}} c_{i,next_i} \tag{3.19}$$

$$next_{end(k)} = start(k) \qquad \forall k \in K, \tag{3.20}$$

$$veh_{end(k)} = veh_{start(k)} = k \qquad \forall k \in K, \tag{3.21}$$

$$\textsc{AllDifferent}(next), \tag{3.22}$$

$$veh_i = veh_{next_i} \qquad \forall i \in V', \tag{3.23}$$

$$arr_i + (s_i + t_{i,next_i}) \leq arr_{next_i} \qquad \forall i \in V \cup \{start(k) : k \in K\}, \tag{3.24}$$

$$arr_i \in [e_i, \ell_i] \qquad \forall i \in V, \tag{3.25}$$

$$next_i \in V' \qquad \forall i \in V', \tag{3.26}$$

$$veh_i \in K \qquad \forall i \in V', \tag{3.27}$$

$$arr_{start(k)} = 0 \qquad \forall k \in K. \tag{3.28}$$

Eqn. (3.19) encodes the cost minimization objective function. Constraint (3.20) ensures that the end visit for a vehicle is followed by its starting visit, thus ensuring the vehicle returns to the depot. Constraint (3.21) details the vehicle assigned to start and end depot visits. Constraint (3.22) uses the AllDifferent constraint to ensure that all of the variables in *next* take on different values (e.g., each visit has a distinct successor visit). Constraint (3.23) ensures that the same vehicle is involved in a visit and its successor visit. Constraint (3.24) enforces transition distances and Constraint (3.25) time window requirements. The remainder of the constraints denote the domains of the decision variables. The formulation has $3 \cdot (|V| + 2 \cdot |K|)$ integer variables.

The above model can be adjusted for heterogeneous fleets with vehicle-dependent transition times (i.e., the speed of each vehicle is different) by indexing an augmented transition matrix, t^k_{ij}, with variable veh_i in Constraint (3.24). The formulation, as stated, exploits the symmetry of the vehicles by ensuring node visit variables are not duplicated across the fleet. Further, some recent works make use of the Circuit global constraint in place of AllDifferent [130, 129] with some modifications to the formulation. The Circuit global constraint uses more intelligent filtering designed specifically for sets of successor variables that need to form a single cycle [72].

3.3.1.2 Interval Variable Sequencing Models

Encouraged by their success for modeling and solving scheduling problems [127], researchers have also explored optional interval and sequence variables for CP formulations of routing problems. The alternative resource model detailed by Constraints (3.29) through (3.35) is similar to an existing model in the literature for the multiple vehicle TSP [201]. The primary decision variable is interval variable x^k_i, representing the visit of vehicle $k \in K$ to vertex $v_i \in V_{0,N+1}$.

$$\min \quad \sum_{k\in K} \sum_{i \in V_{N+1}} \textsc{Pres}(x^k_i) \cdot c_{\textsc{Prev}_{\pi^k}(i),i} \tag{3.29}$$

$$\sum_{k\in K} \textsc{Pres}(x^k_i) = 1 \qquad \forall i \in V, \tag{3.30}$$

$$\textsc{NoOverlap}(\pi^k, \{t^k_{ij} : (i,j) \in A\}), \tag{3.31}$$

$$\textsc{First}(\pi^k, x_0),\ \textsc{Last}(\pi^k, x_{N+1}) \qquad \forall k \in K, \tag{3.32}$$

$$x^k_i : \textsc{optIntervalVar}(s_i, [e_i, \ell_i]), \qquad \forall i \in V, k \in K, \tag{3.33}$$

$$x^k_0 : \textsc{intervalVar}(0, [0,0]), x^k_{N+1} : \textsc{intervalVar}(0, [\ell_{N+1}, \ell_{N+1}]) \qquad \forall k \in K, \tag{3.34}$$

$$\pi^k : \textsc{sequenceVar}(\{x^k_1, \dots, x^k_{N+1}\}) \qquad \forall k \in K, \tag{3.35}$$

Eqn. (3.29) is the fleet cost minimization objective function. Constraint (3.30) ensures that each customer is visited exactly once across the fleet, while Constraint (3.31) ensures that the visits conducted by a vehicle do not overlap, while considering travel times between locations. Constraint (3.32) enforces the depot visits to be first and last in each sequence, and the remainder of the constraints dictate the interval and sequence variable domains. The formulation has $|K| \cdot |V_{0,N+1}|$ optional interval variables and $|K|$ sequence variables.

We note that the above formulation for homogeneous vehicle fleets would duplicate the constraints, interval variables, and sequence variables needlessly (similar to the three-index vehicle flow MILP formulation). One of the technical contributions of this dissertation is the introduction of a so-called single resource transformation for modeling homogeneous VRP variants with CP formulations that involve interval and sequence variables. The details of this technique are introduced and elaborated on in Chapters 4 and 5.

In this dissertation, the CP formulations we propose for the various problems studied all use optional interval and sequence variables. The reasoning for this decision is threefold. First, existing research has demonstrated the superiority of the CP Optimizer constraint solver [127] over other constraint solvers in the context of scheduling problems, particularly when interval variable formulations are used [42]. Second, other studies on CP formulations for VRP have shown that models involving interval and sequence variables in CP Optimizer outperform successor variable formulations on harder problems [35]. Third, the strong scheduling characteristics in some of the problems investigated (particularly those in Chapter 6) lend themselves to the expressivity of this modeling paradigm. As an additional note, an analysis of the literature shows that a significant portion of CP routing work in recent years has also used interval and sequence variables [200, 77, 141, 92, 120, 88].

3.3.2 Related Work

Early work in CP focused on models for the TSP with time windows [166, 167]. These efforts proposed successor variable formulations and a branch-and-bound algorithm for the single vehicle problem, and argued in favor of the flexibility of CP, noting the ease at which the formulation can be adjusted to incorporate more complex side constraints such as multiple time windows [167]. Shaw investigated the first CP-based approaches for solving VRPs with capacity constraints and time windows [188], proposing a neighborhood search heuristic integrated with CP tree search to find solutions that, at the time, were competitive with OR-based metaheuristics. Shortly after, De Backer et al. proposed a successor model for VRP and introduced a system for integrating it with iterative improvement heuristics [49].

CP approaches have also been used to solve routing subproblems in sophisticated decomposition-based approaches to VRP. In these approaches, CP is typically used to model a single vehicle subproblem. Among the first examples of such work is that of Rousseau et al. that explores solving VRPs with time windows using CP-based column generation [180]. This algorithm used the set partitioning MILP formulation of VRP (see Section 3.2.1.3) for the master problem, and a successor variable CP formulation for the TSP with capacity and time window constraints subproblems. The results of their algorithm were comparable to OR-based techniques of the time, however, they constitute a more flexible approach [180]. In a similar vein, Cortés et al. propose a CP-based column generation approach, as part of a larger branch-and-price algorithm, for a technician dispatching routing problem [40]. Similar to the earlier work of Rousseau et al., the approach uses a set partitioning MILP formulation master problem and a successor variable CP formulation for the subproblems. Finally, Lam and Van Hentenryck propose a successor model CP formulation to solve resource-constrained scheduling subproblems as part of a branch-and-price-and-check algorithm [129].

In this work we are primarily concerned with CP formulations for multi-vehicle problems. We summarize recent work on such problems in the following sections.

3.3.2.1 Homogeneous Fleets

As with our MILP model classification scheme presented in Section 3.2.2.1, in this section we discuss recent CP formulations for homogeneous fleets though, in this case, without limiting our discussion to electric routing problems.

Berbeglia et al. proposed a number of works using CP for dial-a-ride problems. They provide successor variable formulations for these problems, and include the CP approach within a hybrid tabu search algorithm [18, 17]. Lam and Van Hentenryck propose a CP formulation based on successor variables for their VRP with location congestion [129]. Their formulation includes all of the successor variables (for all of the vehicles) in the scope of the same Circuit constraint. The authors accomplish this by linking the end of each vehicle's route to the start of the next vehicle, instead of forming distinct cycles for each vehicle.

Vali and Salimifard proposed an interval variable formulation for the mTSP [201]. Their model is similar to the one in Section 3.3.1.2, except it does not model time windows. Gedik et al. propose a similar interval variable-based formulation for the *team orienteering problem with time windows* (TOPTW), along with some variable ordering recommendations [77]. Liu et al. propose an interval variable CP formulation for the *senior transportation problem* (STP), a problem that involves pickup and delivery constraints and ride time restrictions [141]. The authors proceed to compare their CP formulation with a MILP approach and a number of decomposition alternatives with favorable results. Kinable et al. investigate the routing of a homogeneous fleet of snow plows using interval variable formulations [119, 120]. They show that their CP formulation was able to find considerably better routes than those provided by an integer programming approach. Finally, Murín and Rudová propose an interval variable CP formulation to schedule a fleet of homogeneous mobile robots in a manufacturing context [160].

All of the interval variable-based approaches described above utilize explicit vehicle tracking formulations that duplicate model variables for each vehicle in the fleet. For most of these works, this explicit tracking can be avoided via the single resource transformation [23] as detailed in Chapter 4.

3.3.2.2 Heterogeneous Fleets & Vehicle Synchronization

Many works in the CP routing literature involve heterogeneous fleets. Often, there are problem-driven reasons for vehicles of different types to be synchronized in time and space, similar to the MILP work summarized in Section 3.2.2.4. In this section we summarize CP formulations for heterogeneous fleets, once again not limiting our discussion to electric routing problems.

Work on routing/scheduling a heterogeneous fleet of mobile social robots in a retirement home environment proposed an interval variable CP formulation as well as a heuristic decomposition technique designed to solve larger instances [199]. The authors compared their CP approach for this complex problem with both MILP and AI planning approaches with favorable results. The problem was later extended to consider the approximate routing of retirement home residents to produce more realistic daily schedules [25]. This problem definition required synchronization between users and robots for social activities. Chapter 6 of this dissertation further explores these efforts.

Lam et al.'s work on joint vehicle and crew routing proposed a CP formulation based on successor variables for heterogeneous fleet route optimization [130]. As part of the problem, operating crews are able to interchange vehicles resulting in the need for space-time synchronization constraints. They compare their CP formulation with a MILP model with favorable results. Related recent work by Ha et al. proposes an interval variable CP formulation for a VRP variant involving regular customers and special customers [88]. In this case, special customers must be visited by both a regular vehicle and a special vehicle. They represent this characteristic by linking the start times of corresponding interval variables within their formulation, similar to the work in this dissertation in Chapter 6.

Cappart et al. investigated the use of CP for solving transportation problems involving heterogeneous fleets [35]. In contrast to previous work on the STP [141], this work on the *patient transportation problem* (PTP) varied the capabilities of each vehicle (e.g., patients in wheelchairs required special vehicles). Since the patients were not moving in the environment, except when being transported, no synchronization constraints were needed. The authors implement both interval variable and successor variable CP models and find that the former provides better performance [35].

Finally, there also exists work within CP on heterogeneous fleets involving trucks and UAVs [92, 26]. In both of these works, interval variable formulations are posed for route optimization that requires trucks and UAVs to be at common locations at a common time. In Chapter 5, we detail a CP formulation for a variant of this with range-constrained UAVs that must meet up with mobile recharging vehicles on road networks.

3.3.2.3 VRP with Renewable Resources

Although the research in this dissertation constitutes the first formal investigations into electric vehicle routing problems with CP, there are a limited number of existing works that are particularly relevant. These works involve the modeling of routing problems with renewable resources.

As noted previously, one relevant area of work is dial-a-ride and pickup-and-delivery problems [17, 141, 35]. In these problems, vehicles make both pickups and deliveries, whether they take the form of cargo items or people. In contrast to electric routing problems, however, each of the vertices in these problems is usually mandatory to visit, and there often exist strong constraints linking pairs of vertices (i.e., each pickup/delivery pair). Furthermore, the magnitude of the resource recovered is not a function of the time spent at the location, among other differences. For formulations that use interval variables, there are similarities to the models within this dissertation. For example, Liu et al. use cumulative function expressions to model their renewable resources [141] as we do with vehicle energy (see Chapter 4).

Di Gaspero et al. investigate related work on bike share system balancing [55]. In the problem studied, locations are designated as bike 'sinks' or bike 'sources', the former only allowing bikes to be added and the latter bikes to be removed. Source nodes, in a sense, are similar to recharging stations as they allow vehicles to accumulate a resource needed for a future task. The analogy is not without its differences, however, as the consumption of the resource in electric routing is a direct function of the distance traveled. Additionally, in their work, the duration of visits is not dependent on the amount of resource recovered. The authors propose a successor variable model for their problem that uses integer variables to track the load of a vehicle after a visit. Routes are constrained with the Circuit constraint in a similar fashion to the work of Lam and Van Hentenryck [129].

Kinable et al.'s work on snow plow routing is highly relevant to this dissertation. In their original investigation, they looked at routing a fleet of snow plows in the streets of Pittsburgh, Pennsylvania [119]. The fleet of heterogeneous vehicles could consume and replenish both fuel and salt. The authors proposed both a three-index vehicle flow MILP formulation and a CP formulation with interval and sequence variables. Their graph representation of the problem augmented the vertices to allow multiple visits to the salt/fuel depots. Additionally, as in the work in this dissertation, they use cumulative function expression variables and constraints to model these renewable resources. They conclude that the CP approach provides much better performance than the MILP formulation. As described, the snow plow routing problem has distinct similarities to the GVRP, as fueling times are assumed constant. We note that the later version of this work did not include fuel constraints as refueling could often be achieved at salt depots [120]. The multigraph recharging path CP formulations proposed in Chapter 4 of this dissertation, when combined with the single resource transformation for vehicles of the same type, could be promising candidates for the further development of snow plow routing methods.

The very recent work of Ham and Park proposes both MILP and CP approaches to electric vehicle routing under time-of-use electricity pricing [90]. Their CP approach follows the alternative resource model as detailed in Chapter 4 and in Booth and Beck [23].

As aforementioned, the work of Tran et al. [199] and Booth et al. [25] use interval variable formulations to route battery-powered social robots in a retirement home environment. This work is further extended as a core contribution of this dissertation in Chapter 6.

3.4 Summary

In this chapter we provided a review of the literature relevant to this dissertation. We introduced vehicle routing problems (VRPs) and detailed both MILP and CP formulations for a basic variant of VRP with time windows.

In the context of MILP modeling, we provided a classification scheme for organizing our discussion of the rapidly growing literature body on electric vehicle routing. Our investigation of the MILP literature showed that the majority of modeling efforts for homogeneous fleet problems leveraged the two-index vehicle flow formulation, however, there is a growing body of work that uses the recharging path multi-graph modeling technique. These aspects of electric vehicle routing modeling are discussed in more detail in Chapter 4. Further, work involving heterogeneous fleets typically elected for three-index vehicle flow formulations. Our review highlights the diverse fleet mixes that are being routed with MILP approaches, including those involving electric trucks, buses, UAVs, and mobile robots. This area of literature is particularly relevant to Chapter 6. Finally, our review of the mobile recharging station literature revealed an exciting and relatively new area of research relevant to Chapter 5 of this dissertation.

In the context of CP, we reviewed the existing literature on CP for VRPs noting that work on electric routing problems, prior to the contributions of this dissertation, is extremely limited. There have, however, been CP formulation efforts for VRP with synchronization constraints involving heterogeneous fleets. These efforts are particularly relevant to Chapters 5 and 6 where fleet synchronization constraints are used for mobile charging and for joint human-robot activities. Finally, we identify work in CP modeling for problems with renewable resources that are perhaps the most closely related to electric routing problems and note similarities and differences.

Overall, our review of the literature suggests that while MILP efforts in the space of electric routing are expansive and growing, CP is a nascent modeling technology for these problems. The relative performance of CP formulations against their MILP counterparts for routing problems involving renewable resources suggests that the technology is competitive with MILP-based techniques. This observation is thoroughly investigated for four different problems within this dissertation. Further, our investigation demonstrates that while electric routing may have been originally motivated for the optimization of traditional logistics fleets, the characteristics of these problems are relevant in many areas of study, including ground-based and aerial robot coordination.

Chapter 4

Constraint Programming for Vehicle Routing and Pickup and Delivery Problems with Electric Vehicles

In previous chapters, we provided background for the mixed-integer linear programming (MILP) and constraint programming (CP) paradigms for modeling and solving constraint satisfaction and optimization problems. We then summarized the large body of work on electric vehicle routing and related problems, with a specific emphasis on modeling approaches in both MILP and CP.

In this chapter, we present novel CP models for two electric vehicle routing problems, the electric vehicle routing problem with time windows (EVRPTW) and the pickup and delivery problem with time windows and electric vehicles (PDPTW-EV). The developed models are flexible and require only minor modifications to go from one problem to the next. The primary contributions of the chapter include the introduction of a single resource transformation for CP models involving interval, sequence, and cumulative function expression variables. The transformation is superior to the alternative resource CP model on all experiment classes for the EVRPTW, and requires up to an order of magnitude less memory to implement in the utilized solver. We also introduce routing formulations in CP based on recharging path multigraphs.

Contributions to Co-Authored Work. The work in this chapter extends our previously published research in the *Proceedings of the Sixteenth International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research* [23]. Co-author J. Christopher Beck supervised this work.

4.1 Introduction

Driven by emission regulations, government subsidies, and the benefits of an eco-friendly image, electric vehicle (EV) utilization in logistics has seen significant growth in recent years [51]. Outside of logistics, EVs have experienced a growing adoption within the consumer automotive industry [176] and have shown promise in car sharing pilot projects [146]. While not currently cost-competitive with internal combustion engines due to high acquisition costs and limited operational range [64], the benefits of EVs coupled with an increasing number of socially and environmentally-aware consumers are driving the adoption of the technology. The industry has also seen significant investment in the development of required recharging infrastructure. As with traditional fleets, the case for EVs can be significantly bolstered via effective route planning.

The vehicle routing literature has recently addressed this emerging technology through the introduction of the electric vehicle routing problem with time windows (EVRPTW) [183], building on previous work conducted on green logistics, including the green vehicle routing problem (GVRP) [60]. The problem involves routing a fleet of homogeneous vehicles to satisfy customer demands while adhering to the battery capacity and range of the fleet of EVs. The EVRPTW literature has seen considerable research activity, including the development of sophisticated exact approaches [52, 34], metaheuristics [183, 63], and the introduction of increasingly rich problem definitions driven by real-world logistics use cases [159, 99], including pickup and delivery problems involving electric vehicles [85, 82]. There have been, however, no efforts thus far to explore the use of CP to model and solve vehicle routing or pickup and delivery problems involving electric vehicles. A comprehensive review on the electric vehicle routing literature was presented in Chapter 3.

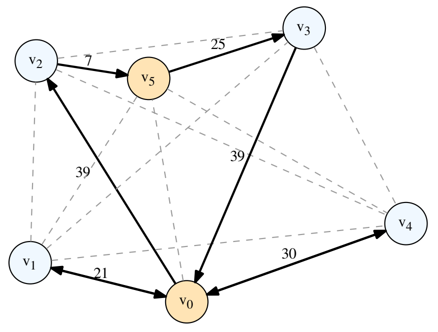
Contributions of chapter. Recognizing EV routing as a strategic area for methodological development, we investigate the use of MILP and CP to solve EV routing problems. The contributions of this chapter are as follows:

1. We propose the first CP approaches for the EVRPTW and the *pickup and delivery problem with time windows and electric vehicles* (PDPTW-EV). Our CP models are based on both augmented graph and recharging path multigraph problem representations, following previous work in the MILP literature. To our knowledge, this is the first time that these graph representations have been used to model the problem in CP.

2. We introduce a single resource transformation for both the EVRPTW and PDPTW-EV in CP. The transformation significantly extends the size of EV routing problems that can be solved with the technology, allowing model expression with orders of magnitude less memory consumption. Such a transformation can be applied to other homogeneous routing and scheduling problems.

3. For the augmented graph formulations of the EVRPTW we demonstrate, through empirical evaluation, that our single resource CP approach outperforms the alternative resource CP model on all experiment classes, and outperforms the augmented graph MILP model for nearly all medium-to-large problem classes. Furthermore, we demonstrate that the single resource CP approach based on multigraph recharging paths is competitive with its corresponding MILP model for fleet minimization problems, specifically for problem instances with longer horizons. We also demonstrate that the proposed CP approaches for the PDPTW-EV are beneficial for large problems, often outperforming their MILP counterparts in terms of feasibility and solution quality.

Outline of chapter. This chapter is organized as follows. Section 4.2 defines both the EVRPTW and PDPTW-EV problems. Section 4.3 details related work for the problems studied. Section 4.4 presents MILP models for both problems and Section 4.5 presents CP models for each problem, identifies various model strengthening techniques, and discusses alternate modeling strategies. Section 4.6 summarizes



Requests

Customer (i)	**Demand** (q_i)	**Time window** ($[e_i, \ell_i]$)	**Service time** (s_i)
1	10	[355, 407]	90
2	20	[176, 228]	90
3	20	[744, 798]	90
4	30	[737, 809]	90

Vehicles

Capacity (C)	**Battery** (Q)	**Consumption rate** (per unit distance, h)	**Recharge rate** (per unit time, g)
200	80	1.00	1.00

Figure 4.1: Example of an EVRPTW instance with four customers (blue nodes) and two recharging stations (depot included; yellow nodes). Bold arcs represent the optimal tours of three vehicles.

various computational analyses and discusses the performance of the algorithms. Finally, Section 4.7 provides concluding remarks.

4.2 Problem Definition

In this section we define both the EVRPTW and PDPTW-EV, including examples for each.

4.2.1 The Electric Vehicle Routing Problem with Time Windows

The EVRPTW is a static optimization problem that aims to route a fleet of electric vehicles to satisfy a set of customer requests. Following existing notation [183], we define an instance of the problem on a graph, $G'(V', A)$. We let $V' = V \cup F'$ be the set of N vertices where $V = \{v_1, \dots, v_n\}$ represents n customer requests, F is the set of recharge stations, and $F' = \{v_{n+1}, \dots, v_N\}$ is the set of augmented recharge station vertices that includes dummy vertices to allow multiple visits to each of the recharging stations in F. We let vertices v_0 and v_{N+1} correspond to start and end instances of the vehicle depot, where each vehicle starts and ends its route. Sets with depot subscripts include the indicated instances of the depot (i.e., $V'_{N+1} = V' \cup \{v_{N+1}\}$ and $V'_{0,N+1} = V' \cup \{v_0, v_{N+1}\}$). The problem is then defined on a graph with vertices $V'_{0,N+1}$ and undirected arcs $A = \{(i,j) \mid i,j \in V'_{0,N+1}, i \neq j\}$. Each arc is assigned a distance, travel time, and energy consumption, d_{ij}, t_{ij}, and $p \cdot d_{ij}$, respectively, where p is a constant energy consumption rate. Vehicles are initially positioned at the depot and start with maximum capacity C, while customer vertices, $i \in V$, are assigned a positive demand, $q_i \leq C$, and a time window, $[e_i, l_i]$.[1] The time window of the start depot, $[e_0, \ell_0]$, is $[0,0]$, and the end depot, $[e_{N+1}, \ell_{N+1}]$, is $[H, H]$, where H is the problem horizon. Each recharging station vertex in the augmented set has a time window of the entire horizon, namely $[e_i = 0, l_i = H], \forall i \in F'$. Customer vertices, $i \in V$, have a service time s_i. The depot instances each have a null service time and the service time at recharging stations is a decision variable. Vehicles have maximum battery capacity Q and recharge linearly at rate g. The problem then requires the minimization of an objective function, often a combination of fleet size and travel distance. In this work, we also investigate minimizing the time to last customer request/delivery. For this objective function, the time window restrictions are removed.

[1]Service must start within the time window.

Example 4.2.1. *Figure 4.1 provides an example EVRPTW instance involving four customer requests and two recharging stations (one of which is the depot). The optimal solution for the fleet distance minimization objective involves three vehicles with vertex sequences:* $(v_0 \to v_1 \to v_0), (v_0 \to v_2 \to v_5 \to v_3 \to v_0)$, *and* $(v_0 \to v_4 \to v_0)$. *These sequences are identified by bold arcs in the figure. The total distance travelled in the optimal solution is* $(2 \cdot 21) + (39 + 7 + 25 + 39) + (2 \cdot 30) = 212$. *Only the second route uses a recharge station vertex.*

4.2.2 The Pickup and Delivery Problem with Time Windows and Electric Vehicles

The PDPTW-EV aims to route a fleet of electric vehicles to satisfy a set of pickup and delivery requests. Figure 4.2 provides an example instance involving two pickups, two deliveries, and two recharging stations (depot included). The PDPTW-EV extends the EVRPTW by redefining the customer vertex set, V, into a set of n pickup, P, and n delivery, D, vertices such that $V := P \cup D = \{v_1, \ldots, v_n\} \cup \{v_{n+1}, \ldots, v_{2n}\}$ and recharge station vertices are re-indexed as $F' := \{v_{2n+1}, \ldots, v_N\}$. V is defined such that delivery vertex $v_{i+n} \in D$ corresponds to pickup vertex $v_i \in P$. Pickups have a positive demand and associated deliveries have a negative demand (i.e., $q_i = -q_{i+n}, \forall i \in P$). Each pickup must be made before its associated delivery and each pickup and delivery pair must be serviced by the same vehicle. As with the EVRPTW, the problem requires the minimization of an objective function, often a combination of fleet size and travel distance.

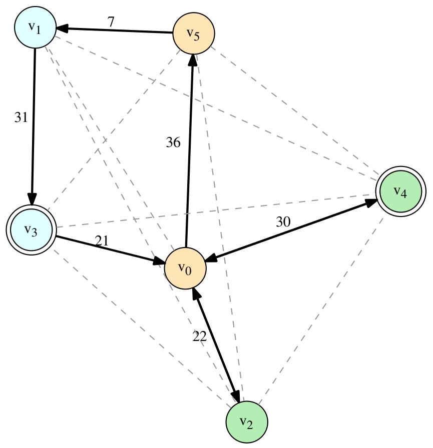
Example 4.2.2. *Figure 4.2 provides an example PDPTW-EV instance involving four customer requests and two recharging stations (one of which is the depot). The optimal solution for the fleet distance minimization objective function involves two vehicles with vertex sequences:* $(v_0 \to v_5 \to v_1 \to v_3 \to v_0)$ *and* $(v_0 \to v_2 \to v_0 \to v_4 \to v_0)$. *These sequences are identified by bold arcs in the figure. The fleet distance travelled in the optimal solution is* $(36 + 7 + 31 + 21) + ((2 \cdot 22) + (2 \cdot 30)) = 199$. *Each route uses a recharge station vertex.*

4.2.3 Assumptions

In this chapter, we make the following assumptions for both the EVRPTW and the PDPTW-EV:

- Fleet vehicles are homogeneous.
- Only full vehicle recharges are permitted.
- Fleet vehicles have sufficient capacity to meet the cargo demand of each request (i.e., $C \geq q_i, \forall i \in V$) for EVRPTW and $C \geq q_i, \forall i \in P$ for PDPTW-EV).
- Fleet vehicles recharge at a constant rate (i.e., linear recharging profile).

These assumptions are common in the literature, and follow the problems as they were originally defined. As discussed in the literature review in Chapter 3, these problems would be classified as homogeneous routing with simple recharging and simple consumption.



Pickups

ID (i)	**Demand** (q_i)	**Time window** ($[e_i, \ell_i]$)	**Service time** (s_i)	**Del. ID**
1	30	[176, 228]	90	3
2	40	[263, 325]	90	4

Deliveries

ID	**Demand**	**Time window**	**Service time**	**Pick. ID**
3	-30	[355, 407]	90	1
4	-40	[737, 809]	90	2

Vehicles

Capacity (C)	**Battery** (Q)	**Consumption rate** (per unit travel time, p)	**Recharge rate** (per unit time, g)
200	80	1.00	1.00

Figure 4.2: Example of a PDPTW-EV instance with two pickups (regular circle nodes), two deliveries (double circle nodes), and two recharging stations (depot included; yellow nodes). Travel times are noted on the arcs. Bold arcs represent the optimal tours of two vehicles.

4.3 Related Work

Research on energy-aware, environmentally conscious vehicle routing is a relatively new area with a flurry of research in recent years [140]. In response to a growing commitment within the United States to investigate alternative fuel sources, the vehicle routing literature introduced the GVRP, detailing a MILP formulation, construction heuristic, and clustering algorithm [60]. Since the introduction of the GVRP, EV routing has grown dramatically, with initial EVRPTW work for homogeneous fleets and full re-charges including a MILP model and a hybrid variable neighborhood search/tabu search solution technique [183]. Subsequent research was developed with approaches for heterogeneous vehicles [99], and partial recharging problem variants [63, 52, 114]. Recent work has also been conducted on modeling non-linear energy consumption [159] as well as incorporating richer, industry-driven problem constraints [3].

The routing literature has since expanded the types of problems that are being solved with consideration for energy consumption with, for example, the introduction of the PDPTW-EV [85]. New formulations with intelligent graph pre-processing schemes have also been investigated for the electric traveling salesman problem with time windows (E-TSPTW) [177], the GVRP [4], EVRPTW with non-linear recharging [75], and the PDPTW-EV [82].

Although the literature for mathematical programming-based approaches (e.g., MILP, branch-and-price, branch-and-cut) for the GVRP and electric routing problems is abundant [60, 183, 52, 99], to the author's knowledge, the use of CP for solving these problems has only recently been investigated in our work [23]. While the performance achieved by sophisticated branch-and-price-and-cut algorithms for such problems [52] is unlikely to be surpassed by monolithic modeling methods utilizing off-the-shelf solvers, the practicality and flexibility of such approaches, including MILP and CP, often translates to more widespread adoption. Furthermore, the development of efficient monolithic models is useful as subproblems within decomposition schemes such as logic-based Benders decomposition [101, 37] rely on their efficacy.

In the past couple of decades, the use of CP for modeling and solving scheduling problems has become competitive with MILP-based approaches [125, 127]. In general vehicle routing, CP has been offered as an alternative to mathematical programming approaches for quite some time [188, 14]. Recent applications include work on the multiple traveling salesman problem [201], team orienteering [77], dynamic dial-a-ride routing [17], bike share balancing [55], joint vehicle and crew routing [131], and patient transportation [35, 141], though these efforts do not consider fuel constraints. While CP had not been explicitly proposed for GVRP nor EVRPTW before our work, previous work on snow plow routing [119] and multi-robot task allocation and scheduling [25, 28] propose CP models with consideration for energy consumption and replenishment. The recent work of Ham and Park proposes both MILP and CP approaches to electric vehicle routing under time-of-use electricity pricing [90].

4.4 Mixed-Integer Linear Programming Models

In this section we present MILP models for both the EVRPTW and the PDPTW-EV based on those existing in the literature. Most of the existing work on MILP models for these problems uses either a formulation based on augmented graphs or one that uses a multigraph. We describe each of these approaches in detail here.

4.4.1 Augmented Graph Formulations

To allow multiple visits to the same recharge station, these models use an augmented problem graph, G', introducing dummy vertices for each of the recharge stations. These formulations are more straightforward to implement in solvers and do not require extensive pre-processing schemes which can require non-negligible time for large problems, however, the number of dummy vertices needs to be selected carefully to reduce network size while not restricting multiple beneficial visits. The inclusion of many redundant vertices often results in the deteriorated performance of these approaches, however, they are useful for representing problems with capacitated recharging stations and recharging station synchronization between vehicles.

4.4.1.1 EVRPTW

An existing two-index MILP model from the literature [183], $\mathbb{MILP}^{G'}_{EVRPTW}$, is detailed by Eqns. (4.1) through (4.11). Binary variable x_{ij} is 1 if arc $(i,j) \in A$ is traveled and 0 otherwise. Continuous variables τ_i, u_i, and y_i represent the arrival time, remaining cargo, and remaining energy, respectively, at vertex $i \in V'_{0,N+1}$. This formulation assumes an unlimited number of homogeneous vehicles are available and only permits full vehicle recharges (i.e., if a vehicle visits a recharge station vertex, the service time is the difference between its maximum energy capacity and current energy level, divided by the recharge rate). The augmented recharge station set, F', is constructed such that the number of dummy vertices associated with each recharge station, n_f, represents the number of times the associated recharge station can be visited across all vehicles (with $|F'| = n_f \cdot |F|$). Following the literature, n_f is set to be relatively small, to reduce the network size, but large enough to not restrict multiple beneficial visits [60]. We note that heuristically choosing a value for n_f, as in the literature, can potentially remove optimal solutions.

$$
\begin{array}{llr r}
\min & \alpha \sum_{j\in V'} x_{0j} + \beta \sum_{i\in V'_0} \sum_{j\in V'_{N+1}, i\neq j} d_{ij}x_{ij} & & (\mathbb{MILP}^{G'}_{EVRPTW}) \\
\text{s.t.} & \sum_{j\in V'_{N+1}, i\neq j} x_{ij} = 1 & \forall i \in V, & (4.1) \\
& \sum_{j\in V'_{N+1}, i\neq j} x_{ij} \leq 1 & \forall i \in F', & (4.2) \\
& \sum_{i\in V'_{N+1}, i\neq j} x_{ji} - \sum_{i\in V'_0, i\neq j} x_{ij} = 0 & \forall j \in V', & (4.3) \\
& \tau_i + (t_{ij} + s_i)x_{ij} - l_{N+1}(1 - x_{ij}) \leq \tau_j & \forall i \in V_0, j \in V'_{N+1}, i \neq j, & (4.4) \\
& \tau_i + t_{ij}x_{ij} + g(Q - y_i) - M(1 - x_{ij}) \leq \tau_j & \forall i \in F', j \in V'_{N+1}, i \neq j, & (4.5) \\
& 0 \leq u_j \leq u_i - q_i x_{ij} + C(1 - x_{ij}) & \forall i \in V'_0, j \in V'_{N+1}, i \neq j, & (4.6) \\
& 0 \leq y_j \leq y_i - (p \cdot d_{ij})x_{ij} + Q(1 - x_{ij}) & \forall j \in V'_{N+1}, i \in V, i \neq j, & (4.7) \\
& 0 \leq y_j \leq Q - (p \cdot d_{ij})x_{ij} & \forall j \in V'_{N+1}, i \in F'_0, i \neq j, & (4.8) \\
& e_j \leq \tau_j \leq l_j & \forall j \in V'_{0,N+1}, & (4.9) \\
& x_{ij} \in \{0, 1\} & \forall i \in V'_0, j \in V'_{N+1}, i \neq j, & (4.10) \\
& \tau_0 = 0, u_0 = C, y_0 = Q. & & (4.11)
\end{array}
$$

The objective function is weighted, where $\alpha \in [0, 1]$ identifies the emphasis on fleet size minimization and $\beta \in [0, 1]$ on travel distance minimization. Constraint (4.1) ensures each customer request is satisfied, while Constraint (4.2) restricts each recharge station in the augmented set to be visited at most once (due to the augmented vertices, each recharge station can, therefore, be visited at most n_f times). Constraint (4.3) enforces the flow for non-depot nodes. Constraints (4.4)-(4.5) prevent the formation of subtours, with disjunctive constant $M = (l_{N+1} + g \cdot Q)$. Constraint (4.6) ensures demand fulfillment at customer vertices and Constraints (4.7)-(4.8) constrain energy levels to be feasible. Constraint (4.9) requires customer visits to satisfy the time windows and Constraints (4.10)-(4.11) identify binary and continuous variable domains. The $\mathbb{MILP}^{G'}_{EVRPTW}$ formulation has $|V'_{0,N+1}|^2$ binary variables and $3 \cdot |V'_{0,N+1}|$ continuous variables.

Two Index Formulation. With the exception of x_{ij}, the variables are continuous, modeling visit time, load, and energy level via sequencing constraints. The model represents multiple vehicles by relaxing the unitary out- and in-flow on the start and end depot vertices, respectively. This modeling technique is effective as it does not multiply the number of variables by the number of (symmetric) vehicles.

Problem Variants. The fixed fleet variant can be modeled with the inclusion of a constraint of the form: $\sum_{j\in V'} x_{0j} \leq m$, where m is the fleet size. A variant with heterogeneous vehicles can be modeled for k different vehicle types by adding an index for the vehicle type to the arc, cargo, and energy consumption variables (i.e., x^k_{ij}, u^k_i, and y^k_i), with similar adjustments to the parameters [99]. Additional problem variants, such as partial recharges, can also be considered through the inclusion of additional constraints/variables [52, 34].

4.4.1.2 PDPTW-EV

An augmented graph model for the PDPTW-EV, $\mathbb{MILP}^{G'}_{PDPTW\text{-}EV}$, is given below. To ensure a given pickup and delivery pair are serviced by the same route, continuous variables δ_i are introduced to indicate the route that serves vertex $v_i \in P$. Pickups are assigned a positive value, $q_i, \forall i \in P$, and deliveries are assigned an equal and opposite value, $(-1) \cdot q_i = q_{i+n}, \forall i \in P$. Finally, labeling parameter θ_j is set to be equal to the index of vertex j, and $\theta_{max} = |V|$ is used as a disjunctive constant.

$$
\begin{aligned}
\min \quad & \text{Objective } (\mathbb{MILP}^{G'}_{EVRPTW}) && && (\mathbb{MILP}^{G'}_{PDPTW\text{-}EV}) \\
\text{s.t.} \quad & \text{Constraints } (4.1) - (4.5), (4.7) - (4.10) \\
& u_j \geq u_i + q_i x_{ij} - C(1 - x_{ij}) && \forall i \in V'_0, j \in V'_{N+1}, i \neq j, && (4.12) \\
& 0 \leq u_i \leq C && \forall i \in V'_{N+1}, && (4.13) \\
& \tau_0 = 0, u_0 = 0, y_0 = Q, && && (4.14) \\
& \theta_j x_{0j} \leq \delta_j && \forall j \in V', && (4.15) \\
& \theta_j x_{0j} + \theta_{max}(1 - x_{0j}) \geq \delta_j && \forall j \in V', && (4.16) \\
& \delta_i - \theta_{max}(1 - x_{ij}) \leq \delta_j && \forall i, j \in V', && (4.17) \\
& \delta_i + \theta_{max}(1 - x_{ij}) \geq \delta_j && \forall i, j \in V', && (4.18) \\
& \delta_i = \delta_{i+n} && \forall i \in P, && (4.19) \\
& \tau_i + t_{i,i+n} \leq \tau_{i+n} && \forall i \in P, && (4.20) \\
& \delta_i \geq 0 && \forall i \in V. && (4.21)
\end{aligned}
$$

The formulation is similar to $\mathbb{MILP}^{G'}_{EVRPTW}$ for EVRPTW with additional constraints to ensure each pickup and delivery is serviced on the same route. Constraints (4.12) to (4.14) are updated cargo constraints to reflect pickups and deliveries. Constraints (4.15) and (4.16) label each route by the index of its first visited vertex. Constraints (4.17) and (4.18) ensure the first labels are propagated to the remainder of the vertices involved in the route and Constraint (4.19) ensures that the route label is the same for a corresponding pickup/delivery pair. Finally, Constraint (4.20) expresses the required precedence, namely that pickup visits come before their associated deliveries and Constraint (4.21) the variable domain.

4.4.2 Recharging Path Multigraph Reformulations

Formulations based on multigraphs offer an alternative to augmented path formulations, and use so-called *recharging paths* to avoid the use of dummy vertices. These models have become popular alternatives to the dummy formulation for problems similar to the EVRPTW, such as the GVRP [4], the PDPTW-EV [82], and the E-TSP [177]. Avoiding the use of dummy vertices is convenient, as determining their cardinality is a difficult task, however, the use of recharging paths requires a fairly involved pre-processing of the graph that, as noted in Section 4.6, can take non-negligible time for large problems, precluding the use of the model for applications requiring near-instant results. As such, this model cannot be used with commercial solvers in a purely 'out-of-the-box' fashion. Furthermore, problem variants that involve capacitated recharging stations or the synchronization of vehicles at recharging stations cannot be represented with this modeling technique in a straightforward way. As a final comment, these formulations can become unwieldy if the number of arcs in the resulting multigraph is large.

4.4.2.1 EVRPTW

Following previously presented notation for the PDPTW-EV [82], we formulate the problem on a directed multigraph (i.e., vertices are now connected by multiple arcs) $\mathcal{G} = (V_{0,N+1}, \hat{A})$, where $\hat{A} = \{(i,j)^h : i \in V_0, j \in V_{N+1}, h \in \mathcal{H}(i,j), i \neq j\}$ is the extended arc set and each $h \in \mathcal{H}(i,j)$ is a possible recharging path, a sequence of visits to recharging stations, from customer vertex i to j. Multigraph vertices consist of only customer and depot vertices; recharge vertices are incorporated into the construction of the recharge paths. We let $(i,j)^0$ represent the direct path from i to j without visiting any recharge stations, whereas $(i,j)^h, h \geq 1$ denotes some path $\langle i, w, \ldots, u, j\rangle$ where one, or multiple, recharge stations $w, u \in F$ are visited between i and j. Each of these paths, $(i,j)^h$, is associated with a total distance, d_{ij}^h and travel time t_{ij}^h (not including recharging time). Each of these paths is also associated with a total energy consumption, $p \cdot d_{ij}^h$, energy required to travel to the first recharge station on the path, $\hat{\gamma}_{ij}^h$, energy required to travel from the last recharge station to j, $\tilde{\gamma}_{ij}^h$, and the total energy that can be replenished on the recharge path, Q_{ij}^h.

Following the procedure outlined in previous work [82, 4], we generate the multigraph arc set, $\hat{A}$, with the Floyd-Warshall shortest-path algorithm. First, we define a graph that only involves the recharge station vertices, $G^F = (F, A^F)$, where the arc set A^F includes only those arcs that are energy-feasible (i.e., can be traversed by a vehicle with full starting energy). We then calculate the shortest path between any pair of recharge stations, $w, u \in F, w \neq u$, using Floyd-Warshall. We generate the set of recharge paths that can possibly participate in the optimal solution between i and j, $\mathcal{H}(i,j)$, by adding: i) the direct path $\langle i, j\rangle$ with no recharging, ii) all of the possible single-recharge paths, $\langle i, w, j\rangle, \forall w \in F$, and iii) each of the shortest paths $\langle i, \mathcal{SP}(w,u), j\rangle$ between all pairs of recharge stations $w, u \in F, w \neq u$. We then remove all paths that are dominated by some other path (i.e., the total distance, energy consumption, energy from i to first station and energy from last station to j are all greater than another path).

With our generated multigraph, $\mathcal{G}$, we formulate the EVRPTW using $\mathbb{MILP}^{\mathcal{G}}_{EVRPTW}$ as defined by Constraints (4.22) through (4.35). Decision variables τ_i, u_i, and y_i retain their original definitions, whereas routing decision variable x_{ij}^h now takes on a value of 1 if a vehicle travels to j immediately after i via recharge path h. The formulation also introduces new continuous decision variables Y_{ij} that represent the total amount recharged by the vehicle traveling between i and j.

$$\min \alpha \sum_{j \in V} \sum_{h \in \mathcal{H}(i,j)} x_{0j}^h + \beta \sum_{i \in V_0} \sum_{j \in V_{N+1}} \sum_{h \in \mathcal{H}(i,j)} d_{ij}^h x_{ij}^h \qquad (\mathbb{MILP}^{\mathcal{G}}_{EVRPTW})$$

$$\text{s.t.} \sum_{j \in V_{N+1}} \sum_{h \in \mathcal{H}(i,j)} x_{ij}^h = 1 \qquad \forall i \in V, \quad (4.22)$$

$$\sum_{i \in V_0} \sum_{h \in \mathcal{H}(i,j)} x_{ij}^h = 1 \qquad \forall j \in V, \quad (4.23)$$

$$\tau_i + \sum_{h \in \mathcal{H}(i,j)} t_{ij}^h x_{ij}^h + g \cdot Y_{ij} - \ell_{N+1}(1 - \sum_{h \in \mathcal{H}(i,j)} x_{ij}^h) \leq \tau_j \qquad \forall i \in V_0, j \in V_{N+1}, \quad (4.24)$$

$$u_i - q_i + C(1 - \sum_{h \in \mathcal{H}(i,j)} x_{ij}^h) \geq u_j \qquad \forall i \in V_0, j \in V_{N+1}, \quad (4.25)$$

$$y_i + Y_{ij} - \sum_{h \in \mathcal{H}(i,j)} (p \cdot d_{ij}^h) x_{ij}^h + Q(1 - \sum_{h \in \mathcal{H}(i,j)} x_{ij}^h) \geq y_j \qquad \forall i \in V_0, j \in V_{N+1}, \quad (4.26)$$

$$\sum_{h\in\mathcal{H}(i,j)} Q_{ij}^h x_{ij}^h \geq Y_{ij} \qquad \forall i \in V_0, j \in V_{N+1}, \quad (4.27)$$

$$\sum_{j\in V_{N+1}} \sum_{h\in\mathcal{H}(i,j)} \hat{\gamma}_{ij}^h x_{ij}^h \leq y_i \qquad \forall i \in V, \quad (4.28)$$

$$Q - \sum_{j\in V_0} \sum_{h\in\mathcal{H}(i,j)} \tilde{\gamma}_{ij}^h x_{ij}^h \geq y_j \qquad \forall j \in V, \quad (4.29)$$

$$Y_{ij} \geq (Q - y_i) + \sum_{h\in\mathcal{H}(i,j)} ((p \cdot d_{ij}^h) - \tilde{\gamma}_{ij}^h) x_{ij}^h - Q(x_{ij}^0 + (1 - \sum_{h\in\mathcal{H}(i,j)} x_{ij}^h)) \qquad \forall i \in V_0, j \in V_{N+1}, \quad (4.30)$$

$$e_i \leq \tau_i \leq \ell_i \qquad i \in V, \quad (4.31)$$

$$0 \leq u_i \leq C \qquad \forall i \in V_{0,N+1}, \quad (4.32)$$

$$x_{ij}^h \in \{0,1\} \qquad \forall i \in V_0, j \in V_{N+1}, h \in \mathcal{H}(i,j), \quad (4.33)$$

$$y_i \geq 0 \qquad \forall i \in V_{0,N+1}, \quad (4.34)$$

$$Y_{ij} \geq 0 \qquad \forall i \in V_0, j \in V_{N+1}. \quad (4.35)$$

In formulation $\mathbb{MILP}_{EVRPTW}^{\mathcal{G}}$, the objective function is the same as in previous formulations. Constraints (4.22) and (4.23) are degree constraints ensuring that each customer vertex has an incoming arc and an outgoing arc (visit and departure resp.). Constraint (4.24) ensures that the arrival times of each vertex are temporally consistent, including travel and recharge time, while Constraint (4.25) dictates the required cargo level at each node. Constraint (4.26) ensures that each vertex is labeled with the correct value of remaining energy. Additionally, Constraint (4.27) ensures the energy recharged between two customer vertices is less than the maximum potential for the selected recharge path. Constraint (4.28) dictates that the remaining energy at a vertex must be greater than the energy required for the first visit on its selected recharge path. Constraint (4.29) expresses that the remaining energy at the destination vertex must be less than the vehicle energy capacity minus the energy consumed on the last leg of the recharge path. Constraint (4.30) ensures that each time a vehicle visits a recharge station, it recharges fully.[2] Constraint (4.31) ensures time windows on customer visits are satisfied. The remaining Constraints (4.32) through (4.35) dictate the binary and continuous domains of the decision variables.

The multigraph reformulated model has $3 \cdot |V_{0,N+1}| + |V_0| \times |V_{N+1}|$ continuous variables, and $|V_0| \times |V_{N+1}| \times | \cup_{(i,j)\in V_{0,N+1}} \mathcal{H}(i,j)|$ binary variables. As is evident, the number of binary variables scales with the number of recharge paths; as pointed out in previous work [4], in the worst case the number of recharge paths grows with $|V|^2 \times |F|^2$.

4.4.2.2 PDPTW-EV

Similar to the augmented graph formulation for PDPTW-EV, the multigraph formulation, $\mathbb{MILP}_{PDPTW\text{-}EV}^{\mathcal{G}}$, adds variables and constraints to $\mathbb{MILP}_{EVRPTW}^{\mathcal{G}}$ to ensure the pickup and delivery problem characteristics are modeled. As originally presented [82], the formulation is given as follows:

$$\min \quad \text{Objective } (\mathbb{MILP}_{EVRPTW}^{\mathcal{G}}) \qquad (\mathbb{MILP}_{PDPTW\text{-}EV}^{\mathcal{G}})$$

[2]This constraint is a corrected version of the constraint that appears in Goeke [82].

s.t. Constraints (4.22) − (4.24), (4.26) − (4.35)

$$u_i + q_i - C(1 - \sum_{h \in \mathcal{H}(i,j)} x_{ij}^h) \leq u_j \qquad \forall i \in V_0, j \in V_{N+1}, \tag{4.36}$$

$$\theta_j \sum_{h \in \mathcal{H}(i,j)} x_{0j}^h \leq \delta_j \qquad \forall j \in V, \tag{4.37}$$

$$\theta_j \sum_{h \in \mathcal{H}(i,j)} x_{0j}^h + \theta_{max}(1 - \sum_{h \in \mathcal{H}(i,j)} x_{0j}^h) \geq \delta_j \qquad \forall j \in V, \tag{4.38}$$

$$\delta_i - \theta_{max}(1 - \sum_{h \in \mathcal{H}(i,j)} x_{ij}^h) \leq \delta_j \qquad \forall i, j \in V, \tag{4.39}$$

$$\delta_i + \theta_{max}(1 - \sum_{h \in \mathcal{H}(i,j)} x_{ij}^h) \geq \delta_j \qquad \forall i, j \in V, \tag{4.40}$$

$$\delta_i = \delta_{i+n} \qquad \forall i \in P, \tag{4.41}$$

$$\tau_i + t_{i,i+n} \leq \tau_{i+n} \qquad \forall i \in P. \tag{4.42}$$

Constraint (4.36) is an updated cargo constraint to reflect pickups and deliveries. The additional Constraints (4.37) through (4.42) are identical to those in $\mathbb{MILP}^{G'}_{PDPTW\text{-}EV}$, namely Constraints (4.15) through (4.42), with modifications to account for the presence of recharge paths and recharge path decision variable x_{ij}^h.

4.5 Constraint Programming Models

In this section we present CP formulations for the EVRPTW and the PDPTW-EV. The proposed formulations follow the augmented graph and multigraph strategies employed in MILP modeling. Our CP models are posed as scheduling formulations with optional activities [126, 127]. As is becoming increasingly common in CP-based approaches [25, 35, 141], the proposed models make use of three primary decision variable types: optional interval variables, sequence variables, and cumulative function expressions, as introduced in Section 2.2 above.

4.5.1 Augmented Graph Formulations

In this section we present CP models based on the augmented graph formulation. Similar to the augmented graph MILP formulations, these models duplicate recharge station variables to permit multiple visits.

4.5.1.1 EVRPTW

As with the MILP formulations, we first present models for the EVRPTW and extend these to capture the characteristics of the PDPTW-EV.

Alternative Resource Model. Our first CP model for the EVRPTW follows the traditional alternative resource model for formulating VRPs in CP [119, 35], and, in contrast to the two-index MILP presented in the previous section, explicitly represents the vehicles. The modeling approach is illustrated in Figure 4.3. At a high level, this approach associates a sequence variable, defined over a set of optional interval variables representing customer and recharge station visits, with each of the vehicles.

To model the EVRPTW in CP, we must determine an upper bound on the number of vehicles that may be used in an optimal solution.

Lemma 4.5.1. *The smallest general upper bound on $|K|$ is the number of customer requests, $|V|$.*

Proof. Consider a problem instance where all customers have the same single-instant time window such that $e_i = e_j, \forall i < j \in V$, $e_i = \ell_i, \forall i \in V$, and $t_{ij} > 0, \forall i \neq j \in V$. Clearly, since all customer requests must happen at the same time, any feasible solution to the problem must involve $|K| = |V|$ vehicles. $\square$

As such, we define an upper bound on the number of vehicles to be equal to the number of customer requests, $|K| = |V|$, representing a scenario where each customer is serviced by a separate vehicle. For each customer request, $i \in V$, we introduce a mandatory interval variable, $\bar{x}_i$. We create an optional interval variable, x_i^k, for each vertex, $i \in V'$, for each vehicle, $k \in K$. We also introduce start and end interval variables, x_0^k and x_{N+1}^k, with null duration for each vehicle to represent the depot.

The model considers a set of $|V'_{0,N+1}|$ interval variables and a sequence variable, π^k, for each vehicle $k \in K$. Each interval variable x_i^k, for all $i \in V'_{0,N+1}$, represents the time period in which the vehicle visits i. Thus, expressions $\textsc{StartOf}(x_i^k)$ and $\textsc{EndOf}(x_i^k)$ correspond to the arrival and departure time of vehicle k at location i, respectively. The expression $\textsc{Pres}(x_i^k) = 1$ if vehicle $k \in K$ visits location i (i.e., the interval variable is present in the solution), and 0 otherwise. Sequence variable π^k is defined over the set of interval variables involving vehicle k, and represents the sequence of visits. Vehicle load consumption and energy level consumption/replenishment are modeled with cumulative function expressions. We let $\mathcal{C}^k$ and $\mathcal{Q}^k$ be cumulative function expressions representing the load and energy level of vehicle $k \in K$ throughout its route.

Our alternative resource CP model is detailed by Eqns. (4.43)-(4.56). The objective represents the minimization of fleet size and distance traveled. Constraint (4.43) ensures that each customer is serviced by one vehicle and Constraint (4.44) enforces that tasks assigned to a vehicle, represented by the sequence variable π^k, do not interfere temporally, including travel times. Constraints (4.45)-(4.46) ensure that the vehicle load does not fall below zero over the planning horizon, represented as a cumulative function expression with negative impact for served customers. Constraints (4.47)-(4.48) ensure vehicle energy stays within permissible limits, also represented as a cumulative function expression with negative impact for travel between locations and a positive impact for vehicle recharging. We note that the impact for energy replenishment tasks includes the negative contribution of the travel to the recharge station. Constraint (4.49) ensures that during a recharge task, the energy of the vehicle is set to its capacity; whenever a vehicle recharges, it does so fully. Constraint (4.51) ensures each recharge station is used at most n_f times across the fleet, where $F'(i)$ represents all dummy recharge stations associated with real recharge station $i \in F$. Constraint (4.50) enforces that the start and end depot instances for a vehicle be first and last in the sequence variable for that vehicle, while Constraints (4.52)-(4.56) provide the definitions of the interval and sequence variables.

$$\min \quad \sum_{k \in K} \Big(\alpha\ \textsc{Pres}(x_0^k) + \beta \sum_{i \in V'_{N+1}} \textsc{Pres}(x_i^k) \cdot d_{\textsc{Prev}_{\pi^k}(i),i} \Big) \qquad (\mathbb{CP}^{G'\text{-}AR}_{EVRPTW})$$

$$\text{s.t.} \quad \textsc{Alternative}(\bar{x}_i, \{x_i^1, \ldots, x_i^{|V|}\}) \qquad \forall i \in V, \qquad (4.43)$$

$$\textsc{NoOverlap}(\pi^k, \{t_{ij} : (i,j) \in A\}) \qquad \forall k \in K, \qquad (4.44)$$

$$\mathcal{C}^k = \textsc{StepAtStart}(x_0^k, C) - \sum\nolimits_{i \in V} \textsc{StepAtStart}(x_i^k, q_i) \qquad \forall k \in K, \qquad (4.45)$$

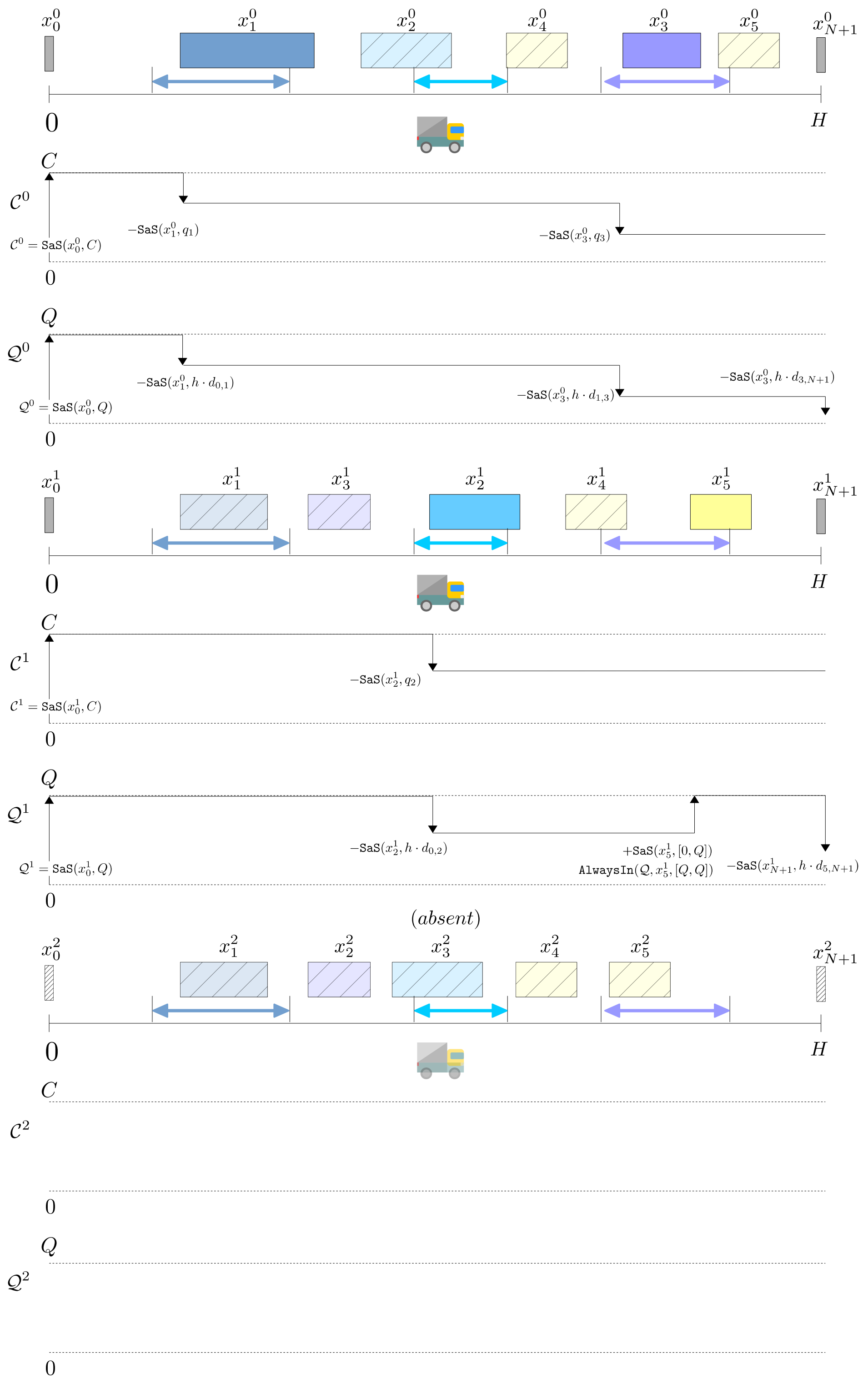

x_0^0
x_1^0
x_2^0
x_4^0
x_3^0
x_5^0
x_{N+1}^0
0
H
C
$\mathcal{C}^0$
$-\texttt{SaS}(x_1^0, q_1)$
$-\texttt{SaS}(x_3^0, q_3)$
$\mathcal{C}^0 = \texttt{SaS}(x_0^0, C)$
0
Q
$\mathcal{Q}^0$
$-\texttt{SaS}(x_1^0, h \cdot d_{0,1})$
$-\texttt{SaS}(x_3^0, h \cdot d_{1,3})$
$-\texttt{SaS}(x_3^0, h \cdot d_{3,N+1})$
$\mathcal{Q}^0 = \texttt{SaS}(x_0^0, Q)$
0
x_0^1
x_1^1
x_3^1
x_2^1
x_4^1
x_5^1
x_{N+1}^1
0
H
C
$\mathcal{C}^1$
$-\texttt{SaS}(x_2^1, q_2)$
$\mathcal{C}^1 = \texttt{SaS}(x_0^1, C)$
0
Q
$\mathcal{Q}^1$
$-\texttt{SaS}(x_2^1, h \cdot d_{0,2})$
$+\texttt{SaS}(x_5^1, [0, Q])$
$\texttt{AlwaysIn}(\mathcal{Q}, x_5^1, [Q, Q])$
$-\texttt{SaS}(x_{N+1}^1, h \cdot d_{5,N+1})$
$\mathcal{Q}^1 = \texttt{SaS}(x_0^1, Q)$
0
(*absent*)
x_0^2
x_1^2
x_2^2
x_3^2
x_4^2
x_5^2
x_{N+1}^2
0
H
C
$\mathcal{C}^2$
0
Q
$\mathcal{Q}^2$
0

Figure 4.3: Alternative resource model for problem with $|V| = 3$ and a single recharge station, $|F| = 1$, with $n_f = 2$ (such that $|F'| = 2$). A horizon is created for each vehicle where $|K| = |V|$ and time windows are instantiated. All customer and recharge tasks are optional. Cumulative function expressions represent vehicle load and energy level (where notation SaS corresponds to StepAtStart in models). Vehicles are represented explicitly. The last vehicle is not used in this particular solution (set as absent), but is included in the visualization as it was initialized in the model; we note that optimal solution could have used all three vehicles.

$$\textsc{AlwaysIn}(\mathcal{C}^k, [0,H], [0,C]) \qquad \forall k \in K, \tag{4.46}$$

$$\begin{aligned}\mathcal{Q}^k = {} & \textsc{StepAtStart}(x_0^k, Q) \\ & - \sum\nolimits_{i \in V'_{N+1}} \textsc{StepAtStart}(x_i^k, p \cdot d_{\textsc{Prev}_{\pi^k}(i),i}) \\ & + \sum\nolimits_{i \in F'} \textsc{StepAtStart}(x_i^k, g \cdot \textsc{Length}(x_i^k)) \qquad \forall k \in K,\end{aligned} \tag{4.47}$$

$$\textsc{AlwaysIn}(\mathcal{Q}^k, [0,H], [0,Q]) \qquad \forall k \in K, \tag{4.48}$$

$$\textsc{AlwaysIn}(\mathcal{Q}^k, x_i^k, [Q,Q]) \qquad \forall i \in F', k \in K, \tag{4.49}$$

$$\textsc{First}(\pi^k, x_0^k),\ \textsc{Last}(\pi^k, x_{N+1}^k) \qquad \forall k \in K, \tag{4.50}$$

$$\sum_{k \in K} \sum_{j \in F'(i)} \textsc{Pres}(x_j^k) \le n_f \qquad \forall i \in F, \tag{4.51}$$

$$x_i^k \ : \ \textsc{optIntervalVar}([0, Q \cdot g^{-1}], [0,H]) \qquad \forall i \in F', k \in K, \tag{4.52}$$

$$x_i^k \ : \ \textsc{optIntervalVar}(s_i, [e_i, l_i]) \qquad \forall i \in V, k \in K, \tag{4.53}$$

$$\bar{x}_i \ : \ \textsc{intervalVar}(s_i, [e_i, l_i]) \qquad \forall i \in V, \tag{4.54}$$

$$x_0^k \ : \ \textsc{intervalVar}(0, [0,0]), x_{N+1}^k \ : \ \textsc{intervalVar}(0, [H,H]) \qquad \forall k \in K, \tag{4.55}$$

$$\pi^k \ : \ \textsc{sequenceVar}(\{x_0^k, \ldots, x_{N+1}^k\}) \qquad \forall k \in K. \tag{4.56}$$

As the alternative resource formulation explicitly represents each vehicle, the number of variables can become unwieldy for larger problems. Specifically, the formulation has $|V| + |K| \cdot |V'_{0,N+1}|$ interval variables, $|K|$ sequence variables, and $2|K|$ cumulative function expression variables.

Cumulative Resource Constraint. Similar to a previous CP formulation for patient transportation [35], we strengthen the baseline formulation with a cumulative resource constraint. We define an auxiliary integer variable representing the number of vehicles in the fleet, $z = \sum_{k \in K} \textsc{Pres}(x_0^k)$. The cumulative constraint is then $\textsc{Cumulative}(\bar{x} \cup \{x_i : i \in F'\}, z)$, which expresses that at any time point in the horizon, the total number of customer interval variables, $\bar{x}$, and present recharge interval variables is bounded by the number of vehicles in the fleet.

Symmetry Breaking Constraints. Due to the large number of homogeneous vehicles, the use of symmetry breaking can be effective. We introduce a constraint of the form $\textsc{Pres}(x_0^k) \ge \textsc{Pres}(x_0^{k+1})$, ensuring vehicles are used in a lexicographic order. We then specify that if a vehicle depot task is not present, it cannot be assigned any other activities via $\textsc{Pres}(x_0^k) \ge \textsc{Pres}(x_i^k), \forall i \in V'_{N+1}$.

Single Resource Model. Our second CP model, inspired by the modeling efficiency of the two-index MILP for homogeneous vehicles in $\mathbb{MILP}^{G'}_{EVRPTW}$, utilizes a single resource transformation to significantly reduce the number of variables. The transformation represents the problem as an interval variable sequence over an augmented horizon and, like the MILP, does not explicitly represent the vehicles. This modeling strategy, while common in MILP models for VRPs, has been rarely used in CP. In previous work on joint vehicle and crew routing, a similar strategy was used to artificially join the end of one route to the beginning of another when using the Circuit global constraint [131, 128], which prevents the formation of subtours among a set of integer variables. However, to our knowledge, the single resource transformation has never been proposed for scheduling-based CP models involving interval, sequence and cumulative function expression variables. The transformation using these formalisms is challenging as the modeling paradigm does not permit the "resetting" of time; we detail how this is accomplished in the remainder of this section.

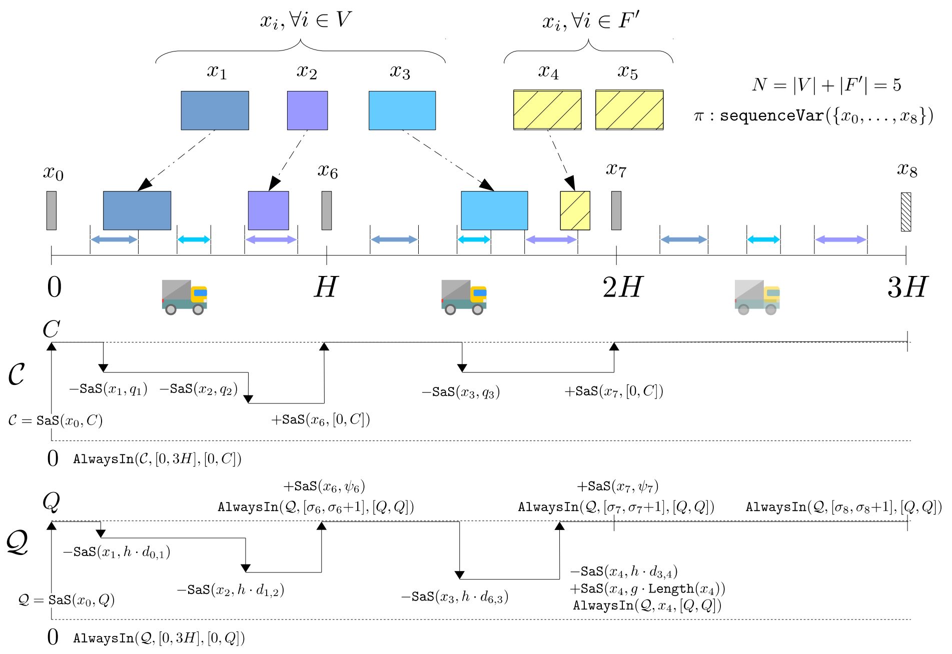

$x_i, \forall i \in V$
$x_i, \forall i \in F'$
x_1
x_2
x_3
x_4
x_5
$N = |V| + |F'| = 5$
π : sequenceVar($\{x_0, \dots, x_8\}$)
x_0
x_6
x_7
x_8
0
H
$2H$
$3H$
C
$\mathcal{C}$
$-$SaS(x_1, q_1)
$-$SaS(x_2, q_2)
$-$SaS(x_3, q_3)
$+$SaS($x_7, [0, C]$)
$\mathcal{C} =$ SaS(x_0, C)
$+$SaS($x_6, [0, C]$)
0 AlwaysIn($\mathcal{C}, [0, 3H], [0, C]$)
$+$SaS(x_6, ψ_6)
AlwaysIn($\mathcal{Q}, [\sigma_6, \sigma_6{+}1], [Q, Q]$)
$+$SaS(x_7, ψ_7)
AlwaysIn($\mathcal{Q}, [\sigma_7, \sigma_7{+}1], [Q, Q]$)
AlwaysIn($\mathcal{Q}, [\sigma_8, \sigma_8{+}1], [Q, Q]$)
Q
$\mathcal{Q}$
$-$SaS($x_1, h \cdot d_{0,1}$)
$-$SaS($x_2, h \cdot d_{1,2}$)
$-$SaS($x_3, h \cdot d_{6,3}$)
$-$SaS($x_4, h \cdot d_{3,4}$)
$+$SaS($x_4, g \cdot$ Length(x_4))
AlwaysIn($\mathcal{Q}, x_4, [Q, Q]$)
$\mathcal{Q} =$ SaS(x_0, Q)
0 AlwaysIn($\mathcal{Q}, [0, 3H], [0, Q]$)

Figure 4.4: Single resource transformation for problem with $|V| = 3$ and a single recharge station, $|F| = 1$, with $n_f = 2$ (such that $|F'| = 2$). A horizon segment is created for each potential vehicle and time windows are duplicated. All customer tasks are mandatory with disjoint start time domains and energy tasks (optional) have start time domain of $[0, 3H]$. A cumulative function expression represents vehicle load and energy level (where notation SAS corresponds to STEPATSTART in models). Vehicle assignments can then be inferred by the start times of the tasks themselves. The last horizon segment is not used (set as absent), but is included in the visualization as it was initialized in the model; we note that the optimal solution could have used all three horizons.

We visualize the single resource model in Figure 4.4. The transformation augments the problem horizon from H to $|V| \cdot H$, generating a horizon for each potential vehicle used. In addition to the start and end depot instances, v_0 and v_{N+1}, we introduce a set of auxiliary depot instances, $\Omega = \{v_{N+2}, \dots, v_{N+|V|}\}$, representing the end depots of the additional horizon segments. We define the notation $V'_{0,N+1,\Omega} = V'_{0,N+1} \cup \Omega$ and undirected arcs $A' = \{(i,j) | i, j \in V'_{0,N+1,\Omega}, i \neq j\}$. Similarly, we denote $\Omega_{N+1} = v_{N+1} \cup \Omega$ and $\Omega_{0,N+1} = \{v_0, v_{N+1}\} \cup \Omega$. A depot instance, represented as an interval variable, x_i, is assigned with null duration for $i \in \Omega_{0,N+1}$. These interval variables have start time σ_i, such that $\sigma_0 = 0, \sigma_{N+1} = H, \sigma_{N+2} = 2H$, and so forth. We then create a mandatory interval variable, x_i, for each customer request, $i \in V$, and an optional interval variable for each recharge station instance in the augmented set, $i \in F'$. Our model uses a single sequence variable, π, defined over the set of all interval variables, and a single cumulative function expression to model vehicle load, $\mathcal{C}$, with another for energy level, $\mathcal{Q}$. Additionally, at the start of each end depot instance, $i \in \Omega_{N+1}$, the state of the vehicle must be reset to initial conditions. Thus, the cumulative function expressions for vehicle load and energy have auxiliary positive impacts bringing them to their maximum capacity states. The start time domain for customer requests, $i \in V$, becomes a set of disjoint time windows, where each request time window is replicated over each of the horizon segments. The start domain for customer requests is the entire augmented horizon and the disjoint time windows are enforced with constraints. The start time domain for recharge tasks, $i \in F'$, becomes the entire augmented horizon.

$$
\begin{aligned}
\min \quad & \alpha \sum_{i \in \Omega_{N+1}} \textsc{Pres}(x_i) + \beta \sum_{i \in V'_{\Omega,N+1}} \textsc{Pres}(x_i) \cdot d_{\textsc{Prev}_\pi(i),i} && (\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}) \\
\text{s.t.} \quad & \textsc{NoOverlap}(\pi, \{t_{ij} : (i,j) \in A'\}) && (4.57) \\
& \textsc{ForbidExtent}(x_i, \phi_i) \qquad \forall i \in V, && (4.58) \\
& \mathcal{C} = \textsc{StepAtStart}(x_0, C) \\
& \qquad - \textstyle\sum_{i \in V} \textsc{StepAtStart}(x_i, q_i) \\
& \qquad + \textstyle\sum_{i \in \Omega_{N+1}} \textsc{StepAtStart}(x_i, [0, C]) && (4.59) \\
& \textsc{AlwaysIn}(\mathcal{C}, [0, |V| \cdot H], [0, C]) && (4.60) \\
& \mathcal{Q} = \textsc{StepAtStart}(x_0, Q) \\
& \quad - \textstyle\sum_{i \in V'} \textsc{StepAtStart}(x_i, p \cdot d_{\textsc{Prev}_\pi(i),i}) \\
& \quad + \textstyle\sum_{i \in F'} \textsc{StepAtStart}(x_i, g \cdot \textsc{Length}(x_i)) \\
& \quad + \sum_{i \in \Omega_{N+1}} \textsc{StepAtStart}(x_i, \psi_i) && (4.61) \\
& 0 \le \psi_i \le Q - p \cdot d_{\textsc{Prev}_\pi(i),i} \qquad \forall i \in \Omega_{N+1}, && (4.62) \\
& \textsc{AlwaysIn}(\mathcal{Q}, [0, |V| \cdot H], [0, Q]) && (4.63) \\
& \textsc{AlwaysIn}(\mathcal{Q}, x_i, [Q, Q]) \qquad \forall i \in F', && (4.64) \\
& \textsc{AlwaysIn}(\mathcal{Q}, [\sigma_i, \sigma_i + 1], [Q, Q]) \qquad \forall i \in \Omega_{N+1}, && (4.65) \\
& \textsc{First}(\pi, x_0) && (4.66) \\
& x_i \ : \ \textsc{optIntervalVar}([0, Q \cdot g^{-1}], [0, H \cdot |V|]) \qquad \forall i \in F', && (4.67) \\
& x_i \ : \ \textsc{intervalVar}(s_i, [0, H \cdot |V|]) \qquad \forall i \in V, && (4.68) \\
& x_i \ : \ \textsc{intervalVar}(0, \sigma_i) \qquad \forall i \in \Omega_{0,N+1}, && (4.69) \\
& \pi \ : \ \textsc{sequenceVar}(\{x_0, \ldots, x_{N+|V|}\}). && (4.70)
\end{aligned}
$$

Our single resource CP model is detailed by Eqns. (4.57)-(4.70). Objective $(\mathbb{CP}^{G'\text{-}SR}_{EVRPTW})$ is our fleet and distance minimization objective function. Constraint (4.57) enforces temporal feasibility of the interval variable sequence, π, including travel times. To make sure customers are serviced during a valid time window, we use Constraint (4.58), where $\phi_i = \{0, \ldots, |V| \cdot H\} \setminus \bigcup_{\delta \in \{0,\ldots,|V|\}} \{\delta H + e_i, \ldots, \delta H + l_i + s_i\}$. The ForbidExtent constraint prevents an interval variable, x_i, from being scheduled during any time point within the augmented horizon that is not also within one of the disjoint time windows. We discuss a number of alternatives for modeling disjoint time windows with CP in the next section. Constraints (4.59)-(4.60) ensure vehicle load feasibility. Constraints (4.61)-(4.63) ensure vehicle energy level feasibility while Constraints (4.64)-(4.65) dictate any present recharges, as well as horizon end tasks, must charge the vehicle to full energy level. To ensure the resetting of energy level at the end of each horizon, $i \in \Omega_{N+1}$, we use a positive impact StepAtStart with magnitude in $[0, Q - p \cdot d_{\textsc{Prev}_\pi(i),i}]$ expressed by Constraint (4.62), and the AlwaysIn expressed by Constraint (4.65). These components are illustrated in Figure 4.4. The position of the start depot in the interval variable sequence, π, is expressed through Constraint (4.66) and Constraints (4.67)-(4.70) identify variable domains.

Optional Horizon Segments. Initially, a horizon segment must be created for each potential vehicle, recalling that the upper bound used is the number of customer requests, $|V|$. This augmented horizon significantly increases the start time domain of the recharge tasks, even though most high quality solutions only use a small fraction of the vehicles allotted. To improve upon this, we develop a technique, similar to the symmetry breaking in the alternate resource model, where horizon segments can be set absent. First, we set all auxiliary end depot instances, $x_i, \forall i \in \Omega$, as optional interval variables. Next, we introduce an integer variable for each of the end depot instances, w_i, and constrain its value to be the start time of the interval variable (0 if the variable is set as absent), via $\textsc{StartOf}(x_i) = w_i, \forall i \in \Omega_{N+1}$. We then constrain the end time of the set of customer and recharge visit tasks to be bounded by the maximum w_i value, $\textsc{EndOf}(x_j) \leq \max_{i \in \Omega_{N+1}} w_i, \forall j \in V'$. Finally, we impose an ordering on the present depot instances using: $\textsc{Pres}(x_i) \geq \textsc{Pres}(x_{i+1}), \forall i \in \Omega \setminus v_{N+|V|}$. Similar to the alternative resource model, we introduce an additional integer variable, z, representing the number of vehicles in the fleet and constrain it such that: $z = \sum_{i \in \Omega_{N+1}} \textsc{Pres}(x_i)$.

4.5.1.2 PDPTW-EV

In this section we present the CP formulations for PDPTW-EV. Conveniently, in contrast to the MILP approaches, these formulations do not require the introduction of any new variables; all of the required relationships can be expressed with new constraints over the existing set of variables.

Alternative Resource Model. For the alternative resource model, we introduce a vehicle for each pickup/delivery pair, such that $|K| = |P|$. Since the alternative resource model explicitly represents each of the vehicles, enforcing the precedence and route requirements for pickup and delivery is simple.

$$\begin{aligned}
\min \quad & \text{Objective } (\mathbb{CP}^{G'\text{-}AR}_{EVRPTW}) && && (\mathbb{CP}^{G'\text{-}AR}_{PDPTW\text{-}EV}) \\
\text{s.t.} \quad & \text{Constraints } (4.43)-(4.44), (4.46)-(4.56) && && \\
& \mathcal{C}^k = \sum_{i \in V} \textsc{StepAtStart}(x_i^k, q_i) && \forall k \in K, && (4.71) \\
& \textsc{Pres}(x_i^k) = \textsc{Pres}(x_{i+n}^k) && \forall i \in P, k \in K, && (4.72) \\
& \textsc{EndBeforeStart}(x_i^k, x_{i+n}^k, t_{i,i+n}) && \forall i \in P, k \in K. && (4.73)
\end{aligned}$$

Constraint (4.71) is an updated cargo constraint reflecting pickups and deliveries. Constraint (4.72) ensures that if vehicle $k \in K$ is assigned the pickup, it is also assigned the delivery. Constraint (4.73) then constrains the end of the pickup to be before the delivery via use of the EndBeforeStart constraint, which states that the start of interval variable x_{i+n}^k, representing the delivery for pickup i, must be at least $t_{i,i+n}$ time units after the end of interval variable x_i^k, the pickup.

Single Resource Model. For the single resource model, we model a horizon for each pickup/deliver pair such that $|\Omega_{N+1}| = |P|$. The single resource model, as it does not explicitly track each vehicle, requires a slightly different set of constraints.

$$\begin{aligned}
\min \quad & \text{Objective } (\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}) && (\mathbb{CP}^{G'\text{-}SR}_{PDPTW\text{-}EV}) \\
\text{s.t.} \quad & \text{Constraints } (4.57)-(4.58), (4.60)-(4.70) &&
\end{aligned}$$

$$\mathcal{C} = \sum\nolimits_{i \in V} \textsc{StepAtStart}(x_i, q_i) + \sum\nolimits_{i \in \Omega_{N+1}} \textsc{StepAtStart}(x_i, [0, C]), \tag{4.74}$$

$$\textsc{AlwaysIn}(\mathcal{C}, x_i, [0, 0]) \qquad \forall i \in \Omega_{N+1}, \tag{4.75}$$

$$\lfloor \frac{\textsc{Start}(x_i)}{H} \rfloor = \lfloor \frac{\textsc{Start}(x_{i+n})}{H} \rfloor \qquad \forall i \in P, \tag{4.76}$$

$$\textsc{EndBeforeStart}(x_i, x_{i+n}, t_{i,i+n}) \qquad \forall i \in P. \tag{4.77}$$

Constraint (4.74) is an updated cargo constraint reflecting pickups and deliveries. Constraint (4.75) ensures that cargo levels are reset to zero at the start of each horizon. Constraint (4.76) ensures that pickup and delivery pairs are assigned to the same horizon and, thus, the same vehicle. The floor operation is expressed with the FLOOR constraint, commonly available in CP solvers, rounding the expression within the constraint down to the nearest integer. Constraint (4.77) then ensures that the end of the pickup occurs before the associated delivery using a similar constraint to the alternative resource model.

4.5.2 Recharging Path Multigraph Reformulations

These CP models use the single resource transformation and are defined on the recharging path multigraph, $\mathcal{G}$. Using recharging paths results in CP formulations with no optional recharge tasks, however, the selection of transition distances and energy consumption between customer visits (or pickups/deliveries) now becomes a decision to be optimized over. We present both alternative and single resource models based on recharging paths.

4.5.2.1 EVRPTW

In this section we present both alternative and single resource recharging path-based CP formulations for the EVRPTW.

Alternative Resource Model. The recharging path alternative resource model is defined by Eqns. (4.79) to (4.95). This formulation is similar to its augmented graph counterpart, $\mathbb{CP}^{G'\text{-}AR}_{EVRPTW}$, however, vehicle energy is formulated differently to account for the presence of recharge paths. We elect to use integer energy tracking variables instead of cumulative function expressions as the recharging path formulation does not reason over the specific fluctuations of energy levels. We define the set of arcs connecting all vertices in the multigraph with the direct links (i.e., no recharge visits) as $\hat{A}^0$. In the model, integer variable y_i^k indicates the energy of vehicle $k \in K$ at the start of task $i \in V_{0,N+1}$ and integer variable Y_i^k the amount of energy recuperated by vehicle $k \in K$ from the visit before i to visit $i \in V_{N+1}$. Recall that $\hat{\gamma}_{ij}^h$ and $\tilde{\gamma}_{ij}^h$ represent the energy required to travel from i to the first recharge station and from the last recharge station to j on path $h \in \mathcal{H}(i,j)$, respectively. Recall that Q_{ij}^h and d_{ij}^h represent the total energy replenished and the total distance traveled on recharge path h, respectively.

In order to reduce the excessive use of indexing decision variables with other decision variables, which was found to perform poorly in initial experiments, we elect to use TABLE constraints to capture the selection of recharging paths in the CP models. A TABLE constraint specifies the list of solutions (tuples) that are permitted to be assigned to a vector of decision variables. In our case, we enforce a TABLE constraint for each pair of vertices $(i,j) \in \hat{A}^0, i \neq j$. For a given pair of vertices, (i,j), the vector of decision variables is given by: $\Upsilon_{i,j} := (\underline{\hat{\gamma}_{i,j}}, \underline{\tilde{\gamma}_{i,j}}, \underline{Q_{i,j}}, \underline{d_{i,j}})$, where the naming of the variables is chosen to

Table 4.1: Recharging paths in CP: Table constraint example for vertex pair (i, j) with three possible paths including the direct path. Each row represents a valid assignment to the decision variables. Assignments that take values from different rows are not permitted by the constraint.

$h \in \mathcal{H}(i,j)$	$\underline{\hat{\gamma}_{i,j}}$	$\underline{\tilde{\gamma}_{i,j}}$	$\underline{Q_{i,j}}$	$\underline{d_{i,j}}$
h = 0	$\hat{\gamma}^0_{i,j}$	$\tilde{\gamma}^0_{i,j}$	$Q^0_{i,j}$	$d^0_{i,j}$
h = 1	$\hat{\gamma}^1_{i,j}$	$\tilde{\gamma}^1_{i,j}$	$Q^1_{i,j}$	$d^1_{i,j}$
h = 2	$\hat{\gamma}^2_{i,j}$	$\tilde{\gamma}^2_{i,j}$	$Q^2_{i,j}$	$d^2_{i,j}$

coincide with their parameter counterparts (i.e., $\underline{d_{i,j}}$ is a variable representing the distance traveled from i to j), but an underline is used to clarify that they are decision variables for a given pair of vertices and not defined for a path. For instance, the desired domain of variable $\underline{\hat{\gamma}_{i,j}}$ would be the set of parameters, such that: $\underline{\hat{\gamma}_{i,j}} \in \bigcup_{h\in\mathcal{H}(i,j)}\{\hat{\gamma}^h_{i,j}\}$. An example of a posted Table constraint for vertex pair (i, j) is given by Table 4.1. In the context of this example, the constraint $\text{Table}(\Upsilon_{i,j})$ effectively requires that the decision variables take on values specified by a row (i.e., variables taking on the values of the second row corresponds to a path selection of $h = 1$).

$$\min \quad \sum_{k\in K}\left(\alpha\ \text{Pres}(x_0^k) + \beta \sum_{i\in V_{N+1}} \text{Pres}(x_i^k)\cdot \underline{d_{\text{Prev}_{\pi^k}(i),i}}\right) \qquad (\mathbb{CP}^{\mathcal{G}\text{-}AR}_{EVRPTW})$$

$$\begin{aligned}
\text{s.t.}\quad & \text{Alternative}(\bar{x}_i, \{x_i^1, \dots, x_i^{|V|}\}) && \forall i \in V, && (4.78)\\
& \text{NoOverlap}(\pi^k, \{t_{ij} : (i,j) \in \hat{A}^0\}) && \forall k \in K, && (4.79)\\
& \mathcal{C}^k = \text{StepAtStart}(x_0^k, C) - \textstyle\sum_{i\in V} \text{StepAtStart}(x_i^k, q_i) && \forall k \in K, && (4.80)\\
& \text{AlwaysIn}(\mathcal{C}^k, [0,H], [0,C]) && \forall k \in K, && (4.81)\\
& \text{Table}(\Upsilon_{i,j}) && \forall (i,j) \in \hat{A}^0, i \neq j, && (4.82)\\
& \text{Start}(x_i^k) \geq \text{Pres}(x_i^k)\big(\text{End}(x^k_{\text{Prev}_{\pi^k}(i)}) \\
& \quad + (\underline{d_{\text{Prev}_{\pi^k}(i),i}}/\zeta) + (Y_i^k/g)\big) && \forall i \in V_{N+1}, k \in K, && (4.83)\\
& Y_i^k \leq \text{Pres}(x_i^k)\cdot \underline{Q_{\text{Prev}_{\pi^k}(i),i}} && \forall i \in V_{N+1}, k \in K, && (4.84)\\
& \underline{Q_{\text{Prev}^k_{\pi}(i),i}} > 0 \rightarrow Y_i^k \geq \text{Pres}(x_i^k)\Big((Q - y^k_{\text{Prev}_{\pi^k}(i)}) \\
& \quad + p\cdot \underline{d_{\text{Prev}_{\pi^k}(i),i}} - \underline{\tilde{\gamma}_{\text{Prev}_{\pi^k}(i),i}}\Big) && \forall i \in V_{N+1}, k \in K, && (4.85)\\
& y_{\text{Prev}_\pi(i)} \geq \text{Pres}(x_i)\cdot \hat{\gamma}^{p_i}_{\text{Prev}_\pi(i),i} && \forall i \in V_{\Omega,N+1}, && (4.86)\\
& y_i^k \leq Q - \underline{\tilde{\gamma}_{\text{Prev}_{\pi^k}(i),i}} && \forall i \in V, k \in K, && (4.87)\\
& y_i^k \leq y^k_{\text{Prev}_{\pi^k}(i)} - p\cdot \underline{d_{\text{Prev}_{\pi^k}(i),i}} + Y_i^k && \forall i \in V, k \in K, && (4.88)\\
& \text{First}(\pi^k, x_0^k),\ \text{Last}(\pi^k, x^k_{N+1}) && && (4.89)\\
& \bar{x}_i\ :\ \text{intervalVar}(s_i, [e_i, \ell_i]) && \forall i \in V, && (4.90)\\
& x_i^k\ :\ \text{optIntervalVar}(s_i, [e_i, \ell_i]) && \forall i \in V, k \in K, && (4.91)\\
& x_0^k\ :\ \text{intervalVar}(0, [0,0]), x^k_{N+1}\ :\ \text{intervalVar}(0, [H,H]) && \forall k \in K && (4.92)\\
& \pi^k\ :\ \text{sequenceVar}(\{x_0^k, \dots, x^k_{N+1}\}) && \forall k \in K, && (4.93)\\
& y_i^k \geq 0 && \forall i \in V_{0,N+1}, k \in K, && (4.94)
\end{aligned}$$

$$Y_i^k \geq 0 \qquad \forall i \in V_0, k \in K, \quad (4.95)$$

$$\underline{\hat{\gamma}_{i,j}}, \underline{\tilde{\gamma}_{i,j}}, \underline{Q_{i,j}}, \underline{d_{i,j}} \geq 0 \qquad \forall (i,j) \in \hat{A}^0, i \neq j. \quad (4.96)$$

The model uses our fleet and distance minimization objective function, where distance variable $\underline{d_{\textsc{Prev}_{\pi^k}(i),i}}$ returns the distance from the visit previous to i. Constraint (4.78) ensures that a customer task is assigned to exactly one vehicle through the use of the Alternative constraint. Constraint (4.79) enforces temporal feasibility of the interval variable sequence, π^k, including travel times over the set of direct arcs, $\hat{A}^0$. Since this constraint only enforces travel times over direct arcs, it is a redundant constraint, however, it boosts the performance of the solver. Constraints (4.80)-(4.81) express the required cargo constraints; each of these is the same as in the corresponding augmented graph model. Constraint (4.83) ensures that the temporal requirements for the selected recharging path are satisfied, including the time taken to travel and recharge along the selected path. In this case, transition time is expressed by dividing the distance variable by ζ, a parameter that represents vehicle speed. Constraints (4.84) and (4.85) define the bounds on the recharge variable Y_i^k, relative to the visit made previous to i. Constraint (4.85) is only activated if the selected path is not a direct path. Constraints (4.86) to (4.88) dictate the bounds on the remaining energy variables, y_i^k. Constraint (4.89) fixes the start and end tasks for each vehicle sequence, π^k, to be the instances of the depot. The remainder of the constraints, Constraints (4.90) through (4.96), dictate the domains of the decision variables in the formulation.

Single Resource Model. Next, we present a single resource recharging path CP formulation in Eqns. (4.97) through (4.114). The model leverages augmented horizons to reduce the number of redundant interval variables in a similar fashion to the augmented models previously presented. The model also uses the same decision variable vector, $\Upsilon_{i,j}$, to express the required Table constraint as done in the alternative resource model.

$$\min \quad \alpha \sum_{i \in \Omega_{N+1}} \textsc{Pres}(x_i) + \beta \sum_{i \in V_{\Omega,N+1}} \textsc{Pres}(x_i) \cdot \underline{d_{\textsc{Prev}_\pi(i),i}} \qquad (\mathbb{CP}^{\mathcal{G}\text{-}SR}_{EVRPTW})$$

$$\text{s.t.} \quad \textsc{NoOverlap}(\pi, \{t_{ij} : (i,j) \in \hat{A}^0\}) \qquad (4.97)$$

$$\textsc{ForbidExtent}(x_i, \phi_i) \qquad \forall i \in V, \quad (4.98)$$

$$\begin{aligned} \mathcal{C} = {} & \textsc{StepAtStart}(x_0, C) \\ & - \textstyle\sum_{i \in V} \textsc{StepAtStart}(x_i, q_i) \\ & + \textstyle\sum_{i \in \Omega_{N+1}} \textsc{StepAtStart}(x_i, [0, C]) \end{aligned} \qquad (4.99)$$

$$\textsc{AlwaysIn}(\mathcal{C}, [0, |V| \cdot H], [0, C]) \qquad (4.100)$$

$$\textsc{Table}(\Upsilon_{i,j}) \qquad \forall (i,j) \in \hat{A}^0, i \neq j, \quad (4.101)$$

$$\textsc{Start}(x_i) \geq \textsc{Pres}(x_i) \Big(\textsc{End}(x_{\textsc{Prev}_\pi(i)}) + (\underline{d_{\textsc{Prev}_\pi(i),i}}/\zeta) + (Y_i/g) \Big) \qquad \forall i \in V_{\Omega,N+1}, \quad (4.102)$$

$$Y_i \leq \textsc{Pres}(x_i) \cdot \underline{Q_{\textsc{Prev}_\pi(i),i}} \qquad \forall i \in V_{\Omega,N+1}, \quad (4.103)$$

$$\begin{aligned} \underline{Q_{\textsc{Prev}_\pi(i),i}} > 0 \rightarrow Y_i \geq \textsc{Pres}(x_i) \Big(& (Q - y_{\textsc{Prev}_\pi(i)}) \\ & + p \cdot \underline{d_{\textsc{Prev}_\pi(i),i}} - \underline{\tilde{\gamma}_{\textsc{Prev}_\pi(i),i}} \Big) \end{aligned} \qquad \forall i \in V_{\Omega,N+1}, \quad (4.104)$$

$$y_{\textsc{Prev}_\pi(i)} \geq \textsc{Pres}(x_i) \cdot \hat{\gamma}^{p_i}_{\textsc{Prev}_\pi(i),i} \qquad \forall i \in V_{\Omega,N+1}, \quad (4.105)$$

$$y_i \leq Q - \underline{\tilde{\gamma}_{\text{Prev}_\pi(i),i}} \qquad \forall i \in V, \quad (4.106)$$

$$y_i \leq y_{\text{Prev}_\pi(i)} - p \cdot \underline{d_{\text{Prev}_\pi(i),i}} + Y_i \qquad \forall i \in V, \quad (4.107)$$

$$\text{First}(\pi, x_0) \qquad (4.108)$$

$$x_i \ : \ \text{intervalVar}(s_i, [0, H \cdot |V|]) \qquad \forall i \in V, \quad (4.109)$$

$$x_i \ : \ \text{intervalVar}(0, \sigma_i) \qquad \forall i \in \Omega_{0,N+1}, \quad (4.110)$$

$$\pi \ : \ \text{sequenceVar}(\{x_0, \ldots, x_{N+|V|}\}), \qquad (4.111)$$

$$y_i \geq 0 \qquad \forall i \in V_{0,N+1}, \quad (4.112)$$

$$Y_i \geq 0 \qquad \forall i \in V_0, \quad (4.113)$$

$$\underline{\hat{\gamma}_{i,j}}, \underline{\tilde{\gamma}_{i,j}}, \underline{Q_{i,j}}, \underline{d_{i,j}} \geq 0 \qquad \forall (i,j) \in \hat{A}^0, i \neq j. \quad (4.114)$$

The objective function is the same as the alternative resource recharging path model with the inclusion of augmented horizon interval variables. Constraint (4.97) enforces temporal feasibility of the interval variable sequence, π, including travel times over the set of direct arcs, $\hat{A}^0$. Since this constraint only enforces travel times over direct arcs, it is a redundant constraint, however, it boosts the performance of the solver. While this auxiliary constraint does not enforce all of the temporal reasoning required, it was found to strengthen the model. Constraint (4.98) enforces the time windows of the customer visits and Constraints (4.99)-(4.100) the required cargo constraints; each of these is the same as in the corresponding augmented graph model. Constraint (4.102) ensures that the temporal requirements for the selected recharging path are satisfied, including the time taken to travel and recharge along the selected path. In this case, transition time is expressed by dividing the distance variable by ζ, a parameter that represents vehicle speed. Constraints (4.103) and (4.104) define the bounds on the recharge variable Y_i, relative to the visit made previous to i. Constraint (4.104) is only activated if the selected path is not a direct path. Constraints (4.105) to (4.107) dictate the bounds on the remaining energy variables, y_i. The remainder of the constraints, Constraints (4.108) through (4.114), dictate the domains of the decision variables in the formulation. Finally, we implement the same horizon optionality for this model that was implemented in the single resource model for the augmented graph formulation.

4.5.2.2 PDPTW-EV

In this section we present both alternative and single resource recharging path-based CP formulations for the PDPTW-EV.

Alternative Resource Model. The extension of the previously proposed $\mathbb{CP}^{\mathcal{G}\text{-}AR}_{EVRPTW}$ model to the PDPTW-EV involves the addition of two constraints in a similar fashion to the extension proposed for the augmented graph model.

$$\min \quad \text{Objective } (\mathbb{CP}^{\mathcal{G}\text{-}AR}_{EVRPTW}) \qquad (\mathbb{CP}^{\mathcal{G}\text{-}AR}_{PDPTW\text{-}EV})$$

$$\text{s.t.} \quad \text{Constraints } (4.79) - (4.79), (4.81) - (4.96)$$

$$\mathcal{C}^k = \sum\nolimits_{i \in V} \text{StepAtStart}(x_i^k, q_i) \qquad \forall k \in K, \quad (4.115)$$

$$\text{Pres}(x_i^k) = \text{Pres}(x_{i+n}^k) \qquad \forall i \in P, k \in K, \quad (4.116)$$

$$\textsc{EndBeforeStart}(x_i^k, x_{i+n}^k, t_{i,i+n}) \qquad \forall i \in P, k \in K. \tag{4.117}$$

Constraint (4.115) is an updated cargo constraint to reflect pickups and deliveries. Constraint (4.116) ensures that pickup and delivery pairs are assigned to the same vehicle. Constraint (4.117) then ensures that the end of the pickup occurs before the associated delivery.

Single Resource Model. The extension of the previously proposed $\mathbb{CP}^{\mathcal{G}\text{-}SR}_{EVRPTW}$ model to the PDPTW-EV involves the addition of two constraints in a similar fashion to the extension proposed for the augmented graph model.

$$\begin{aligned}
\min \quad & \text{Objective } (\mathbb{CP}^{\mathcal{G}\text{-}SR}_{EVRPTW}) && (\mathbb{CP}^{\mathcal{G}\text{-}SR}_{PDPTW\text{-}EV})\\
\text{s.t.} \quad & \text{Constraints } (4.97) - (4.98), (4.100) - (4.114) &&\\
& \mathcal{C} = \sum\nolimits_{i \in V} \textsc{StepAtStart}(x_i, q_i) + \sum\nolimits_{i \in \Omega_{N+1}} \textsc{StepAtStart}(x_i, [0, C]), && (4.118)\\
& \textsc{AlwaysIn}(\mathcal{C}, x_i, [0, 0]) \qquad \forall i \in \Omega_{N+1}, && (4.119)\\
& \lfloor \frac{\textsc{Start}(x_i)}{H} \rfloor = \lfloor \frac{\textsc{Start}(x_{i+n})}{H} \rfloor \qquad \forall i \in P, && (4.120)\\
& \textsc{EndBeforeStart}(x_i, x_{i+n}, t_{i,i+n}) \qquad \forall i \in P. && (4.121)
\end{aligned}$$

Constraint (4.118) is an updated cargo constraint to reflect pickups and deliveries. Constraint (4.119) ensures that cargo levels are reset to zero at the start of each horizon. Constraint (4.120) ensures that pickup and delivery pairs are assigned to the same horizon (i.e., vehicle) using the Floor operator. Constraint (4.121) then ensures that the end of the pickup occurs before the associated delivery.

4.6 Computational Analysis

Our empirical investigation of the proposed approaches explores a number of different objective functions:

i. Fleet distance minimization ($\alpha = 0, \beta = 1$).

ii. Fleet size minimization ($\alpha = 1, \beta = 0$).

iii. Fleet size minimization with distance minimization as a secondary objective ($\alpha = 1, \beta = \xi$), where ξ is a sufficiently small number to lexicographically order the objective components.

We reiterate that the intent of this work is to investigate the performance of off-the-shelf optimization models for electric vehicle routing problems; state-of-the-art results for EVRPTW distance minimization, for example, are found using sophisticated branch-price-and-cut techniques bolstered by customized labeling algorithms [52]. As has been noted before in the literature, the accurate implementation of sophisticated branch-price-and-cut techniques is a long and difficult task [62], whereas the monolithic approaches in this paper can be implemented using intuitive, high-level modeling languages such as OPL [202].

Table 4.2: Problem instances, EVRPTW experiments. Each value represents the number of instances for a given size/characteristic combination. $|V| \leq 15$ are small instances containing 5, 10, and 15 customers. Clustered, random, and mix refer to the geographical distribution of customer vertices. $|F|$ values are averages across the instances.

			Short Horizon			**Long Horizon**		
$\|V\|$	$\|F\|$	Total	Clustered	Random	Mix	Clustered	Random	Mix
$\leq$15	4.2	36	6	6	6	6	6	6
25	21	56	9	12	8	8	11	8
50	21	56	9	12	8	8	11	8

4.6.1 Experimental Setup

All optimization models are implemented in C++ and run on the Compute Canada Niagara computing cluster operated by SciNet (http://www.scinetpc.ca). The cluster runs the Linux CentOS 7 operating system and uses Skylake cores at 2.4 GHz. We use CP Optimizer for the CP models and CPLEX for the MILP model from the IBM ILOG CPLEX Optimization Studio version 12.9. All experiments are single-threaded with default inference settings. For the CP models, we prioritize the auxiliary z variable in the search, representing the number of vehicles used. Multigraph construction for the recharging path formulations is done in Python. A time limit of one hour is used for all experiments. For the recharging path multigraph formulations, this time limit includes the time required to construct recharging paths. This time is roughly 1.5 minutes for the largest problems, and thus is negligible overall.

Following the procedure outlined in previous work on the same instances [52], we transform floating point parameter values to integer values such that the problems are amenable to CP modeling, which typically reasons over integer domains. As with most integer transformations, the scaling involved in this process results in much larger variable domain ranges which can have a negative impact on CP approaches. Additionally, for the augmented graph formulations, we heuristically set the number of visits allowable to each recharge station as $n_f = \lceil |V| \cdot 0.2 \rceil$ (i.e., problems with five customers allow a single visit to each recharge station) for EVRPTW problems and $n_f = \lceil |V| \cdot 0.4 \rceil$ for PDPTW-EV problems (pickup and delivery problems typically required more visits to recharge stations).

The CP solver used for experiments, CP Optimizer, does not directly support the implementation of the StepAtStart expression with variable impact (e.g., as detailed in Constraint (4.47)). As such, an expression of the form $\textsc{StepAtStart}(var, impact)$, where *impact* is a variable, is implemented with the expression $\textsc{StepAtStart}(var, [LB, UB])$, where $LB = 0$ and $UB = Q$ for energy modeling, and the constraint $\textsc{HeightAtStart}(var, f) = impact$, which ensures the step impact of var on cumulative function f is *impact*.

4.6.2 EVRPTW

In this section we provide experimental results and analysis for the EVRPTW models. We seed the CP search with a naive solution that allocates each customer visit to its own vehicle. If a vehicle requires a recharge visit to service its request, our naive heuristic attempts to add a single recharge visit to ensure feasibility. If the vehicle requires more than one recharge visit, the heuristic does not provide a feasible solution and the search is initiated without warm-start.

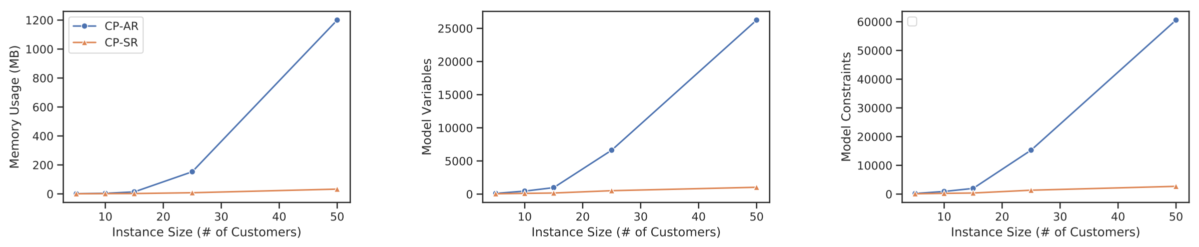


Figure 4.5: Augmented graph CP model size comparison: $\mathbb{CP}^{G'\text{-}AR}_{EVRPTW}$ (blue) vs. $\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}$ (orange). Comparison with respect to memory usage (before search), number of model variables, and number of model constraints.

We summarize the instances used to test our EVRPTW models, and then present results for the augmented graph formulations, followed by the recharging path multigraph reformulations.

4.6.2.1 Instances

We conduct our EVRPTW analysis on problem instances taken from the literature [183, 52]. Instances vary with respect to the number of customer and recharge station vertices, the length of the scheduling horizon (short and long) and the geographical distribution of the customer vertices (random, clustered, and a mixture of both). The benchmark utilized contains a total of 148 instances and is summarized in Table 4.2.

4.6.2.2 Experiment: Augmented Graph CP Model Size Comparison

Our first experiment involves an empirical comparison of the CP augmented graph model sizes. The results for the various metrics are summarized in Figure 4.5. The single resource transformation results in significantly smaller model sizes across all measures, particularly as the size of the problem instance increases. For the largest problem instances tested ($|V| = 50, |F| = 21$), the alternative resource model required over 1GB to store the resulting model, while the single resource transformation required less than 50MB. This represents over an order of magnitude difference for this problem size, and the discrepancy grows for even larger instances. A similar trend is shown for the number of model variables and constraints, suggesting that the single resource transformation is an efficient way to represent the EVRPTW defined over an augmented graph.

4.6.2.3 Experiment: Augmented Graph Formulations

We first present results for the augmented graph formulations. These models are particularly useful for practitioners as they do not require any pre-processing of the problem graph, and can be readily implemented in modern solvers. The results, after one hour of computation time, are illustrated in Table 4.3. The table details the performance of the models in terms of the number of feasible solutions found, the number of instances for which the model found the best solution at the end of the runtime, and the mean relative error (MRE). The MRE compares the best solution found by a given technique to the best lower bound found across all techniques; the results in the table take the average of this across all instances solved by the technique. For example, the formula used for the MRE of a given technique, Ψ, for problems with 50 customers ($P50$) is given by Equation (4.122):

Table 4.3: Experimental results, EVRPTW, augmented graph formulations. The best result of each column for each objective function in bold. One hour runtime limit.

	Short Horizon									Long Horizon								
	# Feasible			# Best			MRE (%)			# Feasible			# Best			MRE (%)		
Method	≤15	25	50	≤15	25	50	≤15	25	50	≤15	25	50	≤15	25	50	≤15	25	50
Fleet Distance																		
$\mathbb{MILP}^{G'}_{EVRPTW}$	**18**	**29**	**29**	**18**	4	3	**5.2**	56.4	64.6	**18**	**27**	**27**	**18**	**15**	12	**1.6**	**42.9**	**45.0**
$\mathbb{CP}^{G'\text{-}AR}_{EVRPTW}$	17	**29**	19	7	0	0	7.4	70.0	78.4	17	**27**	19	2	1	0	12.2	58.4	71.6
$\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}$	**18**	**29**	**29**	11	**25**	**26**	7.4	**52.0**	**55.4**	16	**27**	**27**	3	11	**15**	9.4	47.1	46.4
Fleet Size																		
$\mathbb{MILP}^{G'}_{EVRPTW}$	**18**	**29**	**29**	17	0	0	18.1	77.7	88.2	**18**	**27**	**27**	**16**	0	0	19.4	79.9	86.1
$\mathbb{CP}^{G'\text{-}AR}_{EVRPTW}$	17	**29**	21	15	1	0	18.2	81.7	89.5	17	**27**	23	12	12	6	19.6	52.5	73.4
$\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}$	**18**	**29**	**29**	**18**	**29**	**29**	**17.2**	**52.8**	**62.5**	16	**27**	**27**	**16**	**27**	**27**	**9.4**	**33.1**	**45.0**
Combined																		
$\mathbb{MILP}^{G'}_{EVRPTW}$	**18**	**29**	**29**	**17**	3	1	**19.1**	63.5	67.4	**18**	**27**	**27**	**17**	0	0	14.0	61.7	76.9
$\mathbb{CP}^{G'\text{-}AR}_{EVRPTW}$	17	**29**	17	6	0	0	25.2	80.9	90.7	17	**27**	20	3	0	0	30.0	52.7	82.9
$\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}$	**18**	**29**	**29**	12	**26**	**28**	20.0	**52.9**	**52.3**	16	**27**	**27**	3	**27**	**27**	**13.6**	**26.8**	**46.5**

$$MRE_{\Psi,P50} = \sum_{p \in \mathcal{P}} \frac{c(\Psi,p) - c^*(p)}{|\mathcal{P}| \cdot c^*(p)} \times 100 \tag{4.122}$$

where $\mathcal{P} \subseteq P50$ is the set of instances in $P50$ for which feasible solutions were found at the runtime limit, $c(\Psi,p)$ is the best solution found by approach Ψ for problem instance p, and $c^*(p)$ is the best known lower bound for instance p across all methods.

Full Runtime Analysis. The MILP approach, as seen in Table 4.3, offers the best performance[3] for 6/18 (33.3%) of the experiment classes, notably for smaller problems. Out of all the objective functions, the MILP model has the strongest performance on fleet distance minimization, often proving optimality for small instances. The technique exhibits weaker performance, relative to the single resource CP approach, for the remainder of the objective functions. We suspect this is due to a combination of a weaker linear relaxation for the non-distance minimization problems, as evident in significantly higher MRE values throughout the table, as well as the growth of the underlying problem graph due to the recharging node augmentation.

The alternative resource CP model, $\mathbb{CP}^{G'\text{-}AR}_{EVRPTW}$, offers the best performance for none of the experiment classes. This technique, by explicitly tracking each of the vehicles, suffers from large model sizes and less efficient search as problem instances get larger; maintaining interval variable sequences for each vehicle becomes more and more costly (supported by model size experiments illustrated in Figure 4.5).

The single resource approach, $\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}$, offers the best performance for 13/18 (72.2%) of the experiment classes, establishing itself as the strongest performer overall. It maintains the strongest

[3]With respect to the number of instances for which the technique found the best solution.

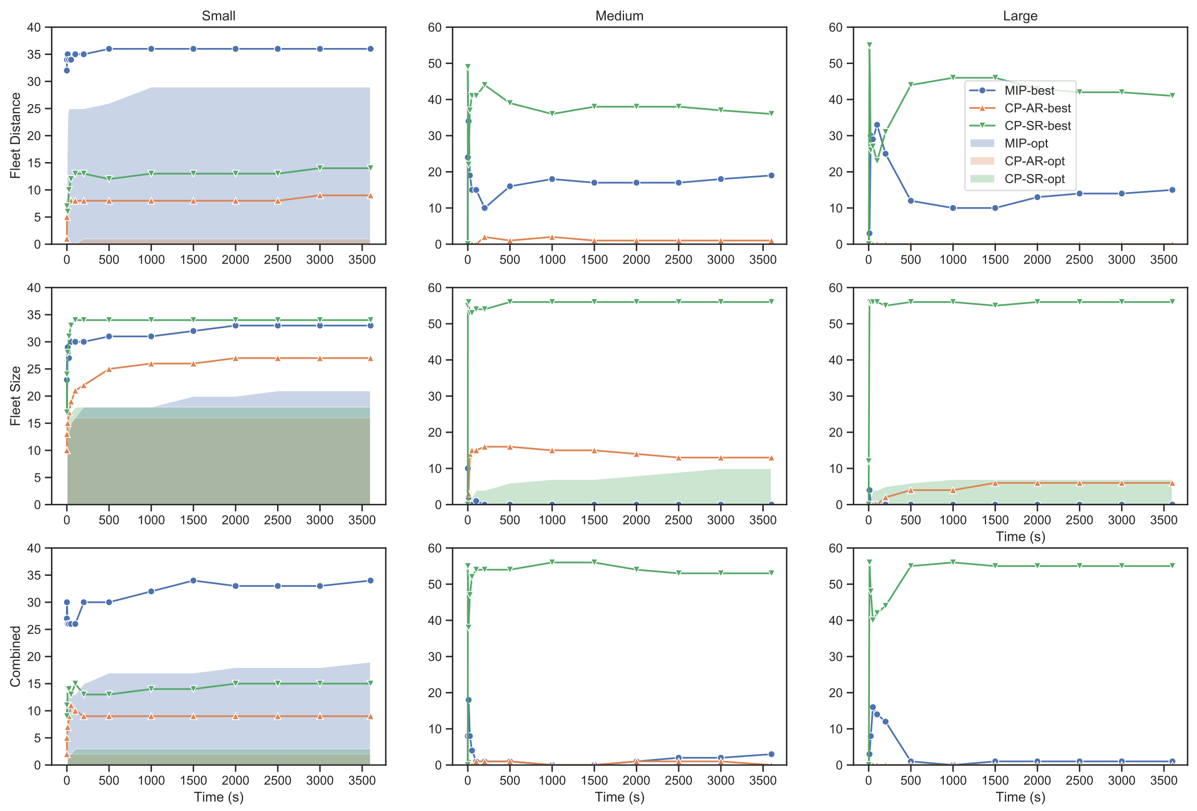


Figure 4.6: EVRPTW experimental runtime results for the augmented graph formulations. Each plot corresponds to a instance size and objective function. Count on the y-axis and time (in seconds) on the x-axis. Line graphs indicate the number of instances for which a method had the best result by a certain time. Area indicates the number of instances for which a method found and proved optimality.

performance for all of the medium and large problem experiments with the exception of medium long horizon fleet distance minimization. The approach benefits from the single resource transformation, leveraging the presence of symmetric vehicles to significantly reduce the number of variables and constraints. Indeed, during experimentation it was found that $\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}$ used significantly less memory than its alternative resource counterpart. The approach performs particularly strongly, relative to MILP, for fleet minimization and the combined objective function.

Time-Based Analysis. We visualize the performance of each of the methods over the course of their allotted runtime in Figure 4.6. In the visualization, the line graphs indicate the number of instances for which a given technique had the best solution at a given time point. The shaded area indicates the number of instances for which a given technique has found and proved the optimal solution up to a given time point. As is clear from the figure, the MILP approach is able to find and prove optimality for a large portion of the small problem instances across all objective functions (blue area). The CP approaches are, overall, weaker at finding and proving optimal solutions, with the exception of the single resource approach for medium and large fleet minimization problems. Overall, it would appear that the initial ten minutes (600 seconds) of the search are the most dynamic with respect to the relative performance of the algorithms; typically the technique that has found the highest number of best solutions at this point will retain this level for the remainder of the runtime (flat line plots). The plots clearly illustrate the dominance of the single resource CP approach, in terms of solution quality, for medium and large

Table 4.4: Experimental results, EVRPTW, recharging path multigraph reformulations. The best result of each column for each objective function in bold. One hour runtime limit.

	Short Horizon									Long Horizon								
	# Feasible			# Best			MRE (%)			# Feasible			# Best			MRE (%)		
Method	≤15	25	50	≤15	25	50	≤15	25	50	≤15	25	50	≤15	25	50	≤15	25	50
Fleet Distance																		
$\mathbb{MILP}^{\mathcal{G}}_{EVRPTW}$	**18**	**29**	**29**	**18**	**29**	**29**	**0.0**	**6.4**	**21.2**	**18**	**27**	**27**	**18**	**27**	**27**	**0.0**	**1.1**	**12.1**
$\mathbb{CP}^{\mathcal{G}\text{-}AR}_{EVRPTW}$	**18**	**29**	15	12	1	0	2.2	13.5	63.9	**18**	**27**	14	9	0	0	4.2	16.3	58.5
$\mathbb{CP}^{\mathcal{G}\text{-}SR}_{EVRPTW}$	17	**29**	23	10	0	0	1.5	11.1	51.9	16	26	25	2	0	0	9.1	13.8	43.6
Fleet Size																		
$\mathbb{MILP}^{\mathcal{G}}_{EVRPTW}$	**18**	**29**	**29**	**18**	19	**25**	**4.6**	29.4	**53.0**	**18**	**27**	**27**	**18**	17	8	5.6	31.4	58.0
$\mathbb{CP}^{\mathcal{G}\text{-}AR}_{EVRPTW}$	**18**	**29**	16	16	4	0	6.9	42.7	78.5	**18**	**27**	14	14	19	11	12.0	25.6	**43.8**
$\mathbb{CP}^{\mathcal{G}\text{-}SR}_{EVRPTW}$	**18**	**29**	24	17	**24**	6	8.8	**28.2**	65.3	16	26	25	16	**24**	**17**	**0.0**	**15.3**	53.3
Combined																		
$\mathbb{MILP}^{\mathcal{G}}_{EVRPTW}$	**18**	**29**	**29**	**15**	**27**	**29**	**6.5**	**24.6**	**41.0**	**18**	**27**	**27**	**15**	**17**	**21**	**2.8**	22.6	**45.1**
$\mathbb{CP}^{\mathcal{G}\text{-}AR}_{EVRPTW}$	**18**	**29**	12	10	0	0	9.3	46.4	78.8	**18**	**27**	14	3	6	1	30.1	30.9	64.2
$\mathbb{CP}^{\mathcal{G}\text{-}SR}_{EVRPTW}$	**18**	**29**	24	12	3	0	9.3	32.0	70.0	16	26	25	7	9	5	6.2	**17.7**	54.2

sized problems across all objective functions.

4.6.2.4 Experiment: Recharging Path Multigraph Reformulations

The results for the recharging path multigraph reformulations are illustrated in Table 4.4. All experiments include the time required to pre-process the graph.

Full Runtime Analysis. Benefiting from the recharging path construction and graph pre-processing, the $\mathbb{MILP}^{\mathcal{G}}_{EVRPTW}$ approach exhibits superior performance on 15/18 (83.3%) of the experiment classes, establishing itself as the strongest overall technique. It outperforms the other methods on distance minimization and combined objective problems by a wide margin, as well as some of the fleet size minimization classes. We note that the approach is able to find and prove optimal solutions, as evidenced by MRE values close to (or at) 0.0%, for small problems as well as medium-sized distance minimization problems. $\mathbb{MILP}^{\mathcal{G}}_{EVRPTW}$ appears to struggle with fleet minimization over long horizons, where the number of vehicles in the optimal solution is lower. Overall, it is evident that, should the practitioner have the expertise and flexibility to set-up the recharging paths, the multigraph MILP model is the most effective technique.

The alternative resource CP multigraph model, $\mathbb{CP}^{\mathcal{G}\text{-}AR}_{EVRPTW}$, is not the best approach for any of the experiment classes. While it provides competitive performance for fleet minimization over long horizons, it is significantly outperformed for all other objective functions. The recharging path pre-processing eliminates the need for optional recharging activities in this formulation, however, optional customer tasks remain present and contribute to large models as instances grow in size.

The single resource CP multigraph model, $\mathbb{CP}^{\mathcal{G}\text{-}SR}_{EVRPTW}$, is the best approach for only 3/18 (16.7%) of

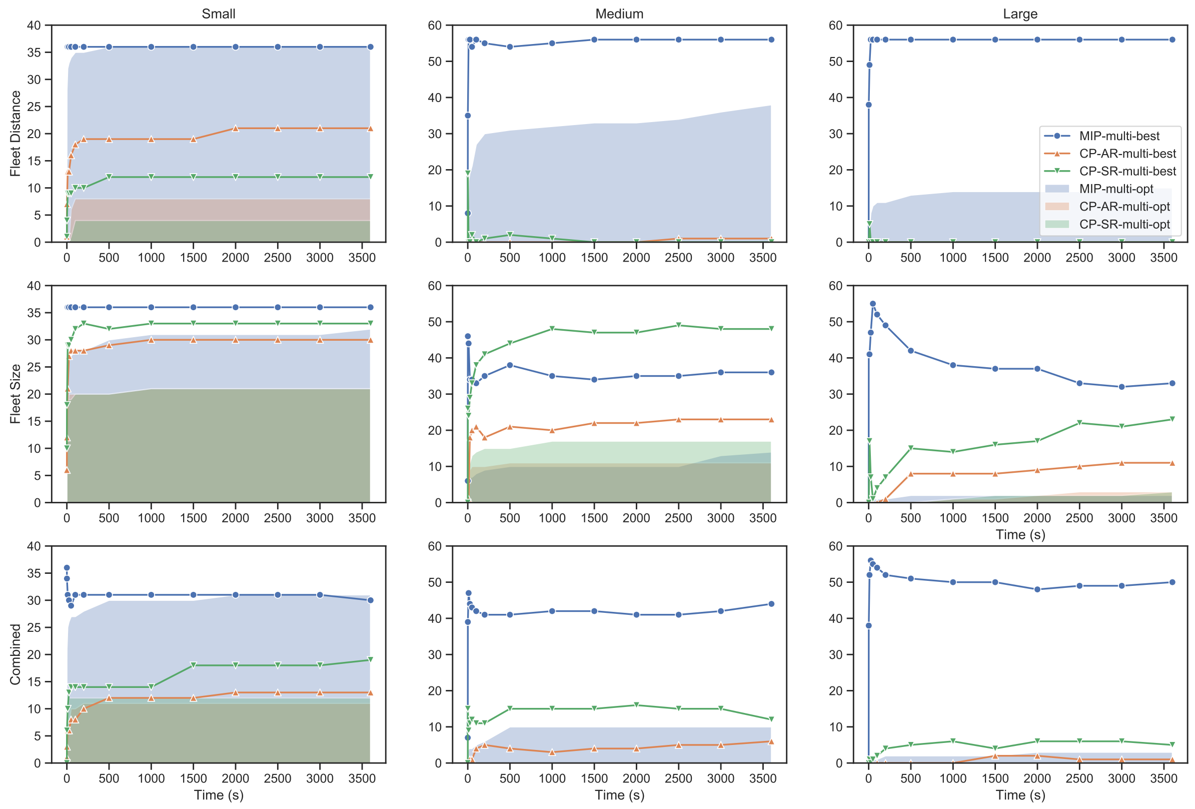


Figure 4.7: EVRPTW experimental runtime results for the recharging path multigraph reformulations. Each plot corresponds to a instance size and objective function. Count on the y-axis and time (in seconds) on the x-axis. Line graphs indicate the number of instances for which a method had the best result by a certain time. Area indicates the number of instances for which a method found and proved optimality.

the experiment classes. While it outperforms its alternative resource counterpart across all experiment classes, it is only able to outperform the MILP-based approach significantly for medium-sized fleet size minimization problems over short horizons, and for medium and large fleet minimization problems over long horizons, where the performance discrepancy is strongest in favour of CP for large problems.

Time-Based Analysis. An analysis of the multigraph models over the course of their runtime is illustrated in Figure 4.7. The plots do not include the multigraph construction time. For the largest problems, multigraph construction takes up to two minutes, suggesting that applications that need near-instant solutions would benefit more from the augmented graph models. As is evident in the figure, there are few changes in relative performance through the runtime; the technique that is able to produce the best solution within the first five minutes or so remains dominant for the full hour. The exception to this trend is for large fleet minimization problems where the single resource CP model seems to gradually catch-up to MILP. The MILP approach is able to find and prove optimality for all of the small instances under a fleet distance minimization objective function (as depicted by the blue shaded area), as well as for many of the small instances with the other objective functions. The MILP technique is also able to prove optimality for a fair number of medium-sized distance minimization problems. The CP approaches are able to prove optimality for a number of fleet size minimization and combined objective functions, albeit fewer than the MILP. The time-based visualization reflects the single resource CP model's dominance for

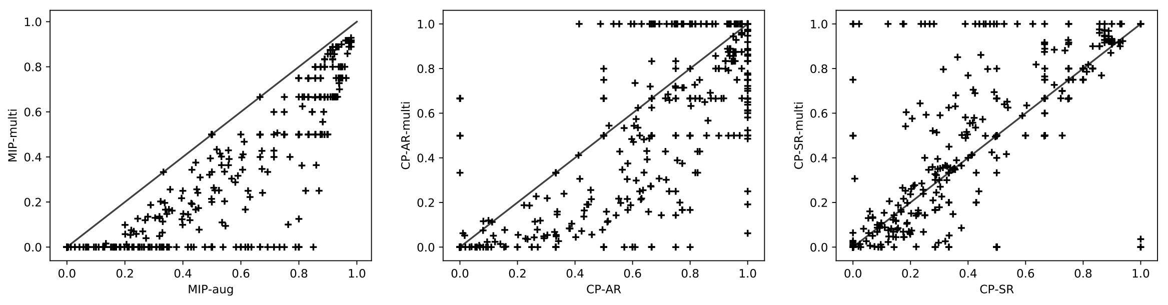


Figure 4.8: EVRPTW formulation comparison (augmented graph versus multigraph). MRE values for each instance plotted. Values on diagonal indicate equal performance for techniques in plot.

medium-sized fleet minimization problems, and its competitive performance on large fleet minimization problems.

4.6.2.5 Formulation Comparison

We provide a direct comparison between the EVRPTW augmented graph and multigraph formulations across all objective functions and problem instances in Figure 4.8. The figure on the left summarizes the MILP comparisons, middle the alternative resource CP comparisons, and the right the single resource CP comparisons. Each data point represents a comparison of MRE values for a single instance, where the bound used for the calculation is the bound retrieved by the multigraph MILP model in all cases.

As indicated by the figure, the multigraph formulations for MILP and the alternative resource CP model offer clear advantages in solution quality over their augmented graph counterparts (represented by the majority of data points below the diagonal). The single resource CP model, on the other hand, appears to favour the augmented graph formulation. This would indicate room for improvement in the multigraph single resource CP modeling approach: the use of integer energy tracking variables, as opposed to cumulative function expressions defined directly over the vehicle visit interval variables, likely result in less effective inference. We leave an investigation into a more effective single resource CP model based on recharging path multigraphs to future work.

We also provide a comparison of the best MILP approach for EVRPTW (the multigraph model) versus the best CP approach (the single resource augmented graph model) in Figure 4.9. We produce a separate plot for each of the objective functions. From the plots it is clear that the multigraph MILP formulation is superior for the fleet distance minimization objective function, finding and proving optimality for many instances (as indicated by the cluster of data points on the x-axis), and only being outperformed by CP for a few instances (data points above the diagonal line). However, for fleet size minimization and the combined objective, the CP approach is more competitive, outperforming MILP on a number of instances and offering comparable performance for many more. These results align with our observations on the relative performance of these techniques. We discuss potential reasons for this in Section 4.6.4.

4.6.3 PDPTW-EV

In this section we conduct an empirical assessment of our PDPTW-EV models. We summarize the instances used and present results for the various formulations. Due to the more complex nature of the

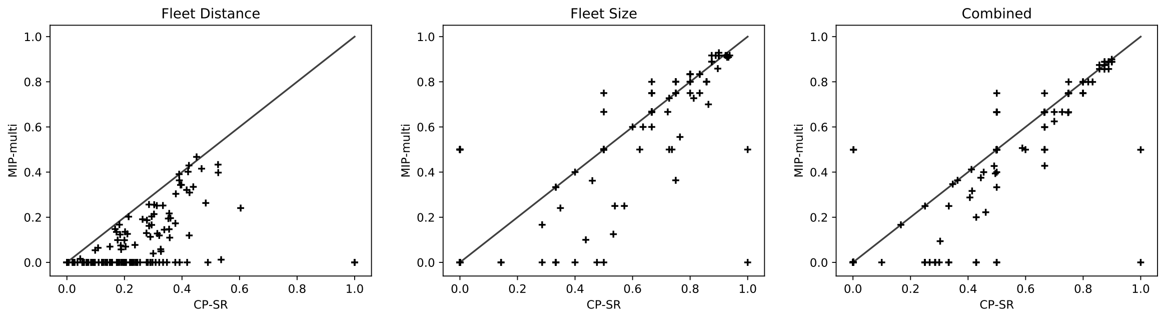


Figure 4.9: EVRPTW formulation comparison (best of MILP versus best of CP). MRE values for each instance plotted. Values on diagonal indicate equal performance for techniques in plot.

PDPTW-EV, we do not use our simple heuristic to warm-start the CP approaches; allocating both a pickup and a delivery to a vehicle often results in the need for more recharge visits than naive solutions for EVRPTW, rendering the naive warm-start less useful.

4.6.3.1 Instances

We conduct our PDPTW-EV evaluation on a set of problem instances taken from the literature [82]. These instances vary with respect to the number of pickups/deliveries (i.e., $|P \cup D| \in \{6, 10, 12, 16\}$) and the number of recharge stations. This benchmark, which we call the 'small' benchmark, contains a total of 36 instances. We also investigated the large benchmark set from the literature [82] (with $|P \cup D| \geq 100$ and $|F| = 21$), which was designed to test metaheuristic approaches, but found that it was too large for our models to even produce feasible solutions in the vast majority of cases. Instead, we take this benchmark of larger instances and remove pickup/delivery requests such that $|P \cup D| = 50$ (leaving recharging stations untouched to preserve feasibility) for a modified version of the benchmark set; we call this test set the 'large' benchmark. The large benchmark has 56 instances for a total of 92 instances.

4.6.3.2 Experiment: Augmented Graph Formulations

The results for the augmented graph PDPTW-EV formulations are presented in Table 4.5.

Full Runtime Analysis. The augmented graph MILP approach ($\mathbb{MILP}^{G'}_{PDPTW\text{-}EV}$) is notably strong on distance minimization and the combined objective function for the small problems. Overall, it performs the best (in terms of solution quality) for 4/12 (33.3%) of the experiment classes after the full one hour runtime. The approach is notably weak for large problems, where it fails to find even a feasible solution for any of the problem instances.

The augmented graph alternative resource CP model ($\mathbb{CP}^{G'\text{-}AR}_{PDPTW\text{-}EV}$) is outperformed by its single resource CP counterpart for all experiment classes except for long horizon fleet minimization and the combined objective function. For these classes, it is able to find the most feasible solutions for large problems, although it is evident that these solutions are not necessarily of high quality as indicated by the high MRE values. The alternative resource approach outperforms MILP for large problems overall, where it is able to find feasible solutions to some problems.

The single resource CP model ($\mathbb{CP}^{G'\text{-}SR}_{PDPTW\text{-}EV}$) has the overall strongest performance out of the augmented graph models, finding the best solutions for most of the instances in 8/12 (66.7%) of the

Table 4.5: Experimental results, PDPTW-EV, augmented graph formulations. The best result of each column for each objective function in bold. One hour runtime limit.

	Short Horizon						**Long Horizon**					
	# Feasible		# Best		MRE (%)		# Feasible		# Best		MRE (%)	
Method	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.
Fleet Distance												
$\mathbb{MILP}^{G'}_{PDPTW\text{-}EV}$	12	0	**12**	0	**8.7**	N/A	**17**	0	**17**	0	**11.5**	N/A
$\mathbb{CP}^{G'\text{-}AR}_{PDPTW\text{-}EV}$	7	5	4	3	9.9	95.2	12	21	2	3	17.8	98.8
$\mathbb{CP}^{G'\text{-}SR}_{PDPTW\text{-}EV}$	**15**	**9**	10	**9**	25.6	**93.8**	16	**23**	6	**23**	19.9	**94.7**
Fleet Size												
$\mathbb{MILP}^{G'}_{PDPTW\text{-}EV}$	12	0	10	0	26.7	N/A	16	0	15	0	18.3	N/A
$\mathbb{CP}^{G'\text{-}AR}_{PDPTW\text{-}EV}$	8	4	8	3	**10.4**	**12.5**	13	**23**	12	7	**6.4**	69.6
$\mathbb{CP}^{G'\text{-}SR}_{PDPTW\text{-}EV}$	**16**	**9**	**16**	**9**	26.7	13.3	**17**	21	**17**	**20**	11.8	**47.3**
Combined												
$\mathbb{MILP}^{G'}_{PDPTW\text{-}EV}$	13	0	**12**	0	25.2	N/A	16	0	**13**	0	15.7	N/A
$\mathbb{CP}^{G'\text{-}AR}_{PDPTW\text{-}EV}$	8	5	4	3	**14.6**	27.2	14	**23**	3	7	12.0	70.3
$\mathbb{CP}^{G'\text{-}SR}_{PDPTW\text{-}EV}$	**16**	**9**	10	**9**	29.4	**13.4**	**17**	21	8	**20**	**11.9**	**43.6**

experiment classes. The approach finds the most feasible solutions for all experiment classes except large, long horizon problems for fleet minimization and the combined objective. Furthermore, it exhibits the strongest performance on all large problems, in terms of solution quality, buy a substantial margin.

The results suggest that while the augmented graph MILP model is an attractive option for small problems, overall, it is not effective for large problems. The increased routing network size, in addition to the auxiliary variables that ensure pickups and deliveries occur on the same route (not present in EVRPTW models) result in the inability to even find feasible solutions, where feasibility was a non-issue for EVRPTW problems of the same size.

4.6.3.3 Experiment: Recharging Path Multigraph Reformulations

The results for the recharging path multigraph PDPTW-EV formulations are presented in Table 4.6. Experiment runtime for the multigraph models includes the time required to pre-process the graph. We also include results for the single resource augmented graph CP model for comparison purposes.

Full Runtime Analysis. From the table, it is clear that the multigraph-based MILP model is the strongest performer out of all the methods for small problems across all experiment classes. For these problems, it is often able to solve problems to proven optimality as indicated by the 0.0% MRE values. While it maintains reasonable performance for large fleet distance minimization problems, it struggles to find feasible solutions for large problems across the other objective functions. These results follow the trend observed for the augmented graph MILP model where performance drops off considerably as instances get larger. Overall, it provides the best results, in terms of solution quality, for 7/12 (58.3%) of the experiment classes.

The multigraph-based alternative resource CP model demonstrates strong performance, in terms of finding feasible solutions, for long horizon problems, with markedly weaker performance for short horizon

Table 4.6: Experimental results, PDPTW-EV, recharging path multigraph formulations (single resource augmented CP formulation for comparison). The best result of each column for each objective function in bold. One hour runtime limit.

	Short Horizon						**Long Horizon**					
	# Feasible		# Best		MRE (%)		# Feasible		# Best		MRE (%)	
Method	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.	Sm.	Lg.
Fleet Distance												
$\mathbb{MILP}^{\mathcal{G}}_{PDPTW\text{-}EV}$	**16**	**9**	**16**	**9**	**0.0**	**1.5**	**18**	6	**18**	6	**1.2**	**4.1**
$\mathbb{CP}^{\mathcal{G}\text{-}AR}_{PDPTW\text{-}EV}$	10	5	1	0	4.0	70.0	16	**24**	8	3	8.4	86.0
$\mathbb{CP}^{\mathcal{G}\text{-}SR}_{PDPTW\text{-}EV}$	10	**9**	9	7	0.5	31.8	13	14	5	6	4.3	85.1
$\mathbb{CP}^{G'\text{-}SR}_{PDPTW\text{-}EV}$	15	**9**	8	6	0.7	32.9	16	23	6	**12**	7.9	75.9
Fleet Size												
$\mathbb{MILP}^{\mathcal{G}}_{PDPTW\text{-}EV}$	**16**	4	**16**	4	3.1	**6.8**	**18**	0	**17**	0	5.6	N/A
$\mathbb{CP}^{\mathcal{G}\text{-}AR}_{PDPTW\text{-}EV}$	10	5	5	0	10.8	63.1	17	**24**	12	12	13.7	58.2
$\mathbb{CP}^{\mathcal{G}\text{-}SR}_{PDPTW\text{-}EV}$	11	**9**	10	**9**	**1.8**	20.7	13	16	13	13	**0.0**	**43.6**
$\mathbb{CP}^{G'\text{-}SR}_{PDPTW\text{-}EV}$	**16**	**9**	**16**	**9**	3.1	11.0	17	21	**17**	**14**	2.9	44.2
Combined												
$\mathbb{MILP}^{\mathcal{G}}_{PDPTW\text{-}EV}$	**16**	5	**13**	4	5.0	**0.0**	**18**	1	**13**	0	**2.8**	50.1
$\mathbb{CP}^{\mathcal{G}\text{-}AR}_{PDPTW\text{-}EV}$	10	4	2	0	15.7	74.1	17	**24**	7	4	20.9	63.5
$\mathbb{CP}^{\mathcal{G}\text{-}SR}_{PDPTW\text{-}EV}$	11	**9**	9	**8**	**1.8**	20.8	13	15	8	11	3.8	**39.6**
$\mathbb{CP}^{G'\text{-}SR}_{PDPTW\text{-}EV}$	**16**	**9**	7	7	8.8	11.1	17	21	6	**12**	3.0	**39.6**

problems. Overall, it does not provide the best performance, in terms of solution quality, for any of the instance classes. This indicates that although it is able to find feasible solutions, the approach does not effectively improve the objective value through the search.

The multigraph-based single resource CP model demonstrates the strongest performance, in terms of solution quality, for 2/12 (16.7%) of the experiment classes. Interestingly, for the instances that this approach finds a feasible solution, it also appears to find solutions with the best objective value. However, it is outperformed by the other CP approaches, overall, in terms of feasibility.

Finally, the augmented graph single resource CP model, included for comparison purposes, exhibits the strongest performance for 6/12 (50%) of the experiment classes.

While the single resource transformation yields dramatic benefits for EVRPTW problems, its performance advantage when applied to the PDPTW-EV is more muted. Due to the explicit vehicle tracking of the alternative resource model, when a pickup is assigned to a vehicle, so is its delivery due to the interval variable presence constraints. The single resource models, on the other hand, use the floor operator to ensure the pickup and delivery are allocated to the same vehicle. Additionally, the number of vehicles allocated to the models for the EVRPTW is equal to the number of customers (in the case where each customer is serviced by a separate vehicle), whereas for the PDPTW-EV it is equal to the number of pickups (since each pickup/delivery pair is serviced by the same vehicle). Since, for our large PDPTW-EV problems, $|P| = 25$ such that $|P \cup D| = 50$, the resulting duplication of optional variables for the alternative resource models is significantly less, as there are only half of the vehicles as in the baseline electric routing problem.

4.6.4 General Discussion

Through empirical testing, the introduction of the single resource transformation was shown to be beneficial in CP modeling for larger problem instances, particularly for the EVRPTW. The advantages in model size (i.e., a large reduction in required memory, number of variables, number of constraints) as well as model performance are apparent, particularly for the EVRPTW experiments, where the single resource modeling approach outperforms the alternative modeling strategy for every experiment class tested, often significantly so. Overall, the CP approaches exhibit stronger performance for fleet minimization and the combined objective functions. Conversely, the MILP-based approaches seemed to favor, often dramatically so, the fleet distance minimization objective (and performed comparatively better on the combined objective that incorporates some weighted distance element versus the fleet minimization objective which does not). One possible reason for this is that, in the fleet distance minimization case, all of the binary x_{ij} (and x^h_{ij}) have a direct contribution of cost, d_{ij} (and d^h_{ij}) to the objective function. Due to the variety of branching heuristics employed by MILP solvers that incorporate a variable's relative impact on objective function value [2], such objective functions are likely accompanied with more impactful branching decisions. The fleet minimization objective, on the other hand, is only directly impacted by depot outflow variables, x_{0j} (and x^h_{0j}), while the remainder of the arc-routing variables contribute no cost (or very little in the case of the combined objective) to the objective function, thus potentially resulting in less impactful branching heuristics.

In terms of the EVRPTW, while the single resource CP approach is dominant for the augmented graph formulation experiments, MILP looks to be the superior method when using recharging path multigraphs. The use of graph pre-processing to embed the recharging aspect of the problem into multigraph arcs significantly reduces the size of the network, particularly when all of the dominated edges are removed from the multigraph. Given that the MILP encodes each arc in the multigraph as a binary variable x^h_{ij}, reducing the size of the multigraph has a direct impact on the number of variables in the formulation. The multigraph CP formulations, on the other hand, represent the visits themselves as the primary decision variables (using interval variables); regardless of the size of the multigraph, the number of visit variables remains the same. As aforementioned, the dominance of the multigraph MILP models is significantly muted (particularly for non-distance objective functions, see above) for large PDPTW-EV problems where the integer route-tracking variables are added, an element not needed in the EVRPTW formulations.

Direct comparisons between the modeling strategies seem to indicate the multigraph approaches are superior to their augmented graph counterparts. However, there are distinct drawbacks: the multigraph models cannot be readily applied to problems out-of-the-box, they require non-negligible instance pre-processing precluding their application to new instances in 'real-time', and they are less flexible to changing problem definitions such as capacitated recharging stations. In this case, the generation of the recharging path multigraph is non-trivial as paths must be carefully constructed while reasoning over other paths that use the same set of recharging stations. The augmented graph formulations, on the other hand, can address these characteristics through the inclusion of side constraints.

For example, to model unary capacity recharge stations, the following constraints are added to the the augmented MILP model:

$$\tau_i + g(Q - y_i) \leq \tau_j, \forall i < j \in F'(f), f \in F \tag{4.123}$$

where $F'(f)$ returns the vertices associated with recharge station $f \in F$. Whereas the temporal constraints in $\mathbb{MILP}^{G'}_{EVRPTW}$ ensure that a given vehicle's tasks are sequenced, these constraints ensure that visits to a given recharge station are sequenced.

For the alternative resource CP model, $\mathbb{CP}^{G'\text{-}AR}_{EVRPTW}$, the following constraints are added to the formulation to enforce the same characteristic:

$$\textsc{NoOverlap}(\{x_i^k : k \in K, i \in F'(f)\}), \forall f \in F \tag{4.124}$$

where, with this constraint, all of the interval variables that represent a visit to a given recharge station are made to not overlap.

Finally, in the single resource model, $\mathbb{CP}^{G'\text{-}SR}_{EVRPTW}$, the augmented horizon characteristic requires a different set of constraints, given as follows:

$$\textsc{Pres}(x_i) = \textsc{Pres}(x_j) \rightarrow \textsc{Mod}(\textsc{End}(x_i), H) \leq \textsc{Mod}(\textsc{Start}(x_i), H), \forall i < j \in F'(f), f \in F \tag{4.125}$$

where the Mod operator retrieves the non-augmented start time of the task, and ensures the tasks are sequenced for a given recharge station.

Furthermore, modeling non-unary capacitated visits (i.e., at most κ visits to a recharge station at once) in the alternative resource CP models is easily achieved with the use of the Cumulative constraint. Modeling this characteristic in the single resource model would be more complicated, and likely involve the introduction of additional interval variables, while achieving this in MILP may even require a re-framing of the model to use a time-indexing technique.

4.7 Conclusions

In this chapter, we extended the set of available techniques for solving electric routing problems by presenting the first approaches for solving the electric vehicle routing problem with time windows (EVRPTW) and the pickup and delivery problem with time windows and electric vehicles (PDPTW-EV) using constraint programming (CP). We present scheduling-based CP formulations for each problem based on augmented graph and recharging path multigraph problem representations. To our knowledge, this is the first time that these representations have been used to model the problem in CP.

The initial augmented graph model uses an alternative resource technique previously applied to other routing problems, while the second uses a single resource transformation for CP models using optional activities, sequence variables, and cumulative function expressions. Our novel recharging path multigraph CP model uses the single horizon transformation in addition to graph pre-processing techniques. The single resource transformation significantly extends the size of EVRPTW problems that can be solved with the technology, and can be applied to other homogeneous routing and scheduling problems.

For the augmented graph formulations of the EVRPTW we demonstrate, through empirical evaluation, that our single resource CP approach significantly outperforms the alternative resource CP model, and outperforms the augmented graph MILP model for nearly all medium-to-large problem classes. Furthermore, we demonstrate that the single resource CP approach based on multigraph recharging paths is competitive with its corresponding MILP model for fleet size minimization problems. For the PDPTW-EV, we demonstrate that the proposed CP approaches, while largely outperformed on small problems, can offer superior performance for larger problem sizes.

Chapter 5

Target Search on Road Networks with Range-Constrained UAVs and Ground-based Mobile Recharging Vehicles

In the previous chapter of this dissertation we introduced constraint programming (CP) modeling strategies for vehicle routing problems involving electric vehicles that consume and replenish energy, providing an alternative to existing mixed-integer linear programming (MILP) approaches developed in the vehicle routing community. We applied these strategies to solve two important problems from the vehicle routing literature, namely the electric vehicle routing problem with time windows (EVRPTW) and the pickup and delivery problem with time windows and electric vehicles (PDPTW-EV), with promising results. The remaining chapters of this dissertation investigate the application of our novel CP methods, as well as the adaptation of existing MILP approaches, to real-world problems involving the routing of battery-powered vehicles in the presence of increasingly complex side constraints.

In this chapter, specifically, we investigate the application of our techniques in another setting, namely the routing and coordination of joint *unmanned aerial vehicle* (UAV) and ground vehicle fleets in the context of large-scale search and track missions on road networks. We leverage the modeling strategies outlined in the previous chapter to develop both MILP and CP approaches to the problem, tackling application-specific modeling challenges, such as vehicle synchronization, in both paradigms. The work in this chapter also proposes a technique for the construction of strategically discretized routing graphs, starting from high-resolution road network data, over which our optimization-based approaches are defined. Together, the routing graph construction pipeline and optimization-based approaches for coordinating vehicle fleets form a promising approach for use in real-time target search situations.

Contributions to Co-Authored Work. The work in this chapter is based on and extends our research published in *IEEE Robotics & Automation Letters [26].* Co-author Chiara Piacentini assisted with target simulation and problem instance generation and supported target search optimization model development. Co-authors Sara Bernardini and J. Christopher Beck supervised this work.

5.1 Introduction

Unmanned aerial vehicles (UAVs) have had impact across a wide variety of industries, including logistics [161], agriculture [87], and surveillance [20]. In the context of the latter, the problem of searching for lost targets has a long history, with theoretical studies dating back to the 1940s [123]. The search problem involves routing a fleet of surveying units to try and find a moving target. Once the target has been found, it can then be tracked; the work in this chapter focuses solely on the search phase of these operations.

While the existing literature surrounding UAV search and track problems is extensive [20, 96, 171, 108], there is little work that looks at the viability of real-world, large-scale target search capabilities using range-constrained, commercially available UAVs. Commercial UAVs have a considerably shorter flight time than fixed-wing military-grade designs [81]. For example, the DJI Matrice 200 has a maximum unloaded flight time of roughly 38 minutes [193], whereas it is not uncommon for military-grade designs to have flight times far exceeding 10 hours [81]. As such, we study the range-constrained multi-UAV target search problem based on commercially available UAV specifications and the use of mobile recharging vehicles (MRVs). The MRVs can travel, via the road network associated with the search area, to meet up with and recharge a UAV. Our investigation looks to identify the viability of commercially available UAVs for search-phase surveillance missions when utilized standalone, as well as when deployed with an accompanying fleet of MRVs.

Contributions of chapter. We investigate the modeling of range-constrained routing for heterogeneous fleets involving both UAVs and MRVs. The contributions of this chapter are as follows:

1. We propose a pipeline for representing the range-constrained target search problem over real-world road networks, starting with a map of the road network and yielding a final routing graph that permits UAVs to recharge via rendezvous with MRVs.

2. We model and solve the resulting graph representation of the problem with *mixed-integer linear programming* (MILP) and *constraint programming* (CP), adopting modeling techniques developed in the previous chapters of this dissertation. The MILP is constructed with a compact formulation that leverages the homogeneity of UAV and MRV fleets while the CP approaches follow the alternative resource and single resource models.

3. We conduct a simulation-based assessment of our methods using real-world road network data from Scotland. We investigate the impact of different durations of UAV recharging operations, as well as the impact on accumulated fleet reward using both nominal and increased UAV flight speeds. Furthermore, we show that while the alternative resource CP model is superior for the real-world instances investigated, the single resource CP model offers attractive performance as problem size scales.

Outline of chapter. The outline of this chapter is as follows. Section 5.2 defines the problem and Section 5.3 summarizes related work. Section 5.4 presents the pipeline for representing the problem over real-world road networks, while Section 5.5 details the MILP and CP target search optimization models. Section 5.6 details experimental setup, results, and analysis, and Section 5.7 provides concluding remarks.

5.2 Problem Definition

The problem studied involves cooperatively routing a fleet of homogeneous UAVs in pursuit of a mobile ground-based agent (the "target"). In this problem variant, in contrast to those presented before [20, 19, 169], each UAV is range-constrained due to limited battery capacity. Ground-based *mobile recharge vehicles* (MRVs) are available to provide the UAVs the opportunity to replenish their battery.

At a high-level, the problem can be posed as follows: given a mixed-fleet of UAVs and MRVs, a map of the road network, and a set of locations likely to contain the target at certain time intervals, determine a temporally feasible and range-compliant search plan for the fleet that maximizes the accumulated expectation of discovering the target. The plan consists of a sequence of *search patterns*, i.e., target search maneuvers performed by the UAVs at specific locations and times, as well as recharge actions involving both UAVs and MRVs. For a UAV to initiate a recharge, it must be in the same location as the MRV at the same time. In this work, we assume that UAVs do not suffer from communication-related range constraints, and that UAVs consume energy while traveling and while conducting a search pattern. The remainder of this section describes the components of the problem in detail. We follow existing work for the majority of the problem definition [20, 169] with minor changes to notation.

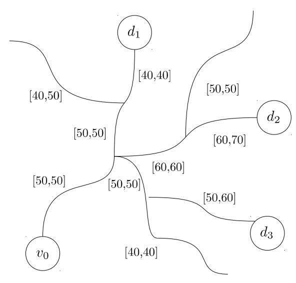
5.2.1 Environment and Target

Our problem considers an environment in two-dimensional Euclidean space characterized by a *road network* attained from the underlying map of the search area and defined by a series of roads, each of which is a sequence of road segments with varying speed limits (see Figure 5.1a). The road network is then discretized into cells with side length δ to produce a *problem graph*, $G = (V, E)$ (Figure 5.1b-5.1c). Each vertex, $v \in V$, in the problem graph represents a cell containing at least one road segment, and each undirected edge, $e \in E$, represents an adjacent pair of vertices connected by a road segment in the underlying road network. Each edge, $e \in E$, is labeled with its minimum and maximum travel speeds, s_e^{min} and s_e^{max}, respectively. The target is characterized by a *last known position* (LKP), v_0, and a set of potential destinations, $d \in D \subset V$. The process of going from the real-world road network to the problem graph, G, with search patterns is illustrated by Figures 5.1a to 5.1c.

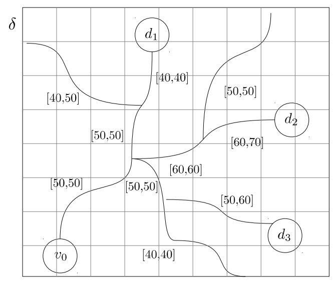
5.2.2 Fleets

The fleet of homogeneous UAVs is denoted $o \in O$. UAVs move between locations of interest at a constant speed of s_O in metres per second. Q denotes the total battery capacity of a UAV (with 0 being the minimum capacity) and the per-unit-distance energy consumption of each UAV is given by g. The total energy consumption for a search pattern is given by h.

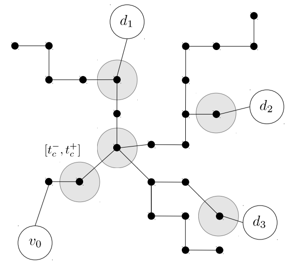
The fleet of homogeneous *mobile recharging vehicles* (MRVs) is denoted $k \in K$. MRVs move along the road network subject to road segment speed limits. To recharge, the UAV and an MRV must meet in the same location at the same time, and this location must contain a road segment. In this work, the recharging of a UAV is executed via a constant time battery swap operation [154], where a UAV meets an MRV and has its battery replaced resulting in a full charge. This operation is assumed to take ξ seconds. MRVs are not range constrained and can recharge a single UAV at a time.



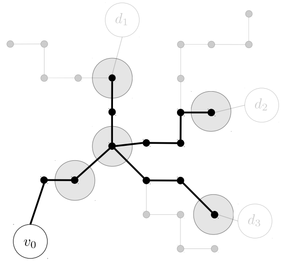
(a) Road network, last known position, v_0, and destinations, $d \in D$.



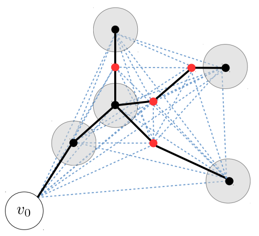
(b) Discretization cells with side length δ.



(c) Problem graph, G, and search pattern locations.



(d) Shortest paths between search patterns.



(e) UAV and MRV routing graph, $\mathcal{G}$.

Figure 5.1: UAV and MRV routing graph construction pipeline.

5.2.3 Target Simulation and Search Patterns

Following previous work [19], the target's motion through the graph, G, is simulated with a standard Monte Carlo simulation (MCS). To identify vertices in the graph that have the highest probability of containing the target at various times, the MCS uses a probability distribution defined over the set of possible target destinations, D, shortest paths to each of those destinations from the LKP, and estimated target travel speeds. For each simulated time step, the MCS selects the vertices with the highest probability of containing the target, and creates a search pattern centered on them, with time windows assigned to reflect when the target can plausibly be in these areas. Additional detail regarding the MCS procedure can be found in existing work [20, 19].

The search pattern itself is a pre-planned search maneuver performed by a UAV at a specific location. Previous work has identified a series of standard manoeuvres that the UAVs can perform, such as spirals and lawnmowers, depending on the topology of the search pattern location, as visualized in Figure 5.2. Spiral manoeuvres are useful for covering more dense, urban areas, while the lawnmower is more effective at searching over elongated sections [169].

Each search pattern, $c \in C$, is characterized by a location, processing time, time window, and reward. We assume, for the purposes of this work, that the manoeuvre is embedded in the processing time of the search pattern. In Figure 5.1c, the locations of possible search patterns are represented by grey circles. The reward for a UAV performing a search pattern is calculated following previous work [20] and represents a measure that the target will be in the area of the search pattern during its time window; a higher reward is better. The time at which a UAV starts a search pattern must be within its time window. The planning horizon, H, is the end of the latest time window, such that no search pattern can be finished after this time. A UAV must complete the full duration of an assigned search pattern before starting another.

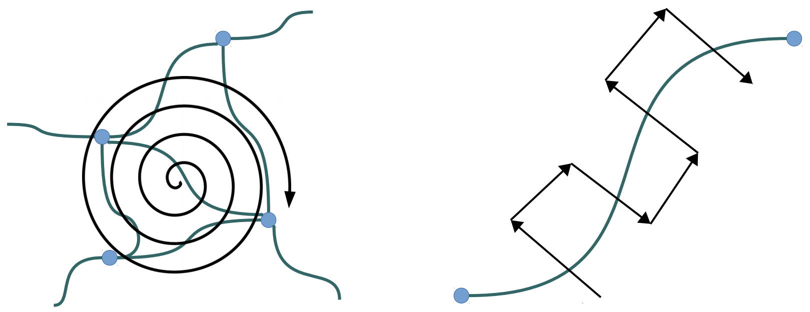


Figure 5.2: Standard UAV search manoeuvres. Spiral (left) manoeuvre for denser, urban areas and lawnmower (right) manoeuvre for less dense, longer, straighter sections. Underlying topology in green, UAV search manoeuvre in black.

5.2.4 Problem Input and Solution

To summarize, as input a problem instance specifies the original road network (problem graph), $G = (V, E)$, last known position of the target, v_0, set of search patterns C, UAV fleet characterized by $\langle O, s_O, Q, g, h\rangle$, and MRV fleet characterized by $\langle K, s_K, \xi\rangle$. The objective is to create a joint UAV/MRV search plan that maximizes accumulated reward. More specifically:

Definition 5.2.1. *A solution to the problem is a search plan consisting of a schedule of search patterns and recharging events assigned to each UAV and a schedule of recharging events assigned to each MRV such that:*

- *UAV schedules are temporally feasible, where these vehicles travel at speed s_O and move between locations of interest in straight lines (i.e., Euclidean distances).*
- *UAV schedules are energy feasible such that energy level of these vehicles does not deplete below 0 and does not exceed their energy capacity Q at any point in the schedule.*
- *MRV schedules are temporally feasible (we do not consider the need for the ground vehicle to refuel), where these vehicles travel at speed s_K along the road network and adhere to the upper speed limits on the involved road segments.*
- *Recharging events involve one UAV and one MRV and both vehicles are synchronized in time and space during the event duration, ξ.*
- *The accumulated reward, the summation of individual rewards of search patterns conducted, of the search plan is maximized.*

5.3 Related Work

Research on UAV deployment has received considerable interest, with early work introducing task assignment methods [21] and the development of capabilities such as automated battery swap-out systems for UAV recharging [154]. The impact of commercial UAVs on logistics resulted in rapid algorithmic development in the operations research and robotics literature [161, 195, 91, 149]. A review of optimization approaches to UAV routing is found in [163].

The routing of range-constrained UAVs while considering recharging has also seen considerable interest. A number of these works propose MILP models for the routing of UAV fleets with fixed location recharge stations [190, 118], while others propose heuristics and approximation algorithms [94]. Recent work incorporates recharging station placement into MILP-based UAV routing models in the context of logistics applications [100, 36]. In these works, the existence of pre-defined potential facility locations results in smaller networks that can be modeled using single-stage optimization. Conversely, our work, which looks to identify valuable locations for UAV/MRV synchronization over large real-world road networks requires significant discretization and filtering prior to optimization, as proposed by our pipeline.

The most relevant recent work considers mobile recharging stations similar to our MRVs [148, 211]. The work of Maini and Sujit plans paths for a single UAV and single MRV, traveling via a road network, for surveillance operations [148]. The authors propose a greedy solution, in contrast to our joint multiple vehicle UAV/MRV routing models, that first finds a path for the MRV and then generates a path for the UAV. The recent work of Yu et al. explores joint routing of a range-constrained UAV and MRVs for site visits [211]. They investigate three problem variants: i) single UAV routing with multiple stationary recharging locations, ii) single UAV/single MRV joint routing, and iii) a single UAV/multiple MRV problem that, for a given UAV path, looks to minimize the number of MRVs used. Each of these problems is similar to ours, though we note that the work of Yu et al. permits the UAV to be transported by the ground vehicle, whereas we do not. Further differences between this work and our own: it does not start from a road network as input, it considers a single UAV, it does not consider time windows for site visits, and the underlying structure is a traveling salesman problem as opposed to the orienteering problem [109] structure in our work.

5.4 Routing Graph Construction

The problem graph, G, which is used to generate search pattern candidates, is too large to use directly in a monolithic optimization model. As such, following existing work in electric vehicle routing [23, 52], we define a *routing graph* to facilitate the development of mathematical models for the problem. Vertices in the routing graph represent locations of interest and arcs represent travel segments between them.

From the discretized problem graph, G (Figure 5.1c), we generate a routing graph, $\mathcal{G} = (\mathcal{V}_{0,N+1}, \mathcal{E})$, where $\mathcal{V}_{0,N+1}$ is the set of $N+2$ vertices that UAVs/MRVs can possibly visit (N is the number of non-depot vertices), and $\mathcal{E}$ the set of edges connecting them. We let v_0, the vertex representing target LKP, represent the start depot for all vehicles. We let v_{N+1} represent the end depot for all vehicles with zero travel distance to all other vertices in $\mathcal{G}$; this vertex is simply introduced for modeling purposes.[1] The set of vertices representing possible instances of search patterns is denoted by $\mathcal{V}^C$. The location, processing time, time window, and reward associated with a particular search pattern instance, $v_i \in \mathcal{V}^C$, are given by ℓ_i, p_i, $[t_i^-, t_i^+]$, and r_i, respectively. We let Δ_{ij}^{Euc} represent the Euclidean distance between vertices v_i and v_j, and Δ_{ij}^{SP} represent the shortest path MRV travel time between v_i and v_j, following the road network and conforming to speed limits.

In the non-range-constrained variant of the problem (i.e., without MRVs), the final routing graph would simply be defined over the depot nodes, $\{v_0, v_{N+1}\}$, and $\mathcal{V}^C$. However, in the range-constrained

[1] UAVs do not return to the start depot. Instead, it is assumed that each UAV can be collected from the location of its last search pattern after the search is concluded.

Algorithm 2: Construct recharge vertex set, $\mathcal{V}^F$

```
Data: Problem graph, $G = (V, E)$, filtering parameter, $\phi$
Result: $\mathcal{V}^F$
$\mathcal{V}^+ \leftarrow \emptyset$
for $(v_i, v_j) \in \mathcal{V}^C \times \mathcal{V}^C : (t_i^- + p_i + \frac{\Delta_{ij}^{Euc}}{s_O}) \leq t_j^+$ do
    $\mathcal{V}^+ \cup$ ShortestPathNodesFiltered$(v_i, v_j, G, \phi)$
$\mathcal{V}^F \leftarrow$ ModifiedHittingSet$(\mathcal{V}^+)$
return $\mathcal{V}^F$
```

variant, the graph must be augmented with vertices allowing opportunities for UAVs to meet MRVs to recharge. We let the set of these recharge vertices be represented by $\mathcal{V}^F$. Constructing the set of recharge vertices, $\mathcal{V}^F$, must be done carefully to avoid dramatically increasing the size of the graph. The procedure we use for accomplishing this is detailed in Algorithm 2.

For each pair of search pattern vertices that can feasibly be executed by the same UAV, the function ShortestPathNodesFiltered computes the shortest path through the problem graph, G, using Dijkstra's algorithm; the edge weights in this case represent the minimum travel time (dictated by road segment speed limits) between the two search pattern locations. The algorithm then returns the nodes involved in the path. To reduce the number of nodes considered, the final step of this function is to filter the set of returned nodes according to filtering parameter ϕ, which ensures nodes from a given path added to $\mathcal{V}^+$ are separated by a distance of at least ϕ. The path is traversed, and a node from the unfiltered set is selected every ϕ metres. For each pair of search patterns, only vertices that have not been seen before are added to $\mathcal{V}^+$.

As a final vertex selection step, ModifiedHittingSet takes as input the set of filtered shortest path vertices, $\mathcal{V}^+$, and solves the integer program (IP) detailed by Eqns. (5.1) through (5.6), similar to a previous approach based on geometric hitting sets [148]. The solution to the IP selects the set $\mathcal{V}^F \subseteq \mathcal{V}^+$ of recharge vertices. Variable α_i is 1 if vertex v_i is selected, while β_{ij} is 1 if vertex v_i is assigned to selected vertex v_j. Δ_{ij}^{Euc} is the euclidean distance between vertices v_i and v_j, and R is the maximum permissible distance between a selected vertex and a vertex assigned to it. We use the shorthand $i \in \mathcal{V}$ to indicate vertex $v_i \in \mathcal{V}$ in the remainder of the chapter.

$$\min \quad \sum_{i \in \mathcal{V}^+} \alpha_i \tag{5.1}$$

$$\text{subject to:} \quad \sum_{j \in \mathcal{V}^+} \beta_{ij} = 1 \qquad \forall i \in \mathcal{V}^+ \tag{5.2}$$

$$\Delta_{ij}^{Euc} \beta_{ij} \leq R\alpha_j \qquad \forall i, j \in \mathcal{V}^+ \tag{5.3}$$

$$\alpha_i = 1 \qquad \forall i \in \mathcal{V}^+ \mid i \in \mathcal{V}^C \tag{5.4}$$

$$\alpha_i \in \{0, 1\} \qquad \forall i \in \mathcal{V}^+ \tag{5.5}$$

$$\beta_{ij} \in \{0, 1\} \qquad \forall i, j \in \mathcal{V}^+ \tag{5.6}$$

The IP model finds the minimum number of selected vertices subject to constraints. Constraint (5.2) ensures each vertex is assigned to a selected vertex. Constraint (5.3) dictates that a vertex can only be assigned to a selected vertex if it is less than R from the selected vertex. Constraint (5.4) ensures that vertices coinciding with search pattern locations are always selected, while the remaining constraints dictate variable domains. Selected vertices v_i with $\alpha_i = 1$ in the optimal solution to the IP are added to $\mathcal{V}^F$.

The final routing graph, $\mathcal{G} = (\mathcal{V}_{0,N+1}, \mathcal{E})$, consists of vertices $\mathcal{V}_{0,N+1} = \{v_0, v_{N+1}\} \cup \mathcal{V}^C \cup \mathcal{V}^F$. The set of non-depot vertices is defined as $\mathcal{V} = \mathcal{V}^C \cup \mathcal{V}^F$. Indices are used to specify sets that contain instances of the depot, e.g., $\mathcal{V}_0 = \{v_0\} \cup \mathcal{V}$, $\mathcal{V}_{N+1} = \mathcal{V} \cup \{v_{N+1}\}$, and $\mathcal{V}_0^F = \{v_0\} \cup \mathcal{V}^F$. The edge set is defined as $\mathcal{E} = \{(i,j) : i \in \mathcal{V}_0, j \in \mathcal{V}_{N+1}, i \neq j\}$.

5.5 Target Search Optimization Models

In this section, we present both MILP and CP models for solving the UAV and MRV routing problem over the generated routing graph, $\mathcal{G}$. Recall that $\mathcal{V}^C$ are the vertices pertaining to search pattern executions and $\mathcal{V}^F$ those that pertain to possible recharge opportunities.

5.5.1 Mixed-Integer Linear Programming

Our MILP model is defined by Eqns. (5.7) through (5.26). Binary decision variable x_{ij} is 1 if a UAV travels from v_i to v_j and 0 otherwise. Similarly, binary decision variable y_{ij} is 1 if a MRV travels from v_i to v_j and 0 otherwise. Continuous variable τ_i represents the arrival time of any vehicle at vertex v_i. Continuous variable e_i indicates UAV energy level upon arrival at vertex v_i.

$$\max \quad \sum_{i \in \mathcal{V}^C} \sum_{j \in \mathcal{V}_{N+1}} x_{ij} r_i \tag{5.7}$$

subject to:

$$\sum_{j \in \mathcal{V}} x_{0j} \leq |O| \tag{5.8}$$

$$\sum_{j \in \mathcal{V}^F} y_{0j} \leq |K| \tag{5.9}$$

$$\sum_{j \in \mathcal{V}_{N+1}, i \neq j} x_{ij} \leq 1 \qquad \forall i \in \mathcal{V} \tag{5.10}$$

$$\sum_{j \in \mathcal{V}^F_{N+1}, i \neq j} y_{ij} \leq 1 \qquad \forall i \in \mathcal{V}^F \tag{5.11}$$

$$\sum_{i \in \mathcal{V}_{N+1}, i \neq j} x_{ji} - \sum_{i \in \mathcal{V}_0, i \neq j} x_{ij} = 0 \qquad \forall j \in \mathcal{V} \tag{5.12}$$

$$\sum_{i \in \mathcal{V}^F_{N+1}, i \neq j} y_{ji} - \sum_{i \in \mathcal{V}^F_0, i \neq j} y_{ij} = 0 \qquad \forall j \in \mathcal{V}^F \tag{5.13}$$

$$\sum_{j \in \mathcal{V}_{N+1}} x_{ij} \leq \sum_{j \in \mathcal{V}^F_{N+1}} y_{ij} \qquad \forall i \in \mathcal{V}^F \tag{5.14}$$

$$\tau_0 + \frac{\Delta^{Euc}_{0j}}{s_O} x_{0j} - H(1 - x_{0j}) \leq \tau_j \qquad \forall j \in \mathcal{V}_{N+1} \tag{5.15}$$

$$\tau_i + (\frac{\Delta^{Euc}_{ij}}{s_O} + p_i) x_{ij} - H(1 - x_{ij}) \leq \tau_j \qquad \forall i \in \mathcal{V}^C, j \in \mathcal{V}_{N+1} \tag{5.16}$$

$$\tau_i + (\frac{\Delta^{Euc}_{ij}}{s_O} + \xi) x_{ij} - H(1 - x_{ij}) \leq \tau_j \qquad \forall i \in \mathcal{V}^F, j \in \mathcal{V}_{N+1} \tag{5.17}$$

$$\tau_0 + \Delta^{SP}_{0j} y_{0j} - H(1 - y_{0j}) \leq \tau_j \qquad j \in \mathcal{V}^F_{N+1} \tag{5.18}$$

$$\tau_i + (\Delta_{ij}^{SP} + \xi)y_{ij} - H(1-y_{ij}) \leq \tau_j \qquad \forall i \in \mathcal{V}^F, j \in \mathcal{V}_{N+1}^F \tag{5.19}$$

$$e_i - (g\Delta_{ij}^{Euc} + h)x_{ij} + Q(1-x_{ij}) \geq e_j \qquad \forall i \in \mathcal{V}^C, j \in \mathcal{V}_{N+1} \tag{5.20}$$

$$Q - g\Delta_{ij}^{Euc}x_{ij} + Q(1-x_{ij}) \geq e_j \qquad \forall i \in \mathcal{V}_0^F, j \in \mathcal{V}_{N+1} \tag{5.21}$$

$$t_i^- \leq \tau_i \leq t_i^+ \qquad \forall i \in \mathcal{V}^C \tag{5.22}$$

$$0 \leq e_i \leq Q \qquad \forall i \in \mathcal{V}_{N+1} \tag{5.23}$$

$$\tau_i \geq 0 \qquad \forall i \in \mathcal{V}_{0,N+1} \tag{5.24}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in \mathcal{V}_0, j \in \mathcal{V}_{N+1} \tag{5.25}$$

$$y_{ij} \in \{0,1\} \qquad \forall i \in \mathcal{V}_0^F, j \in \mathcal{V}_{N+1}^F \tag{5.26}$$

Objective (5.7) maximizes the total accumulated reward of the UAV search routes. Constraints (5.8) and (5.9) ensure the number of UAVs and MRVs routed are limited by their respective fleet sizes. Constraints (5.10) through (5.13) enforce the flow through the graph for both UAVs and MRVs. Constraint (5.14) ensures that if a UAV visits a recharge vertex, it must be also visited by an MRV. By the definition of τ_i, all vehicles that visit vertex v_i are synchronized in time. Constraints (5.15) and (5.16) dictate UAV arrival time at a vertex v_j when the preceding vertex v_i is the depot or a search pattern, while Constraint (5.17) does the same when the preceding vertex is a recharge vertex. Similarly, Constraints (5.18) and (5.19) enforce arrival times at vertices for MRVs. Note that a solution to the model may have a UAV or MRV waiting between vertex visits. The remaining energy of a UAV at vertex v_j given the previous vertex v_i was a search pattern is given by Constraint (5.20), while Constraint (5.21) details the remaining energy if the previous vertex v_i was the depot or a recharge visit. Energy is consumed due to travel between locations and search pattern execution (no energy is consumed while waiting). Finally, Constraints (5.22)-(5.26) express the various domains of the decision variables. Given a solution, the routes of the vehicles can be efficiently determined based on the binary decision variable values. The model contains $|\mathcal{V}_{0,N+1}|^2 + |\mathcal{V}_{0,N+1}^F|^2$ binary variables and $2 \cdot |\mathcal{V}_{0,N+1}|$ continuous variables.

5.5.2 Constraint Programming

Our second approach uses constraint programming (CP). CP has been successfully applied to UAV-routing problems in recent work [92, 91, 195, 173], however, these do not consider the routing of a UAV fleet in tandem with a supporting MRV fleet for mobile recharging. As with existing work [23, 127], we make use of *optional interval*, *sequence*, and *cumulative function expression* variables. Following the previous chapter in this dissertation, we investigate both alternative resource and single resource modeling techniques.

5.5.2.1 Alternative Resource Model

First, we detail our alternative resource CP model. This model tracks each vehicle (UAVs and MRVs) explicitly; solutions provided by the solver directly correspond to implementable schedules without any post-processing required. We introduce optional interval variables, x_i^o, to represent a visit from UAV $o \in O$ to vertex v_i. Similarly, we introduce optional interval variables, y_i^k, to represent a visit from MRV $k \in K$ to vertex v_i. We define sequence variable π^o to represent the sequence of UAV $o \in O$ visits, while sequence variable ρ^k represents the sequence of MRV $k \in K$ visits. Finally, we let cumulative function expression variable $\mathcal{Q}^o$ represent the energy level throughout the schedule of UAV $o \in O$. Our alternative resource CP model is defined by Constraints (5.27) through (5.48).

$$\max \quad \sum_{o\in O}\sum_{i\in\mathcal{V}^C} \textsc{Pres}(x_i^o)\cdot r_i \tag{5.27}$$

subject to:

$$\textsc{NoOverlap}(\pi^o, \{\frac{\Delta_{ij}^{Euc}}{s_O} : (i,j)\in\mathcal{E}\}) \qquad \forall o\in O \tag{5.28}$$

$$\textsc{NoOverlap}(\rho^k, \{\Delta_{ij}^{SP} : (i,j)\in\mathcal{E}\}) \qquad \forall k\in K \tag{5.29}$$

$$\begin{aligned}\mathcal{Q}^o &= \textsc{StepAtStart}(x_0^o, Q)\\ &- \sum_{i\in\mathcal{V}^C} \textsc{StepAtStart}(x_i^o, g\Delta^{Euc}_{\textsc{Prev}_{\pi^o(x_i^o),i}} + h)\\ &+ \sum_{i\in\mathcal{V}^F_{N+1}} \textsc{StepAtStart}(x_i^o, [0, Q - g\Delta^{Euc}_{\textsc{Prev}_{\pi^o(x_i^o),i}}]) \qquad \forall o\in O\end{aligned} \tag{5.30}$$

$$\textsc{AlwaysIn}(\mathcal{Q}^o, [0,H], [0,Q]) \qquad \forall o\in O \tag{5.31}$$

$$\textsc{AlwaysIn}(\mathcal{Q}^o, x_i^o, [Q,Q]) \qquad \forall i\in\mathcal{V}^F, o\in O \tag{5.32}$$

$$\textsc{StartAtStart}(x_i^o, y_i^k) \qquad \forall o\in O, k\in K, i\in\mathcal{V}^F \tag{5.33}$$

$$\sum_{o\in O}\textsc{Pres}(x_i^o)\le 1 \qquad \forall i\in\mathcal{V}^C \tag{5.34}$$

$$\sum_{o\in O}\textsc{Pres}(x_i^o)\le 1 \qquad \forall i\in\mathcal{V}^F \tag{5.35}$$

$$\sum_{k\in K}\textsc{Pres}(y_i^k)\le 1 \qquad \forall i\in\mathcal{V}^F \tag{5.36}$$

$$\sum_{o\in O}\textsc{Pres}(x_i^o)\le \sum_{k\in K}\textsc{Pres}(y_i^k) \qquad \forall i\in\mathcal{V}^F \tag{5.37}$$

$$\textsc{First}(\pi^o, x_0^o), \textsc{Last}(\pi^o, x_{N+1}^o) \qquad \forall o\in O \tag{5.38}$$

$$\textsc{First}(\rho^k, y_0^k), \textsc{Last}(\rho^k, y_{N+1}^k) \qquad \forall k\in K \tag{5.39}$$

$$x_0^o : \textsc{IntervalVar}(0, [0,0]) \qquad \forall o\in O \tag{5.40}$$

$$x_{N+1}^o : \textsc{IntervalVar}(0, [H,H]) \qquad \forall o\in O \tag{5.41}$$

$$x_i^o : \textsc{OptIntervalVar}(p_i, [t_i^-, t_i^+]) \qquad \forall i\in\mathcal{V}^C \tag{5.42}$$

$$x_i^o : \textsc{OptIntervalVar}(\xi, [0,H]) \qquad \forall i\in\mathcal{V}^F, o\in O \tag{5.43}$$

$$y_0^k : \textsc{IntervalVar}(0, [0,0]) \qquad \forall k\in K \tag{5.44}$$

$$y_{N+1}^k : \textsc{IntervalVar}(0, [H,H]) \qquad \forall k\in K \tag{5.45}$$

$$y_i^k : \textsc{OptIntervalVar}(\xi, [0,H]) \qquad \forall i\in\mathcal{V}^F, k\in K \tag{5.46}$$

$$\pi^o : \textsc{SequenceVar}(\{x_0^o,\dots,x_{N+1}^o\}) \qquad \forall o\in O \tag{5.47}$$

$$\rho^k : \textsc{SequenceVar}(\{y_0^k,\dots,y_{N+1}^k\}) \qquad \forall k\in K \tag{5.48}$$

The objective function (5.27) maximizes accumulated search reward across the UAV fleet. Constraints (5.28) and (5.29) ensure that UAV and MRV tasks do not overlap in time, using the NoOverlap global constraint. Constraint (5.30) enforces the effects of the consumption and replenishment of energy for each of the UAVs, $o\in O$, on the cumulative function expression variable $\mathcal{Q}^o$ (with a slight abuse of notation, $\Delta^{Euc}_{\textsc{Prev}_{\pi^o}(x^o_i),i}$ returns the Euclidean distance between the visit previous to x^o_i and x^o_i). Constraint (5.31) ensures that each UAV's battery level remains within permissible bounds throughout the schedule, and Constraint (5.32) ensures that when a UAV recharges, it recharges fully. Constraint (5.33) enforces UAV/MRV synchronization on recharge visits. Only one UAV is permitted to visit a search pattern vertex, expressed with Constraint (5.34). Constraints (5.35) through (5.37) dictate the necessary conditions for recharge vertex visitation. Specifically that at most one UAV and at most one MRV can visit a recharge vertex, and that a UAV can only visit if an MRV visits. The depot visits are constrained to be at the start and end of vehicle sequences with Constraints (5.38) and (5.39), respectively. The remainder of the Constraints (5.40) through (5.48) dictate the domains of the decision variables. The alternative resource model contains $|O|\cdot|\mathcal{V}_{0,N+1}|+|K|\cdot|\mathcal{V}^F_{0,N+1}|$ interval variables, $|O|+|K|$ sequence variables, and $|O|$ cumulative function expression variables.

5.5.2.2 Single Resource Model

Our single resource CP model is defined by Constraints (5.49) through (5.66) and follows the single resource transformation technique recently proposed for electric vehicle routing to leverage the homogeneity of UAV and MRV fleets [23], as detailed in Chapter 4. This transformation compactly represents the routes of multiple vehicles of the same type with a single augmented horizon (Figure 5.3). Given a solution to the model, the assignment of tasks to vehicles can be efficiently determined based on the start time of each task.

This technique uses a set of auxiliary depot instance vertices, $\mathcal{H} = \{v_{N+2}, \ldots, v_{N+\max(|O|,|K|)}\}$, to represent end depots of the additional horizon segments. Vertex and edge sets are extended to include these auxiliary vertices (e.g., $\mathcal{H}_{0,N+1} = \{v_0, v_{N+1}\} \cup \mathcal{H}$, $\mathcal{V}_{0,N+1,\mathcal{H}} = \mathcal{V}_{0,N+1} \cup \mathcal{H}$, and $\mathcal{E}' = \{(i,j) \mid i,j \in \mathcal{V}_{0,N+1,\mathcal{H}}, i \neq j\}$). A depot instance, represented as an interval variable, x_i, is assigned with null duration for $i \in \mathcal{H}_{0,N+1}$. These variables have fixed start time σ_i such that $\sigma_0 = 0, \sigma_{N+1} = H, \sigma_{N+2} = 2H$, etc. Then, optional interval variable x_i, for $i \in \mathcal{V}$, represents a visit to vertex v_i by a UAV in the fleet. Similarly, optional interval variable y_i represents a visit to vertex v_i, for $i \in \mathcal{V}^F$, by an MRV. We let sequence variable π represent the sequence of UAV visits, and sequence variable ρ the sequence of MRV visits. Cumulative function expression variable $\mathcal{Q}$ represents the energy level throughout the augmented UAV schedule.

$$\max \sum_{i\in\mathcal{V}^C} \textsc{Pres}(x_i)\cdot r_i \tag{5.49}$$

subject to:

$$\textsc{NoOverlap}(\pi, \{\tfrac{\Delta^{Euc}_{ij}}{s_O} : (i,j) \in \mathcal{E}'\}) \tag{5.50}$$

$$\textsc{NoOverlap}(\rho, \{\Delta^{SP}_{ij} : (i,j) \in \mathcal{E}'\}) \tag{5.51}$$

$$\textsc{ForbidStart}(x_i, \psi_i) \qquad \forall i \in \mathcal{V}^C \tag{5.52}$$

$$\begin{aligned}\mathcal{Q} = {} & \textsc{StepAtStart}(x_0, Q) \\ & - \sum_{i\in\mathcal{V}^C} \textsc{StepAtStart}(x_i, g\Delta^{Euc}_{\textsc{Prev}_\pi(x_i),i} + h) \\ & + \sum_{i\in\mathcal{V}^F_{N+1,\mathcal{H}}} \textsc{StepAtStart}(x_i, [0, Q - g\Delta^{Euc}_{\textsc{Prev}_\pi(x_i),i}])\end{aligned} \tag{5.53}$$

$$\textsc{AlwaysIn}(\mathcal{Q}, [0, |O| \cdot H], [0, Q]) \tag{5.54}$$

$$\textsc{AlwaysIn}(\mathcal{Q}, x_i, [Q, Q]) \qquad \forall i \in \mathcal{V}^F \tag{5.55}$$

$$\textsc{Mod}(\textsc{Start}(y_i), H) = \textsc{Mod}(\textsc{Start}(x_i), H) \qquad \forall i \in \mathcal{V}^F \tag{5.56}$$

$$\textsc{Pres}(y_i) = \textsc{Pres}(x_i) \qquad \forall i \in \mathcal{V}^F \tag{5.57}$$

$$\textsc{First}(\pi, x_0), \textsc{Last}(\pi, x_{N+|O|}) \tag{5.58}$$

$$\textsc{First}(\rho, y_0), \textsc{Last}(\rho, y_{N+|K|}) \tag{5.59}$$

$$x_i : \textsc{IntervalVar}(0, [\sigma_i, \sigma_i]) \qquad \forall i \in \mathcal{H}_{0,N+1} \tag{5.60}$$

$$x_i : \textsc{OptIntervalVar}(p_i, [0, |O| \cdot H]) \qquad \forall i \in \mathcal{V}^C \tag{5.61}$$

$$x_i : \textsc{OptIntervalVar}(\xi, [0, |O| \cdot H]) \qquad \forall i \in \mathcal{V}^F \tag{5.62}$$

$$y_i : \textsc{IntervalVar}(0, [\sigma_i, \sigma_i]) \qquad \forall i \in \mathcal{H}_{0,N+1} \tag{5.63}$$

$$y_i : \textsc{OptIntervalVar}(\xi, [0, |K| \cdot H]) \qquad \forall i \in \mathcal{V}^F, \tag{5.64}$$

$$\pi : \textsc{SequenceVar}(\{x_0, \dots, x_{N+|O|}\}) \tag{5.65}$$

$$\rho : \textsc{SequenceVar}(\{y_0, \dots, y_{N+|K|}\}) \tag{5.66}$$

Objective (5.49) maximizes accumulated search reward. Constraint (5.50) uses the NoOverlap constraint to ensure UAV tasks do not interfere temporally, with consideration for travel times, and Constraint (5.51) enforces the same restriction for MRVs. Constraint (5.52) uses the ForbidStart constraint to ensure that UAV search pattern tasks can only start during search pattern time windows; the set of forbidden starting times for x_i is denoted by $\psi_i = \{0, \dots, |O| \cdot H\} \setminus \bigcup_{\gamma \in \{0,\dots,|O|-1\}} \{\gamma H + t_i^-, \dots, \gamma H + t_i^+\}$. Constraint (5.53) dictates the effects of consumption and replenishment of UAV energy on the cumulative function expression variable $\mathcal{Q}$ (with a slight abuse of notation, $\Delta^{Euc}_{\textsc{Prev}_\pi(x_i),i}$ returns the Euclidean distance between the visit previous to x_i and x_i). Constraint (5.54) ensures that throughout the entire augmented UAV planning horizon, $[0, |O| \cdot H]$, UAV battery level always remains within $[0, Q]$. Constraint (5.55) ensures that when a UAV swaps a battery, the battery is recharged completely. Constraint (5.56) ensures that UAV and MRV recharge tasks are synchronized across augmented horizons using the Mod constraint based on the modulo operation. $\textsc{Mod}(\textsc{Start}(var), H)$ returns the remainder when $\textsc{Start}(var)$ is divided by the planning horizon, effectively yielding the non-augmented start time of the task. Constraint (5.57) enforces that both a UAV and MRV must be present for a recharge. The depot visits are constrained to be at the start and end of vehicle sequences through Constaints (5.58) and (5.59), respectively. The remainder of the Constraints (5.60) through (5.66) indicate the domains of the decision variables. The model contains $|\mathcal{V}_{0,N+1,\mathcal{H}}| + |\mathcal{V}^F_{0,N+1,\mathcal{H}}|$ interval variables, two sequence variables, and one cumulative function expression variable.

5.6 Experimental Evaluation

In this section we conduct an empirical investigation of the proposed routing graph construction pipeline and target search optimization models. We evaluate the relative performance of the models presented, and discuss whether our approaches are reasonable for use in real-time situations. To assess the advantages of using MRVs in tandem with the UAV fleet, we examine optimization results for *non-range constrained* (NRC), *range constrained no recharging* (RC-NC), and *range constrained with recharging* (RC-C) variants of the problem. NRC, when solved optimally, provides an upper-bound on the best

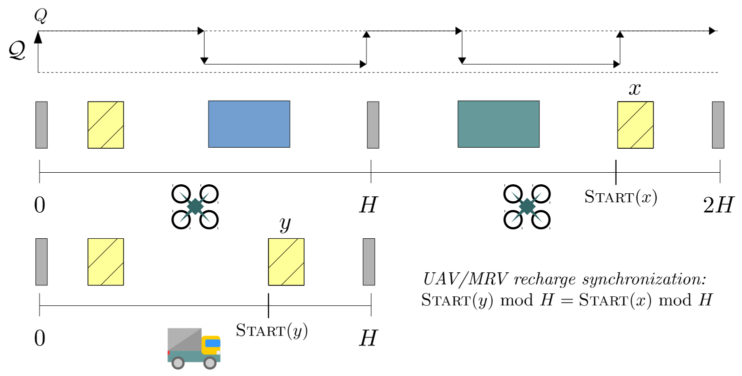

Q
Q
x
0
H
Start(x)
2H
y
0
Start(y)
H
UAV/MRV recharge synchronization:
Start(y) mod H = Start(x) mod H

Figure 5.3: CP model, single resource transformation technique. Two UAVs, $|O| = 2$, with augmented horizon ($2H$). Single MRV, $|K| = 1$. Each UAV conducts one search pattern (blue/green) and is recharged once (yellow) by the MRV.

possible solution to a given problem instance as it allows UAVs to fly without range constraints; in this case, accumulated reward is limited by the temporal aspects of the problem (i.e., time windows, transition times, and planning horizon). RC-NC, when solved optimally, provides a lower bound on the accumulated reward as each UAV has limited battery but is not able to recharge. The RC-C variant includes the fleet of MRVs and represents the range-constrained variant primarily studied in this work.

5.6.1 Setup

The UAV and MRV routing graph construction pipeline is implemented in Python. All shortest path calculations use the NetworkX library [89] and the modified hitting set problem is solved using Gurobi 9.0 via the Python interface. All target search optimization model experiments are implemented in C++ and run on the Compute Canada Niagara computing cluster operated by SciNet (http://www.scinethpc.ca). The cluster runs the Linux CentOS 7 operating system and uses Skylake cores at 2.4 GHz. We use CP Optimizer for the CP models and CPLEX for the MILP model from the IBM ILOG CPLEX Optimization Studio version 12.9. All target search optimization model experiments are single-threaded with default inference settings and a time limit of one hour.

5.6.2 Real-world Instances

Table 5.1: Problem instance details.

Class	Small	Medium	Large
UAVs, $\|O\|$	1	3	5
MRVs, $\|K\|$	1	2	3
Graph vertices, $\|V\|$ (Avg.)	7,645	10,042	16,756
Graph edges, $\|E\|$ (Avg.)	8,839	11,635	19,327

For our simulations, we set UAV parameter values to follow the design specifications of the DJI Matrice 200, a popular and commercially-available rotary wing UAV [193]. UAV speed is set to 23 m/s, maximum range on a full charge is set to 17,940 metres (where energy consumption is assumed to be linear) following flight time with a loaded payload, and search pattern consumption is set to be the

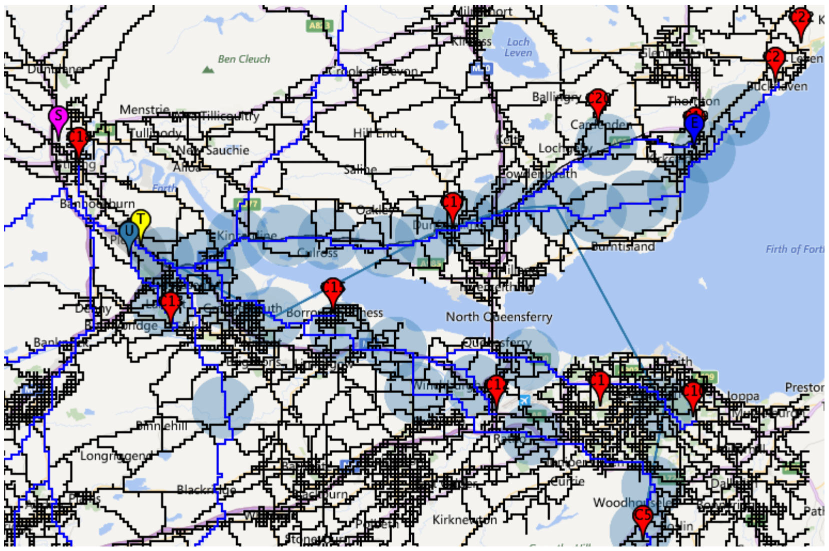


Figure 5.4: Simulation instance problem graph derived from road network in Scotland. Search patterns (blue circles), destinations (red), target LKP (yellow), target actual start and end (purple/blue), UAV position (light blue).

equivalent of the UAV traveling 3,450 metres. We investigate two battery swap durations: instantaneous and 30 seconds, following existing research on automated battery swap technology [154]. We generate problem instances and run simulations for three problem classes: small, medium, and large as illustrated in Table 5.1. Each problem class varies the number of UAVs and MRVs, as well as the scale of the road networks used. To construct our benchmark set, 20 instances are generated for each class, starting from real-world road networks in Scotland, upon which the problem graph is constructed (Figure 5.4), and ending with the routing graph via our pipeline. Each instance considers a different road network discretized with $\delta = 300$ metres, resulting in the average graph characteristics noted in Table 5.1. As is detailed in the table, the scale of the road networks used for large instances is more than twice the size as those for small instances.

5.6.3 Simulation Results and Discussion

In this section we detail results for the routing graph construction pipeline and target search optimization.

5.6.3.1 Routing Graph Construction

The first phase of routing graph construction is running the MCS to identify search pattern locations and time windows, finalizing the problem graph, G. Following the existing approach [19], this phase was found to take an average of 5.2 seconds with a standard deviation of 0.6 seconds across all problems.

The second phase constructs the routing graph, $\mathcal{G}$, from the problem graph, G. For these experiments, we set filtering and coverage parameters, ϕ and R, respectively, to 25% of maximum UAV range, a value empirically found to yield sufficient filtering while providing strategic areas for recharge vertex placement. Due to numerous shortest path calculations and the use of IP in ModifiedHittingSet, this step is quite sensitive to the size of the problem graph. For our experiments, the small and medium instances take, on average, roughly a minute to construct the routing graph, while the large instances can exceed

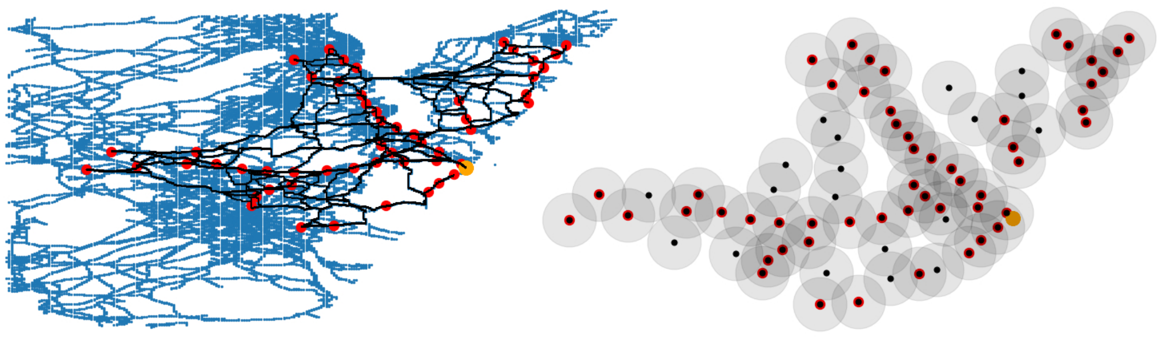


Figure 5.5: UAV and MRV routing graph construction. Real-world road network and shortest paths between pairs of search patterns (left). Discretized routing graph network with recharging vertices (black) and search pattern vertices (red).

five minutes. This suggests that, while the proposed pipeline is reasonable for use in real-time, small-to-medium sized situations, additional optimizations are required for large problems (e.g., using a heuristic to find the centres).

For each problem class, the produced routing graph, $\mathcal{G}$, varies in size. Small instances had routing graphs with an average of 44.4 vertices, medium instances with an average of 62.1 vertices, and large instances with an average of 102.1 vertices. Depending on the underlying topology of the instance, the routing graphs were found to include up to 20 strategically placed (i.e., not coinciding with a search pattern location) vertices to enable synchronized recharging.

Figure 5.5 illustrates parts of the routing graph construction process over real-world road network data. The left side of the figure details the short paths (black) over the road network (blue) between search pattern locations (red). The orange circle in this example represents the last known position of the target. The right side of the figure illustrates the discretized routing graph as a result of our pipeline. In this visualization, the black vertices are recharging vertices and the red vertices are search pattern vertices. The target search optimization models, as covered in the next section, produce coordinated search plans through this graph.

5.6.3.2 Target Search Optimization

In this section we discuss the performance of the target search optimization models. We assess solution quality over time in the context of lower and upper bounds on accumulated fleet reward to gauge the performance of the various approaches. Initially, we investigate performance when UAV speed follows realistic values, after which we conduct similar experiments with increased UAV speeds.

With the constructed routing graphs, we present target search simulation results using realistic DJI Matrice 200 speeds. These results are presented in Figure 5.6 (top). Each of the plots summarizes average results for the problem classes (e.g. small, medium, large). Each plot provides a time-based analysis capturing how closely the approaches approximate the optimal accumulated reward of the NRC variant. The NRC variant can be solved to optimality using MILP as it omits all of the energy constraints/variables and MRVs, making the problem considerably easier to solve.[2] The RC-C variant, however, is very rarely solved to optimality by any of the techniques (specifically, optimality is proven

[2]For similar reasons, the RC-NC variant can also be solved to proven optimality with MILP.

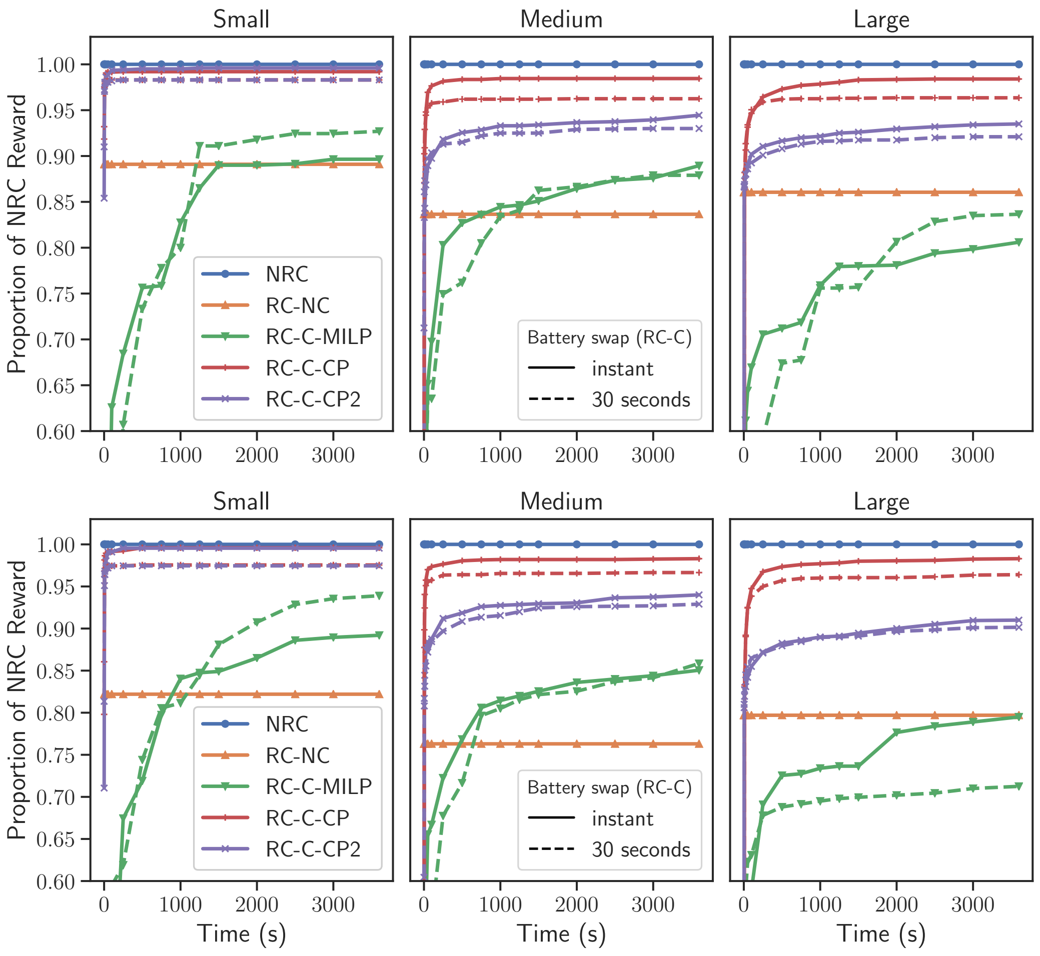


Figure 5.6: Solution quality over time. Comparison of NRC, RC-NC, and RC-C optimization performance. 'CP' refers to the alternative resource CP model and 'CP2' the single resource CP model. All experiments were given one hour of runtime. Plots represent the proportion of reward compared to the optimal reward yielded by solving the NRC problem variant. RC-C results for instantaneous battery swap, and battery swap with duration of 30 seconds. *Top:* Experiments following DJI speed specifications. *Bottom:* Experiments with UAV speed increased by 30%.

for only one small instance by the MILP and single resource CP approaches). At each time step, the average objective for a given method, as a proportion of the NRC objective, is plotted. For the RC-C experiments, plots with solid lines reflect optimization runs with instantaneous battery swaps, while those with dashed lines use a duration of 30 seconds.

In Figure 5.6 (top), the RC-NC variant (orange plot) indicates that deploying the UAV fleet on a single charge can attain from 85% to 90% of the NRC reward upper bound (blue plot). Therefore, at realistic UAV speeds, there is less than 10-15% reward improvement to be gained by leveraging the MRVs. This observation is largely driven by the fact that, as a property of instance generation, search patterns closer to the target's LKP are assigned larger rewards and, additionally, are likely to be reachable with a single charge. Candidate search areas further away from the LKP have diminished reward values to reflect a reduction in confidence that the target will be reacquired. We further expand on this observation in Section 5.6.3.3.

For strategies using MRVs (i.e., RC-C-MILP, RC-C-CP, and RC-C-CP2), it is clear that the CP models (red and purple plots) are able to find the highest reward solutions overall, with average rewards exceeding the RC-NC baseline in less than 10 seconds of solver runtime, even for the largest problems, and approaching those of the NRC variant. The alternative resource CP model (red plots), in particular, illustrates the strongest performance on these experiments, with accumulated rewards above 95% of the NRC bound. The single resource CP model (purple plots) exhibits weaker performance than the alternative resource model, though outperforms MILP by a fair margin.

The MILP method (green lines) exhibits reasonable performance for small instances, but only modestly improves on the RC-NC variant bound for medium class problems and is unable to improve upon it for large problems. Battery swap duration has relatively little effect on both CP approaches for these experiments. For medium and large problems, the solution quality is noticeably worse when 30 second battery swaps are used (versus instantaneous swaps), however, this effect only impacts the proportion of accumulated reward by 1-2%. The impact of swap duration on the MILP approach appears to be more pronounced, but does not, overall, drastically impact results. This finding is attractive as it indicates accumulated reward under realistic battery swap duration is not significantly eroded compared to an idealized instantaneous swap scenario.

The superiority of the alternative resource model over the single resource model is likely driven by a couple of factors. First, the fleet sizes, even in the largest problems with five UAVs and three MRVs, are fairly modest. This is in contrast to the problems studied in Chapter 4 where the size of the optimal electric vehicle fleet was unknown *a priori*, and so an upper bound of vehicles was used for modeling purposes. Smaller fleets reduce the effectiveness of the single resource transformation as the advantages of leveraging vehicle homogeneity become more muted. Additional analysis on the effectiveness of these methods as the problem scales (with respect to both fleet size and graph size) is presented in Section 5.6.3.4. Second, the presence of multiple fleet types requires synchronization with the Mod operator in the single resource model, attempting to align the start times of pairs of interval variables with augmented start time domains. In the case of large fleets, the domain augmentation is significant, likely resulting in less efficient inference. The alternative resource model, on the other hand, leverages the StartAtStart constraint, a standard precedence constraint within the CP paradigm.

The results in Figure 5.6 (bottom) assess performance when UAV speed is increased to 30 m/s. We can see from the figure that the difference between the NRC "ideal" objective values and RC-NC lower bound objective values becomes more pronounced (18-25%) as UAV speed is increased. Faster UAVs are able to satisfy the time windows of more search pattern locations, thus improving the NRC objective, while still remaining range constrained in the RC-NC variant. The impact of battery swap duration on the CP approaches remains relatively small, while the MILP approach exhibits more pronounced variation, particularly for large problems. The primary assessment of the optimization methods remains consistent: the CP approaches are dominant, with the alternative resource CP model providing the best performance. This performance advantage is particularly apparent for large problems where the CP approaches finds solutions of 90-97% of the NRC rewards, while MILP struggles to improve over the RC-NC lower bound of 75-80%.

MILP is known to have a weak linear relaxation when constraints with large disjunctive constants ('big-M' constraints) are used, as in Constraints (5.15) through (5.21) of the MILP model. Furthermore, the augmented graph MILP formulation often results in network sizes that negatively impact the performance of the approach. As detailed in Chapter 4, recharging path multigraph formulations offer one way

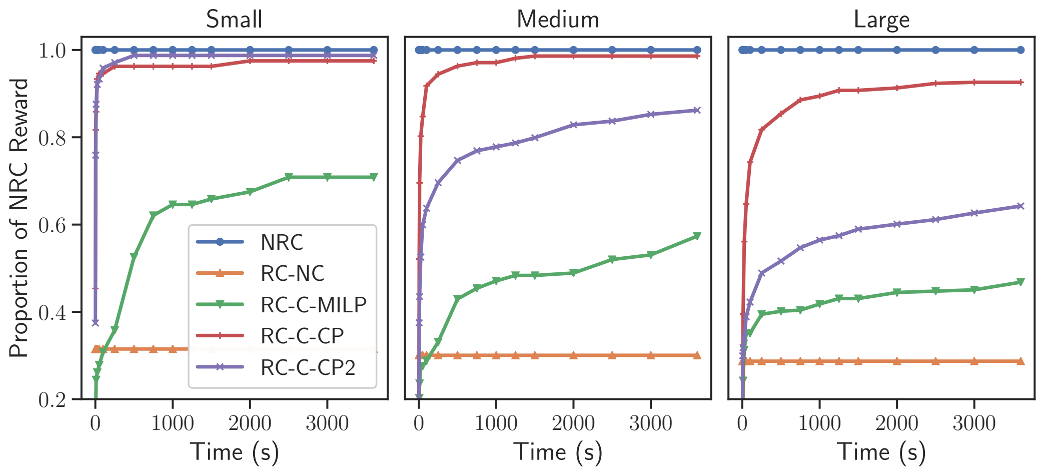


Figure 5.7: Solution quality over time, unit rewards. Comparison of NRC, RC-NC, and RC-C optimization performance. 'CP' refers to the alternative resource CP model and 'CP2' the single resource CP model. All experiments given one hour of runtime. Plots represent proportion of searches compared to the optimal searches yielded by solving the NRC problem variant. RC-C results for instantaneous battery swap only. Experiments following DJI speed specifications.

to alleviate this issue, however, given that the MRVs are mobile recharging stations (whereas in standard electric vehicle routing the recharging stations are static), the multigraph pre-processing schemes for traditional electric vehicle routing are not applicable. Possible techniques that can be investigated to improve upon the baseline MILP approach include branch-and-price/branch-and-cut decompositions that have been applied to problems with similar characteristics [52, 149].

Overall, simulation results suggest that the inclusion of a fleet of MRVs can bolster UAV fleet search performance over a single charge bound, with cumulative reward values often exceeding 90% of the best-case NRC values. Furthermore, the speed at which the CP approaches are able to find solutions exceeding the quality of the RC-NC baseline suggests that our overall approach (i.e., routing graph construction with target search optimization route planning) is promising for use in real-time, small-to-medium sized target search situations.

5.6.3.3 Unit Rewards

As mentioned in the previous section, the seemingly small difference between the accumulated reward of the RC-NC lower bound and the RC upper bound is largely driven by the assignment of larger reward values to search patterns closer to the LKP (and smaller to those further away). To validate this observation, we conduct an additional set of experiments using unit reward values (i.e., $r_i = 1, \forall i \in \mathcal{V}^C$) in the objective function. These experiments are summarized in Figure 5.7.

The primary observation from these experiments is that the difference in proportion of NRC reward (where unit rewards are now used) is significantly larger than those seen in the experiments detailed in Figure 5.6. Specifically, the gap between the NRC and RC-NC bounds is roughly 70%, whereas in the original experiments this value hovered between 10 and 15%. This indicates that the number of search patterns the fleet of UAVs can reach when constrained by range and unable to recharge is only roughly 30% of those that can be reached when the fleet is not range constrained. The relative performance of the RC-C methods remains consistent with previous experiments: the alternative resource model exhibits

Table 5.2: Problem instance details. Scaling experiments.

Class	x2	x3	x4
UAVs, $\lvert O \rvert$	10	15	20
MRVs, $\lvert K \rvert$	6	9	12
Graph vertices, $\lvert V \rvert$ (Avg.)	16,757		
Graph edges, $\lvert E \rvert$ (Avg.)	19,327		

superior results, with the single resource model outperforming the MILP approach.

5.6.3.4 Scaling of Approaches

We conduct an assessment of the performance of the CP approaches by modifying the routing graph construction pipeline of the 'Large' instances from the first set of experiments, and increasing fleet sizes. The summary statistics of these new experiments are given by Table 5.2. Specifically, we reduce the filtering and coverage parameters, ϕ and R, respectively, to 10% of maximum UAV range. As a result, the final routing graphs contain roughly 2-3 times as many vertices as in the baseline experiments. Further, we increase the UAV fleet sizes by multiplying the number of UAVs and MRVs in the 'Large' instances for fleets of size 10 UAVs/6 MRVs (x2), 15/9 (x3), and 20/12 (x4). The MILP approach was omitted from these experiments due to its considerably poor scaling in our initial experiments. For these experiments, the original reward values are used (i.e., non-unit rewards).

The results of our scaling experiments are illustrated in Figure 5.8. The difference between optimal accumulated reward between the NRC and RC-NC variants is roughly 20-25%, indicating substantial potential benefit by integrating the fleet of MRVs. As shown in the leftmost sub-figure (x2 experiments), the single resource CP approach (RC-C-CP2, red) provides superior performance in the first 100 or so seconds, before it is overtaken by the alternative resource CP approach (RC-C-CP, green). Notably, the single resource approach outperforms the RC-NC lower bound very quickly, suggesting it is rapidly able to find solutions leveraging the MRV fleet. As the fleet size increases, the point at which the alternative resource approach produces higher quality solutions (on average) than the single resource approach is pushed later and later. Specifically, this occurs at roughly 200 seconds for the x3 instances and 215 seconds for the x4 instances.

This particular set of experiments illuminates the value of the single resource CP approach for real-time deployment. While the alternative resource approach is able to produce superior solutions when given sufficiently long runtime budgets, the single resource approach is able to efficiently produce non-trivial routing solutions. Further investigation yielded that this is largely driven by the time it takes for the alternative resource model to store the model in memory and conduct various pre-processing routines. Target search deployments that must wait upwards of 200 seconds (in excess of three minutes) for search routes to be produced are unlikely to be successful, even less so when combined with the routing graph construction pipeline time as detailed in Section 5.6.3.1.

5.6.4 Discussion

Our empirical evaluation indicates the superiority of the CP approaches over MILP for route-planning in this problem domain. While both CP approaches outperform MILP in terms of solution quality over time, the alternative resource model, specifically, yields the best performance out of the methods tested.

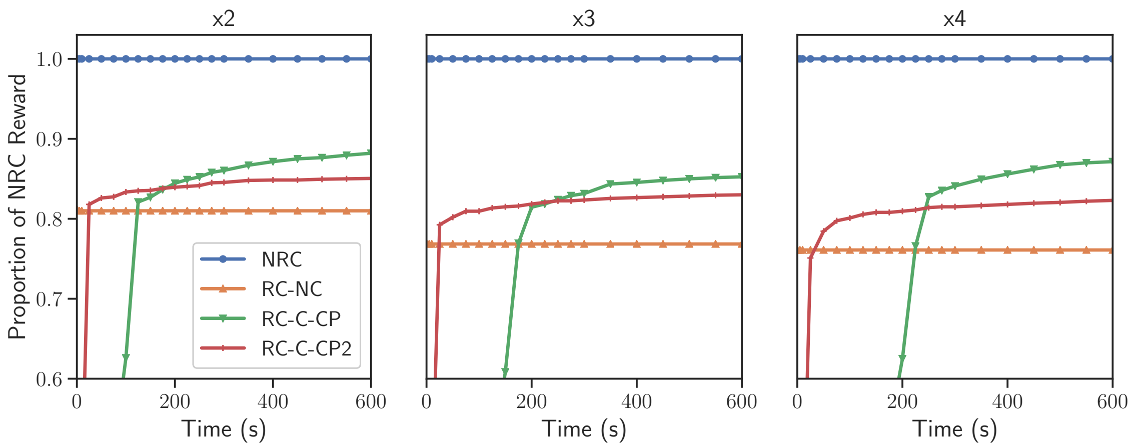


Figure 5.8: Solution quality over time, scaling experiments. Comparison of NRC, RC-NC, and RC-C (CP models only) optimization performance. ‘CP’ refers to the alternative resource CP model and ‘CP2’ the single resource CP model. All experiments given ten minutes of runtime. Plots represent proportion of reward compared to the optimal reward yielded by solving the NRC problem variant. RC-C results for instantaneous battery swap. Experiments follow DJI speed specifications.

Our first set of experiments indicated that the inclusion of MRVs can significantly bolster the ability of the UAV fleet to achieve accumulated rewards near non-range constrained upper bounds. Interestingly, an investigation with the MILP formulation on its own would not have yielded this insight, as the approach struggled to produce rewards above the range-constrained, no recharging lower bound for large problems. The second set of experiments with unit rewards, in particular, illuminates the hampered ability of commercially-available UAVs standalone to conduct range-constrained searches across road networks on the scale of these problem instances. In this case, lower bounds indicate the range-constrained fleet under a no-recharging condition is only able to conduct 30% of the search patterns that could be conducted without range constraints. This observation provides a strong argument for the inclusion of MRVs for support roles in the search fleet. The final experiments indicate that, in the case of real-time deployments where search plans must be rapidly produced, there is benefit to using the single resource transformation over the alternative resource approach, particularly as fleet size increases.

Overall, this research indicates the viability of cost-effective, commercially available UAVs for large-scale target search missions over the use of significantly more costly fixed-wing military-grade designs, with the caveat that a fleet of MRVs must accompany the UAVs to make up for range restrictions.

5.7 Conclusions

In this chapter we studied a range-constrained variant of the multi-UAV target search problem, assessing the viability of commercially available UAVs for target search missions. We proposed a pipeline for representing the problem over real-world road networks and solved the problem using mixed-integer linear programming (MILP) and constraint programming (CP). Our empirical results indicate that the CP approaches are able to provide better solutions than MILP, overall, with the alternative resource CP approach providing superior results for the real-world instances generated. Further investigation revealed that as the problem size scales, the single resource CP approach offers attractive performance for smaller run-time budgets indicating it may be preferred for real-time deployments requiring fast solutions. Overall, our simulation-based experimentation indicates that the use of a fleet of MRVs can significantly improve the accumulated reward of range-constrained UAV fleets over large-scale, real-world road networks.

Chapter 6

Social Robot Routing in Retirement Homes

In the previous chapter of this dissertation, we provided a solution framework for solving multi-UAV target search problems on real-world road networks with ground-based mobile recharging vehicles. Our approach investigated the use of both constraint programming (CP) and mixed-integer linear programming (MILP) for producing search plans, adapting the modeling techniques introduced in Chapter 4 for heterogeneous fleets with synchronization constraints.

Recognizing the similarities between electric vehicle routing and robot scheduling problems, in this chapter, we adapt and apply modeling techniques for electric routing to a complex, real-world multi-robot task allocation and scheduling problem that takes place in a retirement home setting. We call this problem the *social robot routing problem in retirement homes* (SRRP-RH). The problem involves a large number of complex, real-world side constraints relating to vehicle synchronization and task precedence. Equipped with the modeling strategies outlined in the previous chapters, we develop both MILP and CP approaches to the problem.

Contributions to Co-Authored Work. The research in this chapter significantly extends our work published in the *Proceedings of the Twenty-Second International Conference on Principles and Practice of Constraint Programming* [25], and compares the computational performance of the proposed methods to existing approaches from the literature [199, 200]. Co-authors Goldie Nejat and J. Christopher Beck supervised this work.

6.1 Introduction

The progressive aging of populations has important implications in a number of societal areas, including health and social care services for the elderly [50]. Due to this demographic trend, the number of seniors residing in retirement homes has seen a dramatic increase [71]. When combined with a relative reduction in the working age population, greater pressures on the quality of elderly care infrastructure will risk deterioration in the provision of medical services, daily assistance, social interaction, and overall quality of life for residents. As a potential solution to this problem, the role of autonomous robotics in health-care has been investigated for a number of decades, though primarily with respect to robots assisting physical rehabilitation [133]. The design and deployment of socially assistive robots for retirement home applications and elderly care is a more recent development [16]. Such social robots aim to alleviate workforce pressures associated with the daily operation of retirement homes and work to give assistance through the autonomous facilitation of cognitively and socially stimulating activities.

In this chapter, we study the planning and scheduling of a team of socially assistive mobile robots in a retirement home environment. The problem can be posed as an instance of multi-robot task allocation (MRTA) and involves reasoning about which tasks should be conducted (i.e., planning), as well as which robot should facilitate each task and at what time (i.e., scheduling). The research in this chapter is part of a larger project that involves robots providing social and cognitive stimulation through the facilitation of bingo games for multiple residents and telepresence sessions between residents and their family members and friends. That long-term study on the deployment of intelligent human-like mobile robots in retirement homes focuses on the development and use of innovative robotic technologies to provide person-centered cognitive interventions to improve the quality of life of elderly adults [143, 144].

Following our efforts for electric vehicle routing problems, as outlined in Chapter 4, we investigate the use of MILP and CP as part of a task planning system that must allocate, schedule, and facilitate single and multi-resident leisure activities throughout the course of the day, while adhering to resident availability. We call this problem the *social robot routing problem in retirement homes* (SRRP-RH). Previous research on task planning and scheduling for this application [25, 199, 24] extended a single robot version of the problem [145, 28] to the multi-robot case, providing the initial investigation for the application of off-the-shelf techniques, including MILP, CP, AI planning, and a problem-specific decomposition [199]. In this work, we propose MILP and CP approaches based on techniques from the vehicle routing literature.

Contributions of chapter. Thus far, efforts to model and solve SRRP-RH have been disconnected from related literature bodies such as electric vehicle routing. As such, the primary contributions of this chapter are as follows:

1. We classify SRRP-RH according to taxonomies from both the MRTA and VRP literatures and comment on the complexity of the fleet distance minimization variant of the problem.
2. We propose novel MILP and CP models based on augmented graph modeling techniques from the electric vehicle routing literature. These models include consideration for user travel in the environment. Our MILP model leverages vehicle symmetry via a compact formulation to significantly reduce model size compared to a previously proposed model.
3. We compare our approaches to those that have been previously published on the original version of the SRRP-RH that relaxes some constraints on user travel. We demonstrate, through empirical assessment, that our methods are competitive with, and often outperform, existing approaches to the problem, with respect to solution feasibility and quality for a variety of objective functions.
4. We demonstrate, via modeling and experimental analysis, that our proposed approaches are able to capture user travel within the environment, an important aspect of SRRP-RH that helps to ensure that produced routes are achievable in practice.

Outline of chapter. The rest of the chapter is organized as follows. Section 6.2 formally defines the SRRP-RH, alongside a classification with respect to taxonomies from the MRTA and VRP literatures.

Section 6.3 summarizes relevant related work and Section 6.4 details previously proposed MILP and CP models for the studied problem and identifies limitations/weaknesses in the approaches. Section 6.5 outlines the implementation of routing-based modeling strategies for the problem while Section 6.6 details the graph representation. Sections 6.7 and 6.8 present our CP and MILP models for the problem. Section 6.9 presents our experimental results and analysis. Section 6.10 provides concluding remarks.

6.2 Problem Definition

Each instance of the SRRP-RH considers a set of mobile social robots, $r \in R$, that must facilitate a set of human-robot interaction (HRI) activities, $a \in A$, with retirement home residents (i.e., users), $u \in U$, throughout the course of a single day (7:00 AM – 7:00 PM) planning horizon, $t \in T$. In this section, we define the primary problem components, including the retirement home environment, user characteristics, the fleet of social robots, the HRI activities, and the problem inputs and objective functions. The problem definition follows previous work [199] with adjustments made to notation to assist with model presentation. All notation used for the problem definition is provided in Table 6.1.

6.2.1 Environment

Each instance of SRRP-RH has an environment defined as follows.

Definition 6.2.1. *Retirement home environment. The environment is given by,* $\mathcal{L} = \langle L, \mathcal{D} \rangle$*, where* L *is a set of locations of interest and* $\mathcal{D}$ *is a distance matrix. Each element,* $\ell \in L$*, represents an area in the retirement home and is characterized by an* (x, y) *coordinate in Euclidean space. These areas include the robot depot, resident personal rooms, leisure areas, games room, and the set of recharge stations,* F*. The distance matrix,* $\mathcal{D} = \{d_{\ell,\ell'} \mid (\ell, \ell') \in L \times L\}$*, defines travel distances (in metres) between all pairs of locations. A sample facility environment for a problem instance with five users is illustrated in Figure 6.1.*

Each recharging station, $f \in F$, in the environment, including the robot depot, is unary capacity (i.e., charges one robot at a time). Non-depot recharging stations are spatially distributed throughout the environment. The specific number of such stations and their locations are parameters of the problem. In the original problem definition [199], robots start and end the day at any of the recharge stations across the environment. In our problem definition, social robots must start and end the day at the robot depot, an area where they can be safely stored overnight.

6.2.2 Users

The problem considers a set of users, $u \in U$, that reside in the retirement home, defined as follows.

Definition 6.2.2. *Retirement home residents. Each resident,* $u \in U$*, is characterized by the following tuple:* $\langle \Lambda^u, \nu^u, \ell^u, g^u_{min}, g^u_{max} \rangle$*, where* Λ^u *denotes the calendar for user* u*,* ν^u *denotes user movement speed in the environment,* ℓ^u *the location of the user's personal room, and* $[g^u_{min}, g^u_{max}]$ *lower and upper bounds on user bingo game participation throughout the day. Each element,* $\lambda \in \Lambda^u$*, is characterized by* $\langle \ell_\lambda, \theta_\lambda, s_\lambda, \mu(\lambda) \rangle$*, where* ℓ_λ *is the location,* θ_λ *the start time,* s_λ *the duration, and* $\mu(\lambda)$ *is a function that returns the user's availability during* λ*: 1 if the user is available for interaction and 0 if unavailable.*

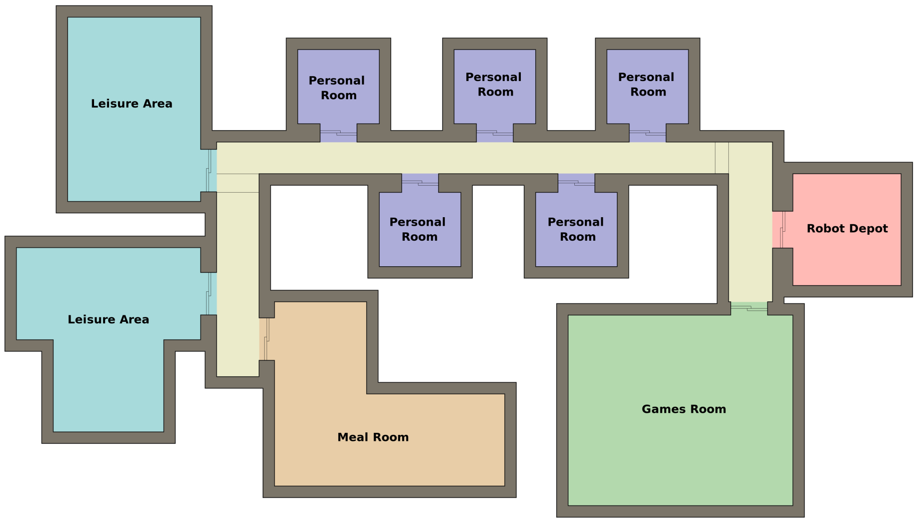


Figure 6.1: Sample SRRP-RH facility environment containing personal rooms for five users (purple), a games room (green), a meals room (orange), a robot depot (red), and two leisure areas (teal). Rooms are color coded to match with user calendars (Figure 6.2) and robot schedules (Figure 6.4).

Each user calendar, Λ^u, consists of time intervals specifying the user's anticipated locations and availability (available for HRI or not). An example of these user calendars is visualized for a problem instance involving five users in Figure 6.2. Each of these intervals is associated with a location within the environment that the user expects to be in during that time interval, a start time, and a duration. Each user's daily calendar includes one hour for breakfast (8:00 – 9:00 AM), lunch (12:00 – 1:00 PM), and dinner (5:00 – 6:00 PM), located in the meal room, as well as additional user-specific commitments (e.g., art classes, family visits, appointments), located either in the user's personal room or in one of the leisure rooms, with duration ranging from 30 minutes to one hour. During these specific intervals, the user is not available for HRI activities (denoted with diagonal lines in the figure). During all other calendar intervals, the user is available for an HRI activity and is anticipated to be in either their personal room or a leisure room (solid colored intervals).[1]

Each user, $u \in U$, also expresses their preference for their minimum, g^u_{min}, and maximum, g^u_{max}, bingo game participation during the day. Users may also request telepresence session activities that must be facilitated in their personal room at a time that the user is available.

Finally, an estimate of user movement speed, ν^u, in metres per minute, is used to approximate user travel time between locations. We note that, as the problem was originally modeled [199], user travel time between locations was not taken into consideration. The inclusion of this characteristic improves the viability of produced solutions by affording the users time to move among locations in the environment.

6.2.3 Social Robots

The retirement home has a fleet of social robots, $r \in R$, that is responsible for facilitating the HRI activities. To facilitate an HRI activity, the social robot and participating user(s) must be at the desired location at the scheduled time, after which they are both unavailable for the duration of the HRI. Each social robot type in the fleet is defined as follows.

[1]Recent work relaxes this assumption and includes a person search algorithm to identify the true location of users [24].

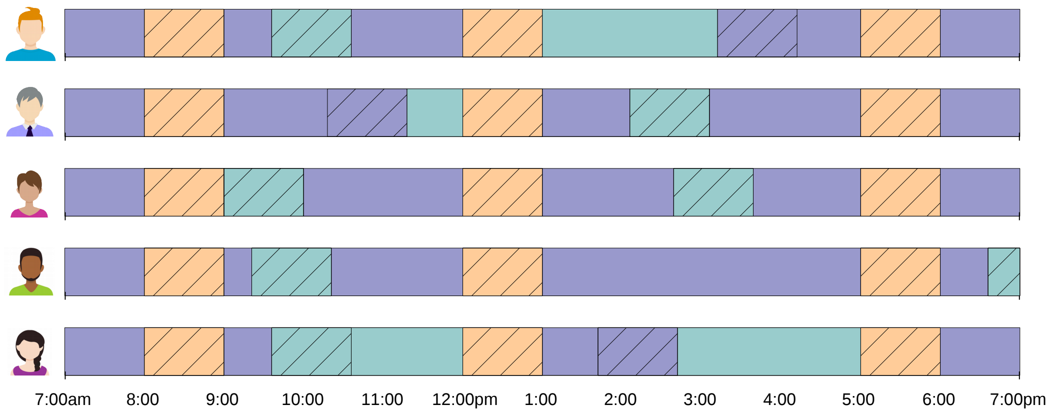


Figure 6.2: User availability calendars for SRRP-RH instance with five users. Color coding indicates the location of calendar block and matches the environment visualization in Figure 6.1 (i.e., a purple interval indicates the user will be in their personal room). Calendar blocks with diagonal patterns indicate user is unavailable for interaction (busy). During solid colored calendar blocks, users can be interacted with either at their calendar-specified location, or in another location (e.g., for a scheduled bingo game in the games room).

Definition 6.2.3. *Social robot types. Each robot type, $k \in K$, is characterized by the following tuple: $\langle \Upsilon(k), \nu^k, h^k, g^k, Q^k, \mathcal{O}^k \rangle$. Function $\Upsilon(k) \in \{1, 2, \ldots, |R|\}$ returns the quantity of robot type $k \in K$ in the fleet such that $\sum_{k \in K} \Upsilon(k) = |R|$. Parameters ν^k, h^k, g^k, and Q^k represent the movement speed, movement energy consumption rate (per unit distance), energy recharge rate (per unit time) and maximum battery capcacity of robot type $k \in K$. Set $\mathcal{O}^k = \{o_a^k : a \in A\}$ denotes the energy consumption rate of robot type $k \in K$ for facilitating task $a \in A$.*

At the start of the day (7:00 AM), each robot is positioned at the robot depot (i.e., the red area in Figure 6.1). Robots of type $k \in K$ start with maximum energy capacity, Q^k. Robots of different types have different travel speeds, ν^k, and energy consumption rates per unit distance, h^k. Throughout the day, robots are able to travel to recharge station locations to recharge. Any recharge station, $f \in F$, can be visited at most n_f times across the robot fleet. Robots recharge their energy linearly at constant rate g^k energy units per unit time. Recharge tasks have their duration bounded by the interval $[0, Q^k/g^k]$. The type of a particular robot $r \in R$ is retrieved using function $\psi(r)$.

6.2.4 Activities

The problem considers individual and group HRI activities, as visualized in Figure 6.3, each requiring a single social robot facilitator. The set of HRI activities is given by A, consisting of telepresence sessions, S, bingo games, G, and bingo game reminders, M, such that $A = S \cup G \cup M$. HRI activities are defined as follows.

Definition 6.2.4. *HRI activity. Each HRI activity, $a \in A$, is characterized by the following tuple: $\langle s_a, \zeta(a), \gamma(a), \alpha(a), \chi(a), P_a \rangle$. The duration of an HRI activity $a \in A$ is given by s_a. The set of candidate locations for activity a is defined by $\zeta(a) = \{\ell \mid (a, \ell) \in \zeta\} \subseteq L$. The robot types that can facilitate activity a are given by $\gamma(a) = \{k \mid (a, k) \in \gamma\} \subseteq K$. The set of users that can be potentially involved in an activity is given by $\alpha(a) = \{u \mid (a, u) \in \alpha\} \subseteq U$. Whether an activity is mandatory or not is given by*

Parameter	**Description**
$u \in U$	Set of human users
$r \in R$	Set of mobile social robots
$k \in K$	Set of mobile social robot types
$t \in T$	Planning horizon, set of time points t with length $\|T\|$
$\ell \in L$	Set of locations in the environment
$f \in F$	Set of recharging stations
n_f	Maximum visits to each recharge station in solution
$a \in A = \{S \cup G \cup M\}$	Set of all HRI activities
$a \in S \subseteq A$	Set of telepresence HRI activities
$a \in G \subseteq A$	Set of bingo game HRI activities
$a \in M \subseteq A$	Set of bingo game reminder HRI activities
$a \in C$	Set of robot recharge activities
$\Lambda^u \subseteq \Lambda$	Calendar for user $u \in U$
g^u_{min}, g^u_{max}	User $u \in U$ bingo preferences, min and max participation
g_{min}, g_{max}	Bingo game min and max participation (across all users)
$\omega_{min}, \omega_{max}$	Bingo game reminder, min and max proximity to game
ν^u	User movement speed (metres per second)
ν^k	Robot type k movement speed (metres per second)
h^k	Travel energy consumption (per unit distance), robot type k
o^k_a	HRI consumption (per unit time) for activity a robot type k
g^k	Energy recharging rate (per unit time), robot type k
Q^k	Maximum energy capacity of robot type k
s_a	Duration of HRI activity $a \in A$
$d_{\ell,\ell'}$	Distance between location ℓ and ℓ' in the environment
$p \in P_a$	Set of time windows during which activity a can be facilitated
$\psi : R \to K \mid \psi(r) = k$	Function (surjective); maps robots to robot types
$\Upsilon : K \to \mathbb{Z}^+ \mid \Upsilon(k) \in \{1, \ldots, \|R\|\}$	Function (general); maps robot types to a quantity
$\chi : T \to \{0, 1\}$	Function (injective); maps HRIs to their optionality
$\gamma \subseteq A \times K$	Relation (binary) betw. HRI activities and robot types
$\zeta \subseteq A \times L$	Relation (binary) betw. HRIs and candidate locations
$\alpha \subseteq A \times U$	Relation (binary) betw. HRIs and human users

Table 6.1: SRRP-RH problem definition notation.

$\chi(a) \in \{0, 1\}$*, where 1 indicates the activity is mandatory, and 0 it is optional. The set* P_a *defines the time windows during which activity* $a \in A$ *can be scheduled, where each window,* $p \in P_a$*, is defined by a lower bound,* $\underline{p}$ *and upper bound,* $\bar{p}$*.*

In this work, the candidate location for bingo game activities contains only the games room and for telepresence activities only the user's personal room (i.e., one location each). Furthermore, all telepresence HRIs are mandatory, while the remaining HRIs (bingo games and reminders) are optional. In the original problem definition [199], all robots could facilitate all tasks. Following real-world use cases, we expand this problem definition to include heterogeneous robot capabilities.

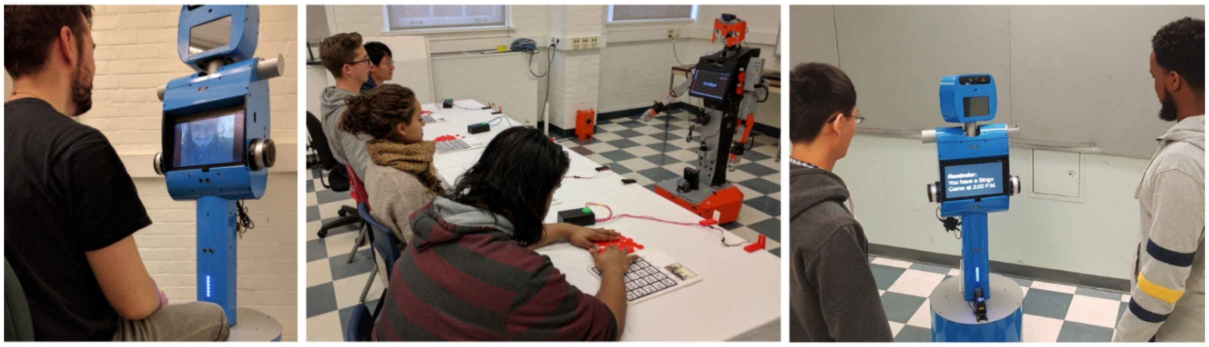


Figure 6.3: Human-robot interaction (HRI) activities with users and heterogeneous robot fleet [24]. Telepresence session (left), bingo game (middle), and bingo game reminder (right).

6.2.4.1 Telepresence Sessions

During the day, a set of telepresence sessions, S, must be held with the users that request them. A user can request multiple telepresence sessions in a day and all of these telepresence sessions must be performed. For privacy, these sessions must take place in the personal room of the associated user (i.e., one candidate location, $|\zeta(a)| = 1, \forall a \in S$, the purple areas in Figure 6.1) and each has a duration of $s_a = 30$ minutes, $\forall a \in S$. Telepresence activities can be scheduled in a user's personal room even if the user is in another location, provided that they are free during the scheduled time and have sufficient time to travel to their personal room. Finally, a time window (or multiple time windows) during which telepresence activities can be held is specified, where each window is denoted by $p \in P_a, \forall a \in S$.

6.2.4.2 Bingo Games

A bingo game, $a \in G$, is a group HRI activity held in the games room (i.e., one candidate location, $|\zeta(a)| = 1, \forall a \in G$) that involves a single robot and multiple users. Each bingo game has a duration, $s_a = 60$ minutes, $\forall a \in G$. In addition to the typical task characterization, bingo games have associated minimum, g_{min}, and maximum, g_{max}, participation to ensure that: i) the games have sufficient participation, and ii) their participation adheres to the games room capacity. A user's participation in a bingo game is optional and any user can participate in any game. The group of users that participate in a bingo game is not known *a priori*, and is a decision variable to be determined during the optimization. Finally, a time window (or multiple time windows) during which bingo games can be held is specified, where each window is denoted by $p \in P_a, \forall a \in G$.

6.2.4.3 Bingo Game Reminders

Each user that participates in a bingo game must be reminded of their involvement prior to the game itself. A bingo game reminder activity, $a \in M$, is a one-to-one HRI activity between a social robot and a user. The set M consists of an HRI activity for each user for each bingo game, for a total of $|M| = |U| \times |G|$ activities. The robot that reminds the user does not need to be the same robot that facilitates the bingo game. A reminder scheduled for a user will be delivered wherever the user calendar specifies they will be at that time, provided they are available for interaction (solid colored blocks in Figure 6.2). Since users are not to be interrupted during meal times, reminders are delivered either in the user's personal room or one of the leisure rooms (i.e., multiple candidate locations, $|\zeta(a)| \geq 1, \forall a \in M$), depending on where the user is at the scheduled time. Reminders have a duration of $s_a = 2$ minutes $\forall a \in M$. A reminder must be delivered at least ω_{min} minutes, and no more than ω_{max} minutes, before the associated bingo game. This ensures that users receive their reminder within a reasonable proximity to the game, such that they do not forget, and with enough notice that they can finish what they are doing and attend the game.

6.2.5 Problem Input, Solution, and Objectives

As input, a problem instance specifies the environment, $\mathcal{L}$, the set of users, U, the social robot fleet, R, and the HRI activity set, A. The objective is then to create a set of social robot and user routes that allow all of the requested telepresence sessions to be facilitated, as well as schedule a set of bingo games with their associated reminders.

Definition 6.2.5. *A solution to an instance of SRRP-RH is a set of social robot and resident user routes bounded by planning horizon $|T|$ such that:*

- *Social robot routes begin and end at the robot depot. User routes begin and end in their respective personal rooms.*
- *Users, $u \in U$, participate in all commitments noted in their user calendars, $\{\lambda \in \Lambda^u \mid \mu(\lambda) = 0\}$, as well as any telepresence activities, they have requested, $\{a \in S \mid u \in \alpha(a)\}$.*
- *Each scheduled activity, $a \in A$, is facilitated by exactly one compatible social robot. Activity start and end times are synchronized in both user and robot routes. Activity location is the same in both user and robot routes.*
- *Each user, $u \in U$, that participates in a bingo game, $a \in G$, receives a reminder HRI for that game within $[\omega_{min}, \omega_{max}]$ minutes prior to the game.*
- *A hosted bingo game activity, $a \in G$, has a number of user participants within the range $[g_{min}, g_{max}]$. Each user, $u \in U$, participates in at least g^u_{min} and no more than g^u_{max} games.*
- *All social robot and user routes are temporally feasible.*
- *All social robot routes are energy feasible (i.e., robot of type $k \in K$ energy level is always bounded by $[0, Q^k]$).*
- *All recharge station visits are temporally sequenced to accommodate the unary capacity stations.*

In addition to finding feasible solutions to the problem, in this work we also explore a number of different objective functions, including:

1. Social robot fleet distance minimization.
2. Bingo game participation maximization.
3. The complex objective function proposed in previous work [199], a combination of activity participation, reminder proximity, and robot fleet energy consumption.

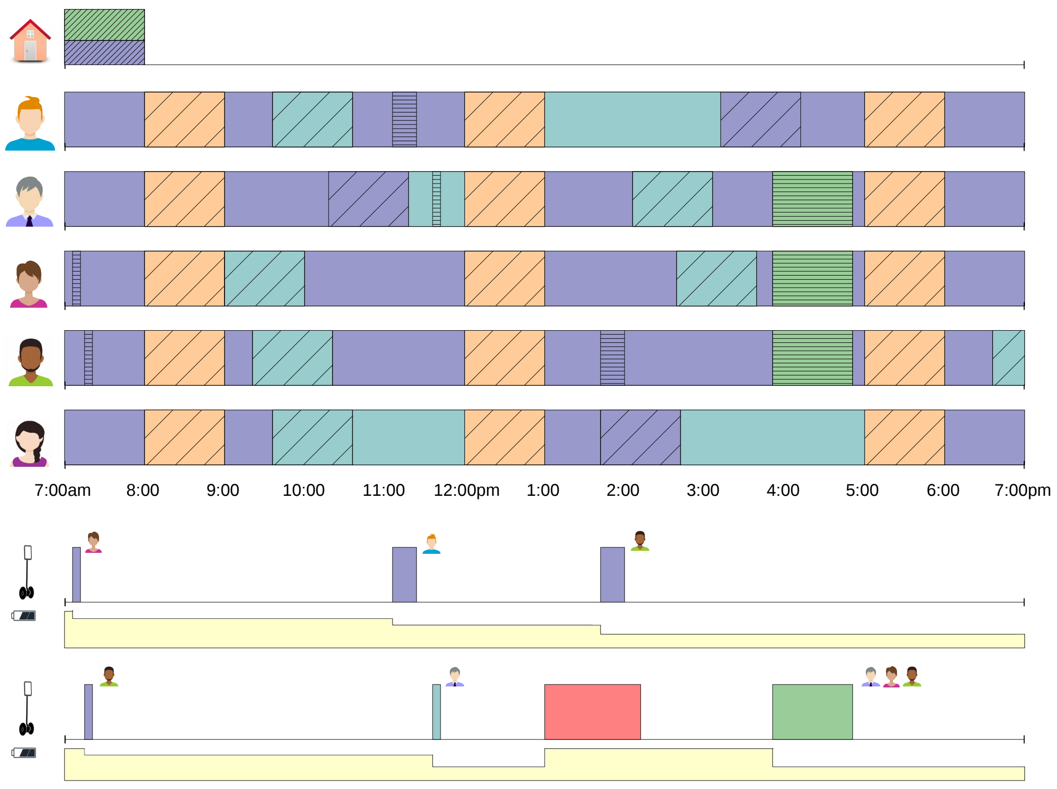


Figure 6.4: Solution for a retirement home MRTA problem instance involving five users, $|U| = 5$, and two homogeneous social robots, $|R| = 2$. Additionally, there are two mandatory telepresence sessions, $|S| = 2$, and one bingo game activity, $|G| = 1$.

While the first objective function is a traditional vehicle routing objective, the latter two represent more problem-driven objectives designed to not only more efficiently route the robot fleet, but also increase the number of HRI tasks facilitated throughout the day.

Example 6.2.1. *A feasible solution to a problem instance involving five users and two homogeneous robots is illustrated in Figure 6.4. The solution consists of two telepresence activities and one bingo game with three participants (with associated reminders). Facility-specific activity time windows for bingo games and telepresence sessions are specified at the top of the diagram. In this example, bingo games and telepresence tasks must occur sometime after 8:00 AM. User schedule blocks with horizontal patterns indicate an HRI activity with a robot. Colored blocks on robot schedules indicate facilitated activities with users in the location indicated by the color, synchronized with the user schedules. Robot energy levels are indicated throughout the day in yellow. Recharging tasks increase energy level and all other tasks decrease energy. We note that unless tasks occur in the same location (e.g., reminder and telepresence in a user personal room), the tasks are never scheduled immediately next to each other due to the need for both users and robots to move within the retirement home. Telepresence sessions are always facilitated by robots in the personal rooms of the associated users, and reminders are facilitated in the anticipated location of the user, provided they are available for interaction.*

6.2.6 Assumptions

In this chapter, our proposed approaches are developed under the following assumptions:

- Robots travel through the environment unhindered. More sophisticated motion planning (e.g., congestion avoidance) is left to future work. Additionally, travel distances, transition times, and travel-incurred energy expenditures are positive valued and satisfy the triangle inequality.
- All robot energy consumption and replenishment rates are linear.
- All robot types can use all recharging stations and recharge stations are unary capacity (i.e., one robot can recharge at a given time).
- All problem parameters are deterministic and the optimization is static.

Many of these assumptions are problematic in the context of the real-world implementation of our produced robot routes, but necessary to make progress on key challenges. The removal of these assumptions constitute strong candidates for future work as we discuss in Section 6.9.5.2.

6.2.7 Problem Classification

We classify the retirement home robot task planning problem according to taxonomies from both the VRP and MRTA literature bodies.

6.2.7.1 Vehicle Routing

Given that the fleet of social robots has a limited range, battery capacity, and need to consider recharging at spatially distributed recharge stations, the problem bears many similarities to the EVRPTW as covered extensively in Chapter 4. The problem also has a synchronization component; both users and robots need to be in a common location at the scheduled start time of a task. For this reason, the problem also contains elements from the vehicle routing problem with multiple synchronization constraints (VRPMS) [58]. VRPMS problems, traditionally, involve customer service requests that require more than one vehicle to serve them; the vehicles must occasionally be synchronized in time and space to transfer materials, for example. In our MRTA problem, the completion of an HRI activity requires robots and users to be synchronized in both time and space. Some activities require one-to-one synchronization between robots and users (i.e., telepresence sessions and reminders), while bingo games require one-to-many synchronization.

A characteristic of our problem that is, to the knowledge of the author not addressed in the VRP literature, is the routing of human users to improve solution viability. It is often the case that humans are involved in route planning as they are required to operate vehicles, however, it is less common for the humans to have their routes planned without vehicle accompaniment.[2]

6.2.7.2 Multi-Robot Task Allocation

According to the initial MRTA taxonomy [79], our problem is classified as ST-SR-TA. The tasks are single-robot tasks (ST), each robot is a single-task robot (i.e., unary capacity resource) (SR), and the resulting allocation is time-extended (TA) (i.e., a schedule). With respect to a more comprehensive

[2]We note that, for example, the Driverlog planning domain considers producing plans for both trucks and drivers [86].

and recent taxonomy [124], our problem is classified as MRTA with cross-schedule dependencies, XD [ST-SR-TA]. This classification pertains to MRTA problems where robots are unable to independently optimize their own schedules as the quality of their schedule depends on another robot. It does not have complex dependencies as there is only a single way to decompose the compound task (bingo game).

For our problem, since the bingo game task and reminder tasks can be assigned to different robots, there exist cross-schedule dependencies (i.e., precedence constraints across robots in the fleet). The existence of this relationship means that a single robot, having been assigned a bingo game activity, but not all of the associated reminders, cannot independently optimize its own schedule. This scenario is observed in Figure 6.4 where the second robot has been allocated the bingo game, however, the first robot has been assigned to deliver the required reminder to one of the users involved.

6.2.8 Complexity

While the SRRP-RH has been studied previously and approached with tree search-based techniques, its complexity has never been addressed.

Proposition 6.2.1. *The SRRP-RH with a robot fleet distance minimization objective function is $\mathcal{NP}$-Hard.*

Proof. The complexity of the SRRP-RH can be readily seen by demonstrating that the problem is a generalization of the vehicle routing problem with time windows (VRPTW), a known $\mathcal{NP}$-hard problem [139]. Given a particular instance of SRRP-RH with $g^u_{min} = g^u_{max} = 0, \forall u \in U$ (i.e., users do not wish to participate in bingo games) the only activities that need to be facilitated are required telepresence sessions, which take place in user personal rooms with facility-imposed time window $p \in P_a$. Additionally, if $\Lambda^u = \emptyset, \forall u \in U$, each user has no commitments throughout the day and can remain in their personal room. With $|K| = 1$, indicating a fleet of homogeneous social robots, and both $h^k \ll Q^k$ and $o^k_a \ll Q^k$, the energy consumption component of the problem becomes inconsequential. The resulting instance is a robot (vehicle) fleet distance minimization problem where the customers are the telepresence session HRIs in the personal rooms of the users. Each robot must start and end the day in the robot depot. This can be clearly mapped to an instance of VRPTW. $\square$

6.3 Related Work

The development of service robots is an active area of research with impressive forecasts for future adoption [7]. While mobile service robots are being used to assist humans with dull, repetitive tasks, such as household chores, recent trends indicate considerable interest in commercial applications for autonomous robots with the ability to interact and engage with humans [29]. This class of mobile service robots, known as social robots, must be capable of facilitating HRIs, requiring human-centric abilities such as facial and speech recognition, in addition to the other core functions (i.e., navigation, obstacle avoidance, etc.). Indeed, over the past decade, the annual RoboCup symposium (widely known for the RoboCup soccer competition) has hosted a RoboCup@Home competition that looks to test the abilities of proposed service robots that are socially relevant and capable of human-robot interaction [210, 192]. For example, in RoboCup@Home 2013, the winning team designed a service robot that could interact naturally with humans using speech, gestures, and body language in the context of a cocktail party [191].

Socially assistive robots, namely those that interact in some meaningful way with a human user, have seen growing attention in the literature, as demonstrated by various studies concerning the use of socially assistive robotics for human-robot interactions with stroke survivors, the elderly, and children with autism [61]. The CoBots initiative focused on the development and long-term testing (over three years) of mobile service robots that perform user-requested tasks within an office environment and ask users for help when necessary [206]. Similarly, the STRANDS project investigated long-term autonomy, developing and deploying mobile service robots in a number of human-populated environments, successfully performing tasks such as receptionist duties [95].

With respect to elderly care, socially assistive robots have been shown to be useful as an interface for the elderly to digital technology, and to increase the quality of elderly life through the provision of companionship [31]. Social robots have been shown to have positive effects on the human psychological state in a number of studies [117, 9]. Previous studies have even indicated that retirement home residents are more positive about mobile robots than the staff of the facility [30]. The work in this chapter is part of a larger project involving a long-term study on the deployment of intelligent human-like mobile robots in retirement homes. Previous work within this project has investigated mobile robotics to promote facilitation of stimulating recreational activities such as bingo [142, 143], search for users within a retirement home environment [185, 24], assist elderly residents with meal-time tasks [151], and recognize emotional body language of human residents [150]. Past work has also investigated the planning and scheduling of the mobile robots within the environment using a variety of techniques, including classical artificial intelligence (AI) planning and discrete optimization [205, 28, 199, 24].

6.4 Existing Formulations

Previous efforts have been made to model this MRTA problem using both MILP and CP [199, 200]. In contrast to modeling strategies in vehicle routing, wherein the problem is first represented as an augmented routing graph, these initial formulations represent activities directly. In this section we present and review these existing modeling efforts and discuss their limitations. We adjust the notation, where possible, to align with our problem definition parameters as illustrated in Table 6.1. We also make notation corrections and clarifications to the models, providing a brief description of the more major changes.

To facilitate modeling, previous work [200] extends the reminder activity set such that there is an optional activity for each candidate location of a reminder HRI. We denote this augmented reminder HRI set as $\hat{M}$ such that $|\hat{M}| = \sum_{a \in M} |\zeta(a)|$. This same previous work also includes robot recharges as activities to be scheduled. The set of recharge activities, C, is constructed such that $|C| = |F| \cdot n_f$. We let C^f denote the recharge tasks associated with station $f \in F$. To enable robot routing, dummy tasks representing instances of the robot depot are denoted $\dot{a}$ and $\ddot{a}$, for the start and end depot instances, respectively. We denote the set of all activities, as well as the depot instances, to be $\hat{A} = S \cup G \cup \hat{M} \cup C \cup \{\dot{a}, \ddot{a}\}$ and the set of non-recharge, non-depot activities to be $\tilde{A} = S \cup G \cup \hat{M}$. $\hat{M}_a \subset \hat{M}$ is defined as the set of reminder activities associated with bingo game activity $a \in G$, $\hat{M}_u$ the set of reminder activities associated with user $u \in U$, and $\hat{M}_{ua}$ the set of reminder activities associated with both activity a and user u. With a slight abuse of notation, $d_{aa'}$ represents the distance between the locations of activities a and a' from the augmented activity set.

Variable	Description
$w_a \in \{0,1\}$	1 if activity a is scheduled and 0 otherwise
$x_{ra} \in \{0,1\}$	1 if activity a is processed by robot $r \in R$ and 0 otherwise
$y_{at} \in \{0,1\}$	1 if activity a starts at time $t \in T$ and 0 otherwise
$z_{raa'} \in \{0,1\}$	1 if activity a starts directly before task a' on robot r and 0 otherwise
$\phi_{uaa'} \in \{0,1\}$	1 if task a is sequenced at some point before task a' for user u, 0 otherwise
$\xi_{ua} \geq 0$	Delivery time to user u for game $a \in G$, 0 if user does not play game
$B_a \geq 0$	Completion time of activity a
$E_a \geq 0$	Energy level of robot that processes activity a at completion of a
$D_a \geq 0$	Duration of activity a
$e_a \in \mathbb{R}$	Energy consumed by activity a (negative for recharge activities)

Table 6.2: Decision variables for time-indexed MILP model presented in Tran et al. [200].

6.4.1 Time-Indexed MILP Model

The existing MILP model, as presented in Tran et al. [200], uses a time-indexed modeling technique. The decision variables for this formulation are summarized in Table 6.2 and the model is given by Objective (6.1) and Constraints (6.2) through (6.38). The set $t \in T_a$ represents the time points during which activity a can feasibly start and T_{ua} the set of time points for which activity a can start for user u. The authors construct these sets by reasoning about activity duration, activity time windows, and user availability in conjunction with activity location.

$$\min \quad \sum_{a \in G} \left(1000 \cdot (|U| - \sum_{a' \in \hat{M}_a} w_{a'}) + 500 \cdot (1 - w_a) \right) + \sum_{u \in U} \sum_{a \in G} \xi_{ua}$$
$$+ \sum_{a \in \tilde{A}} e_a + \sum_{r \in R} \sum_{a,a' \in \hat{A}} z_{raa'} d_{aa'} h^{\psi(r)} \tag{6.1}$$

$$\text{s.t.} \quad w_a = 1 \qquad \forall a \in S \cup \{\dot{a}, \ddot{a}\} \tag{6.2}$$

$$\sum_{a' \in \hat{M}_{ua}} w_{a'} \leq w_a \qquad \forall u \in U, a \in G \tag{6.3}$$

$$\sum_{r \in R} x_{ra} = w_a \qquad \forall a \in \tilde{A} \cup C \tag{6.4}$$

$$\sum_{r \in R} \sum_{a \in C^f} x_{ra} \leq n_f \qquad \forall f \in F \tag{6.5}$$

$$\sum_{t \in T_a} y_{at} = w_a \qquad \forall a \in \tilde{A} \cup C \tag{6.6}$$

$$\sum_{a \in \tilde{A} \cup C \cup \dot{a}} z_{raa'} = x_{ra'} \qquad \forall a' \in \tilde{A} \cup C \cup \ddot{a}, r \in R \tag{6.7}$$

$$\sum_{a' \in \tilde{A} \cup C \cup \ddot{a}} z_{raa'} = x_{ra} \qquad \forall a \in \tilde{A} \cup C \cup \dot{a}, r \in R \tag{6.8}$$

$$x_{ra'} + x_{ra} \geq 2 \cdot z_{raa'} \qquad \forall a, a' \in \tilde{A} \cup C, r \in R \tag{6.9}$$

$$\sum_{r\in R}\sum_{a\in\tilde{A}\cup C\cup\dot{a}} z_{raa'} = w_a \qquad \forall a'\in\tilde{A}\cup C \tag{6.10}$$

$$x_{ra} = 1 \qquad \forall r\in R, a\in\{\dot{a},\ddot{a}\} \tag{6.11}$$

$$\sum_{a'\in\hat{M}_a} w_{a'} \le g_{max} \qquad \forall a\in G \tag{6.12}$$

$$\sum_{a'\in\hat{M}_a} w_{a'} \ge g_{min}\cdot w_a \qquad \forall a\in G \tag{6.13}$$

$$\sum_{t\in T_a} y_{at}\cdot t + D_a = B_a \qquad \forall a\in\tilde{A}\cup C \tag{6.14}$$

$$1-\sum_{a'\in\hat{M}_{ua}} w_{a'} \ge y_{at} \qquad \forall u\in U, a\in G, t\in T\setminus T_{ua} \tag{6.15}$$

$$B_{a'} \le B_a - D_a - \omega_{min} \qquad \forall a\in G, a'\in\hat{M}_a \tag{6.16}$$

$$B_{a'} \ge B_a - D_a - \omega_{max} - V\cdot(1-w_{a'}) \qquad \forall a\in G, a'\in\hat{M}_a \tag{6.17}$$

$$E_a + e_a \ge 0 \qquad \forall a\in C \tag{6.18}$$

$$e_a = -D_a\cdot g^{\psi(1)} \qquad \forall a\in C \tag{6.19}$$

$$B_{a'} \ge B_a + D_a - V(1-w_{a'}) \qquad \forall f\in F, a,a'\in C^f \mid a<a' \tag{6.20}$$

$$D_a = s_a \qquad \forall a\in\tilde{A} \tag{6.21}$$

$$B_{a'} \ge B_a + \sum_{r\in R} z_{raa'}\cdot\frac{d_{aa'}}{\nu^{\psi(r)}} + D_{a'} + V(\sum_{r\in R} z_{raa'} - 1) \qquad \forall a,a'\in\tilde{A}\cup C \tag{6.22}$$

$$E_{a'} \le E_a - \sum_{r\in R} z_{raa'}\cdot\frac{h^{\psi(r)}\cdot d_{aa'}}{\nu^{\psi(r)}} - e_{a'} + V(1-\sum_{r\in R} z_{raa'}) \qquad \forall a,a'\in\tilde{A}\cup C \tag{6.23}$$

$$\sum_{a\in\hat{M}_u} w_a \ge g^u_{min} \qquad \forall u\in U \tag{6.24}$$

$$\sum_{a\in\hat{M}_u} w_a \le g^u_{max} \qquad \forall u\in U \tag{6.25}$$

$$\xi_{ua} \ge B_a - B_{a'} - V(1-w_{a'}) \qquad \forall u\in U, a\in G, a'\in\hat{M}_{ua} \tag{6.26}$$

$$B_{a'} \ge B_a + D_{a'} + V[(\phi_{uaa'}-1)+(w_a+w_{a'}-2)] \qquad \forall u\in U, a,a'\in\tilde{A}\cup C_u \tag{6.27}$$

$$\phi_{uaa'} + \phi_{ua'a} = 1 \qquad \forall u\in U, a,a'\in\tilde{A}\cup C_u \tag{6.28}$$

$$w_a\in\{0,1\} \qquad \forall a\in\hat{A} \tag{6.29}$$

$$x_{ra}\in\{0,1\} \qquad \forall a\in\hat{A}, r\in R \tag{6.30}$$

$$y_{at}\in\{0,1\} \qquad \forall a\in\hat{A}, t\in T \tag{6.31}$$

$$z_{raa'}\in\{0,1\} \qquad \forall a,a'\in\hat{A}, a\ne a', r\in R \tag{6.32}$$

$$\phi_{uaa'}\in\{0,1\} \qquad \forall a,a'\in\hat{A}, a\ne a', u\in U \tag{6.33}$$

$$\xi_{ua}\ge 0 \qquad \forall a\in G, u\in U \tag{6.34}$$

$$B_a \ge 0 \qquad \forall a\in\hat{A} \tag{6.35}$$

$$E_a \ge 0 \qquad \forall a\in\hat{A} \tag{6.36}$$

$$D_a \ge 0 \qquad \forall a\in\hat{A} \tag{6.37}$$

$$e_a\in\mathbb{R} \qquad \forall a\in\hat{A} \tag{6.38}$$

Objective (6.1) minimizes a combination of: i) the number of bingo games that are not facilitated, ii) the number of residents that do not attend each game, iii) the temporal proximity of delivered reminders to the corresponding bingo game, and iv) the energy consumed by the robot fleet. Constraint (6.2) specifies that the telepresence tasks and start/end dummy tasks, $\dot{a}$ and $\ddot{a}$ respectively, are mandatory. Constraint (6.3) ensures that reminders are only facilitated when their corresponding bingo games are also facilitated. Constraint (6.4) ensures that each task, if present, is facilitated by exactly one social robot, while Constraint (6.5) ensures the number of recharges used at a given station does not exceed n_f.[3] Constraint (6.6) ensures that each task starts at exactly one time step. Constraints (6.7) and (6.8) specify that each facilitated task has a task immediately before it, and one immediately after (including the dummy tasks).[4] These effectively function as degree constraints in traditional routing models. Constraint (6.9) links the x_{ra} variables to the $z_{raa'}$ variables while Constraint (6.10) links $z_{raa'}$ to w_a. Constraint (6.11) specifies that the dummy tasks are present for each robot. Constraints (6.12) and (6.13) express the participation requirements on facilitated bingo games. Constraint (6.14) constrains the completion time variables, B_a, to be equal to the start time of a task plus task duration. Constraint (6.15) ensures that bingo game tasks are subject to the calendar availability of all of the resident participants. Constraints (6.16) and (6.17) specify the temporal proximity requirements of reminder tasks to their associated bingo games. Constraint (6.18) ensures that the robot has sufficient energy to travel to recharge tasks before recharging, and Constraint (6.19) defines the energy consumption of recharge tasks. Constraint (6.20) breaks symmetries on recharge tasks at each recharge station and Constraint (6.21) specifies the duration of non-recharge tasks to be their static parameter duration. Constraint (6.22) expresses the disjunctive temporal relationship for task completion time, ensuring tasks facilitated by a given robot do not overlap. Constraint (6.23) expresses a similar relationship, using disjunctive constants to capture robot energy levels. Constraint (6.24) and Constraint (6.25) limit bingo game participation to be within the bounds they have expressed. Constraint (6.26) stores the relative proximity of a reminder task to its corresponding bingo game into variable ξ_{au}. Constraint (6.27) enforces unary capacity on users (i.e., they can only participate in a single task at a time). Constraint (6.28) imposes an ordering on tasks for each user. Finally, Constraints (6.29) through (6.38) detail the domains of the decision variables.

6.4.1.1 Model Discussion

The presented MILP model has a number of limitations. First, the model utilizes a time-indexed modeling strategy, which is highly sensitive to the length of scheduling horizons and the selected scheduling granularity. If, for instance, the retirement home were to extend the scheduling horizon from 12 hours to 14 hours, the time indexed approach would likely suffer as 120 new time points are added to its representation, incurring significantly more binary y_{at} decision variables and supporting constraints. In constrast, shorter horizons will benefit from fewer variables and constraints.

Second, even though the formulation explicitly tracks each robot through the use of x_{ra} assignment and $z_{raa'}$ sequencing variables, it does not adequately model heterogeneous robot fleets. The definition of variable e_a represents the energy consumed (or replenished) by activity a. The consumption or replenishment of an activity is a function of the characteristics of the robot type facilitating it. Constraint (6.19) defines energy consumption of recharge tasks to be the duration of the task multiplied by the replenishment rate of a specific robot, precluding the proper modeling of heterogeneous fleets. A potential

[3]This constraint is added to facilitate comparison between models.

[4]Constraint (6.8) was included as an amendment to the original model in order to facilitate proper activity sequencing.

Variable	**Type**	**Description**
w_a	Interval	Representing real activity $a \in S \cup G \cup \hat{M}$
$\bar{w}_{ra}$	Interval	Clone task of activity $a \in \tilde{A} \cup C$ for robot $r \in R$
$\bar{w}_{ua}$	Interval	Clone task of activity $a \in G$ for user $u \in U$
$w_{\dot{a}}, w_{\ddot{a}}$	Interval	Start and end dummy interval variables
$participants_a$	Integer	Participants in game $a \in G$
$games\_attended_u$	Integer	Games attended by user $u \in U$
del_time_a	Integer	Delivery proximity of reminder $a \in \hat{M}$
e_cons_{ra}	Integer	Energy consump. of activity $a \in \tilde{A} \cup C$ by robot $r \in R$
rs_r	Sequence	Sequence variable for robot $r \in R$
rc_r	Cumul. function expr.	Resource variable for robot $r \in R$
uc_u	Cumul. function expr.	Resource variable for user $u \in U$
chc_f	Cumul. function expr.	Resource variable for recharge station $f \in F$
re_r	Cumul. function expr.	Energy level tracking for robot $r \in R$

Table 6.3: Decision variables for alternative resource CP model presented in Tran et al. [199].

fix could be to redefine the variable to be e_{ra}, the energy consumption of activity a when facilitated by robot $r \in R$, however this has implications on other areas of the model. We should note that, while the model appears to address heterogeneous fleets, the experiments conducted in Tran et al. [199] involve strictly homogeneous fleets, for which the formulation is sufficient.

6.4.2 Alternative Resource CP Model

The existing CP model, as presented in Tran et al. [199], is based on an alternative resource modeling technique. The formulation uses parameter $calendar_a$, which is similar to T_a from the MILP model, however, $calendar_a$ includes all time points that activity a cannot overlap with (as opposed to feasibly start during). First, an interval variable is generated for each activity $a \in S \cup G$ denoted w_a. Next, for each real activity, $a \in \tilde{A} \cup C$, a 'clone' interval variable is created for each robot, $r \in R$, representing a task-to-robot assignment, $\bar{w}_{ra}$. Task-to-user assignment interval variables are created for each bingo game $a \in G$, and each reminder associated with that bingo game, $a' \in \hat{M}_a$. These interval variables are denoted by $\bar{w}_{ua}$. The square matrix Δ_r represents the travel time between all pairs of task locations by a robot $r \in R$, and is defined as $\Delta_r = \{\frac{d_{aa'}}{\nu^{\psi(r)}} : a, a' \in \hat{A}\}$. The parameter $max\_dist_a = \max_{a' \in \hat{A}}(d_{aa'})$ represents the maximum distance from the location of activity a to the location of another activity in the environment. The complete set of decision variables for this formulation is summarized in Table 6.3 and the model is given by Objective (6.39) and Constraints (6.40) through (6.76).

$$\min \quad \sum_{a \in G} 500 \cdot (1 - \textsc{Pres}(w_a)) + 1000 \cdot (|U| - \sum_{u \in U} games\_attended_u) + \sum_{a \in \hat{M}} del\_time_a + \sum_{r \in R} \sum_{a \in \hat{A}} e\_cons_{ra} \tag{6.39}$$

$$\text{s.t.} \quad \textsc{Pres}(w_a) = 1 \qquad \forall a \in S \cup \{\dot{a}, \ddot{a}\} \tag{6.40}$$

$$\textsc{Pres}(w_{a'}) \leq \textsc{Pres}(w_a) \qquad \forall a \in G, u \in U, a' \in \hat{M}_{ua} \tag{6.41}$$

$$\textsc{Alternative}(w_a, \{\bar{w}_{ra} : r \in R\}) \qquad \forall a \in \tilde{A} \tag{6.42}$$

$$\textsc{ForbidExtent}(w_a, calendar_a) \qquad \forall a \in \tilde{A} \tag{6.43}$$

$$\textsc{ForbidExtent}(\bar{w}_{ua}, \Lambda^u) \qquad \forall a \in G, u \in U \tag{6.44}$$

$$\textsc{NoOverlap}(rs_r, \Delta_r) \qquad \forall r \in R \tag{6.45}$$

$$rs_r : \textsc{Sequence}(\{\bar{w}_{ra} : a \in \hat{A}\}) \qquad \forall r \in R \tag{6.46}$$

$$\sum_{a' \in \hat{M}_{ua}} \textsc{Pres}(w_{a'}) = \textsc{Pres}(\bar{w}_{ua}) \qquad \forall a \in G, u \in U \tag{6.47}$$

$$\sum_{u \in U} \sum_{a' \in \hat{M}_{ua}} \textsc{Pres}(w_{a'}) = participants_a \qquad \forall a \in G \tag{6.48}$$

$$g_{min} \cdot \textsc{Pres}(w_a) \leq participants_a \leq g_{max} \qquad \forall a \in G \tag{6.49}$$

$$\sum_{a \in G} \sum_{a' \in \hat{M}_{ua}} \textsc{Pres}(w_{a'}) = games\_attended_u \qquad \forall u \in U \tag{6.50}$$

$$g^u_{min} \leq games\_attended_u \leq g^u_{max} \qquad \forall u \in U \tag{6.51}$$

$$\textsc{Start}(w_{a'}) \leq \textsc{Start}(a_g) - \omega_{min} \qquad \forall a \in G, u \in U, a' \in \hat{M}_{ua} \tag{6.52}$$

$$\textsc{Start}(w_{a'}) \geq \textsc{Start}(a_g) - \omega_{max} \qquad \forall a \in G, u \in U, a' \in \hat{M}_{ua} \tag{6.53}$$

$$rc_r = \sum_{a \in \tilde{A} \cup C} \textsc{Pulse}(\bar{w}_{ra}, 1) \qquad \forall r \in R \tag{6.54}$$

$$rc_r \leq 1 \qquad \forall r \in R \tag{6.55}$$

$$\begin{aligned} uc_u = &\sum_{a \in \hat{M}_u} \textsc{Pulse}(w_a, 1) \\ &+ \sum_{a \in S | u \in \alpha(a)} \textsc{Pulse}(w_a, 1) + \sum_{a \in G} \textsc{Pulse}(\bar{w}_{ua}, 1) \end{aligned} \qquad \forall u \in U \tag{6.56}$$

$$uc_u \leq 1 \qquad \forall u \in U \tag{6.57}$$

$$chc_f = \sum_{r \in R, a \in C^f} \textsc{Pulse}(\bar{w}_{ra}, 1) \qquad \forall f \in F \tag{6.58}$$

$$chc_f \leq 1 \tag{6.59}$$

$$\begin{aligned} re_r = &\textsc{StepAtStart}(w_{\dot{a}}, Q^{\psi(r)}) \\ &+ \sum_{a \in \hat{A} \setminus \{\dot{a}\}} \textsc{StepAtStart}(\bar{w}_{ra}, -e\_cons_{ra}) \end{aligned} \qquad \forall r \in R \tag{6.60}$$

$$\begin{aligned} e\_cons_{ra} = &\textsc{Pres}(\bar{w}_{ra}) \cdot d_{\textsc{Prev}_{rs_r}(a),a} \cdot h^{\psi(r)} \\ &+ \textsc{Length}(\bar{w}_{ra}) \cdot o_a^{\psi(r)} \end{aligned} \qquad \forall a \in \hat{A} \setminus \{\dot{a}\}, r \in R \tag{6.61}$$

$$s_a o_a^{\psi(r)} \leq e\_cons_{ra} \leq s_a o_a^{\psi(r)} + max\_dist_a \cdot h^{\psi(r)} \qquad \forall a \in \tilde{A} \cup \{\ddot{a}\}, r \in R \tag{6.62}$$

$$-Q^{\psi(r)} \leq e\_cons_{ra} \leq max\_dist_a \cdot h^{\psi(r)} \qquad \forall a \in C, r \in R \tag{6.63}$$

$$0 \leq re_r \leq Q^{\psi(r)} \qquad \forall r \in R \tag{6.64}$$

$$del\_time_{a'} = \textsc{Pres}(w_{a'}) \cdot (\textsc{Start}(w_a) - \textsc{Start}(w_{a'})) \qquad \forall a \in G, a' \in \hat{M}_a \tag{6.65}$$

$$\omega_{min} \leq del\_time_{a'} \leq \omega_{max} \qquad \forall a \in G, a' \in \hat{M}_a \tag{6.66}$$

$$\textsc{Pres}(\bar{w}_{ra'}) \leq \textsc{Pres}(\bar{w}_{ra}) \qquad \forall a, a' \in C^f | a < a', r \in R \tag{6.67}$$

$$\sum_{r \in R} \sum_{a \in C^f} \textsc{Pres}(\bar{w}_{ra}) \leq n_f \qquad \forall f \in F \tag{6.68}$$

$$\textsc{Start}(w_a) = \textsc{Start}(\bar{w}_{ua}) \qquad \forall a \in G, u \in U \tag{6.69}$$

$$\textsc{Length}(w_a) = \textsc{Length}(\bar{w}_{ua}) \qquad \forall a \in G, u \in U \tag{6.70}$$

$$\textsc{Length}(w_a) = s_a \qquad \forall a \in \tilde{A} \tag{6.71}$$

$$\textsc{Pres}(w_a) \in \{0, 1\} \qquad \forall a \in \tilde{A} \tag{6.72}$$

$$\textsc{Pres}(\bar{w}_{ra}) \in \{0, 1\} \qquad \forall a \in C, r \in R \tag{6.73}$$

$$0 \leq \textsc{Length}(\bar{w}_{ra}) \leq \frac{Q^{\nu(r)}}{g^{\nu(r)}} \qquad \forall a \in C, r \in R \tag{6.74}$$

$$\textsc{Length}(w_{\dot{a}}) = \textsc{Length}(w_{\ddot{a}}) = 0 \tag{6.75}$$

$$\textsc{Start}(w_{\dot{a}}) = 0,\ \textsc{Start}(w_{\ddot{a}}) = H \tag{6.76}$$

Constraint (6.40) establishes telepresence sessions and dummy activities as mandatory. Constraint (6.41) bounds a user's participation in a reminder activity by the presence of the associated bingo game activity. Constraint (6.42) uses the Alternative global constraint to ensure that whenever an HRI activity variable w_a is present, it starts with exactly one robot facilitation variable. Constraint (6.43) and (6.44) use the ForbidExtent global constraint to ensure HRI activities do not interfere with their time windows, nor user schedules where appropriate. Constraint (6.45) uses the NoOverlap global constraint to enforce temporal sequencing, including travel times, on the tasks a robot can potentially be involved in, defined by the sequence variable in Constraint (6.46). Constraint (6.47) links user reminder participation to user bingo participation. Constraint (6.48) links user reminders to the number of participants in a bingo game, a variable with bounds defined in Constraint (6.49). Constraints (6.50) and (6.51) define and constrain the number of games attended by a given user. Constraints (6.52) and (6.53) ensure reminder activities are delivered within the required proximity of their corresponding bingo game. Constraints (6.54) and (6.55) ensure each robot only processes one task at a time using the Pulse constraint. Constraints (6.56) through (6.59) express similar unary capacity requirements on each user and recharge station. Constraints (6.60) through (6.64) express the impacts on robot energy cumulative function re_r throughout the day. Constraint (6.65) defines the reminder delivery time proximity variable and (6.66) provides its domain. Constraint (6.67) applies symmetry breaking to the use of recharge activities, and Constraint (6.68) ensures no more than n_f recharge tasks are consumed for each station. The remainder of the constraints, Eqns. (6.69) through (6.76), identify the domains of the other decision variables.

6.4.2.1 Model Discussion

Overall, this previously proposed CP formulation sufficiently models the problem, albeit the use of various clone tasks can be difficult to follow.[5] The Pulse constraint defined by Eqn. (6.54) for robot task capacity is actually redundant as the NoOverlap in Constraint (6.45) provides even stronger temporal reasoning by considering robot travel times. Furthermore the original description of the problem omitted Constraint (6.45) which is required to ensure user bingo participation does not interfere with their calendars, Λ^u.

The energy modeling in Constraints (6.60) through (6.62) poses an issue when coupled with the solver used for the experiments (CP Optimizer). Energy modeling using interval and cumulative function

[5]The use of an augmented graph, over which the model is designed, improves the clarity of presentation as detailed in Section 6.8.

expression variables with partial recharges requires a step impact to be applied at the start of the recharge activity *and* at the end. Applying only a start impact precludes the optimization from finding solutions where the energy expended from traveling to the recharge station exceeds the energy replenished at the station. To remedy this, the set of robot energy constraints is modified to the following:

$$\begin{aligned}
re_r = {} & \textsc{StepAtStart}(w_{\dot{a}}, Q^{\psi(r)}) \\
& + \sum_{a \in \hat{A} \setminus \{\dot{a}\}} \textsc{StepAtStart}(\bar{w}_{ra}, -e\_cons_{ra}) \\
& + \sum_{a \in C} \textsc{StepAtEnd}(\bar{w}_{ra}, e\_cons^{end}_{ra}) && \forall r \in R && (6.77)
\end{aligned}$$

$$\begin{aligned}
e\_cons_{ra} = {} & \textsc{Pres}(\bar{w}_{ra}) \cdot d_{\textsc{Prev}_{rs_r}(a),a} \cdot h^{\psi(r)} \\
& + \textsc{Length}(\bar{w}_{ra}) \cdot o^{\psi(r)}_a && \forall a \in \tilde{A} \cup \{\ddot{a}\}, r \in R && (6.78)
\end{aligned}$$

$$e\_cons_{ra} = \textsc{Pres}(\bar{w}_{ra}) \cdot d_{\textsc{Prev}_{rs_r}(a),a} \cdot h^{\psi(r)} \qquad \forall a \in C, r \in R \qquad (6.79)$$

$$e\_cons^{end}_{ra} = \textsc{Length}(\bar{w}_{ra}) \cdot g^{\psi(r)} \qquad \forall a \in C, r \in R \qquad (6.80)$$

$$s_a o^{\psi(r)}_a \le e\_cons_{ra} \le s_a o^{\psi(r)}_a + max\_dist_a \cdot h^{\psi(r)} \qquad \forall a \in \tilde{A} \cup \{\ddot{a}\}, r \in R \qquad (6.81)$$

$$0 \le e\_cons_{ra} \le max\_dist_a \cdot h^{\psi(r)} \qquad \forall a \in C, r \in R \qquad (6.82)$$

$$0 \le e\_cons^{end}_{ra} \le Q^{\psi(r)} \qquad \forall a \in C, r \in R \qquad (6.83)$$

$$0 \le re_r \le Q^{\psi(r)} \qquad \forall r \in R \qquad (6.84)$$

Revised Constraint (6.77) represents the new robot energy tracking expression. Recharge station activities, $a \in C$, now have both StepAtStart and StepAtEnd impacts, constrained to their required values in Constraints (6.79) and (6.80) and with domains expressed by Constraints (6.82) and (6.83), respectively. Constraints (6.78) and (6.81) represent the energy consumption of non-recharge tasks, while Constraint (6.84) provides bounds for robot energy cumulative function expression, re_r, throughout the day.

6.4.3 User Transition Modeling

The primary limitation in both of these recently proposed models is their lack of user transition modeling. While the social robot fleet is constrained such that facilitated tasks must be temporally feasible including robot travel times (accomplished via Constraint (6.22) in the MILP model and Constraint (6.45) in the CP model), such reasoning is not enforced for retirement home residents; unary capacity constraints are enforced to ensure users participate in one activity at a time via Constraint (6.27) in the MILP model and Constraint (6.57) in the CP model, however transition times are ignored. As a result, solutions to the existing models often have user calendar commitments (i.e., appointments) and HRI activities scheduled immediately next to each other. Such solutions are unable to be implemented in a real-world setting and erode the chance of carefully optimized schedules from being successful during execution. While extending user appointment duration to include a 'buffer' at the beginning and end of the appointments (and thus enable the user to move from one location to another) is a potential solution, it is crude and could potentially remove high-quality solutions depending on the layout of the retirement home facility. A more elegant solution to capture this problem characteristic is to produce solutions that account for a more accurate estimate of user transitions in the environment. To accomplish this, the proposed models also track retirement home resident movement throughout the facility, effectively producing routes for these residents in addition to those produced for the social robot fleet. This modeling strategy adds $|U|$ additional heterogeneous "vehicles" (human users) to the routing problem.

6.5 Developing Routing Models

As previously noted, the studied problem is complex with numerous side constraints [199]. Thus, we adapt the electric routing modeling strategies presented in Chapter 4 with the goal of designing models in the formalisms of MILP and CP that leverage advances made within the vehicle routing literature.

Our development of routing models for SRRP-RH follows these steps:

- Represent the SRRP-RH on a graph, following a vertex augmentation scheme similar to the one presented in Chapter 4. It is important to note that the recharging path multigraph technique cannot be straightforwardly used for the SRRP-RH as recharging stations in this problem are *unary capacity*, whereas the multigraph approach assumes stations are uncapacitated.
- Leverage social robot symmetry through the use of compact MILP modeling and symmetry breaking in CP. Users are modeled as heterogeneous agents and have their routes explicitly tracked in both formulations.
- Capture energy expenditure and replenishment while properly accounting for partial recharges.
- Include applicable side constraints.

The remainder of this chapter details our modeling efforts and an empirical assessment of our approaches compared to previous approaches [199].

6.6 Graph Representation

The augmented graph modeling approach represents our problem on a graph, $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \{v_0, v_1, \ldots, v_N, v_{N+1}\}$ is the total set of $N + 2$ vertices (N non-depot vertices and two depot vertices) and $\mathcal{A}$ the set of directed arcs connecting vertices. Whereas in traditional vehicle routing the vertices represent the locations of depots and customer requests, the vertices in the SRRP-RH represent the locations of the robot depot, recharge stations, user calendar commitments, and HRI activities. Start and end instances of the robot depot are represented by vertices in the graph, denoted v_0 (start) and v_{N+1} (end), respectively. Following previously introduced notation (see Section 6.2), the set of physical recharge stations is represented by F, the set of all HRI activities is represented by A, and the set of human user commitments defined by their calendars is represented by $\Lambda^u, \forall u \in U$. All new notation used in the augmented graph representation is summarized in Table 6.4.

Example 6.6.1. *An augmented graph representation for a problem instance involving five users, two homogeneous robots, two telepresence HRIs, and one bingo game HRI is illustrated in Figure 6.5. Start and end dummy vertices are generated at the robot depot (red). Two vertices, representing recharge station visits at the depot, are denoted in yellow. Start and end vertices are generated for each user personal room to facilitate the user starting and ending in their personal room. An extra vertex is generated in the personal rooms of the users involved in a telepresence activity (two of the users). Three*

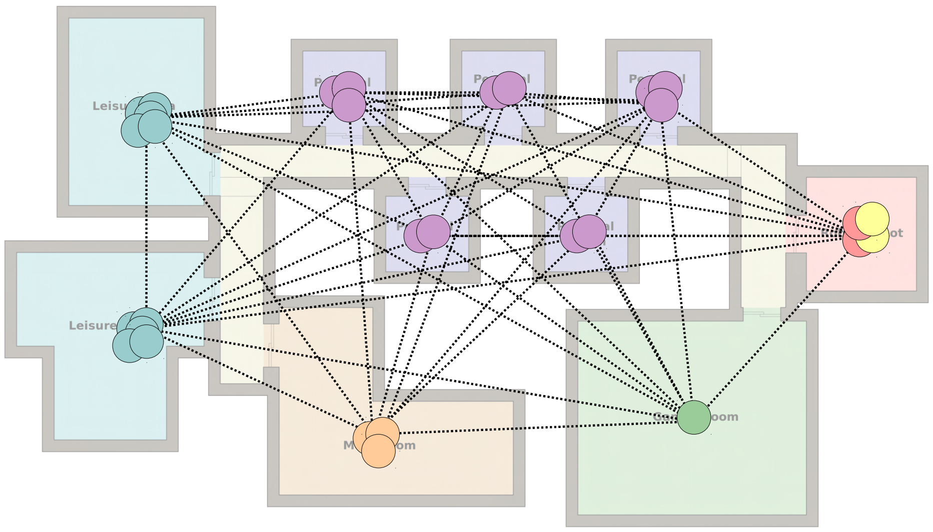


Figure 6.5: Augmented graph, $\mathcal{G}$, for instance with five users, two homogeneous robots, one bingo game HRI, and two telepresence sessions. The underlying facility is the same as the one presented in Figure 6.1. The single recharge station at the robot depot can be visited twice ($n_f = 2$), thus $|\bar{F}| = 2$. Arc weights represent travel time between pairs of vertices and can be set to accommodate the underlying topology.

vertices are generated for the meal room, each associated with one of the mandatory meal times (in this example, these are the only commitments in the user calendars). Since each user arrives and departs meal times at the same time, and they take place in a common location, it suffices that a single vertex is added for each meal time. A single vertex is generated in the games room to represent the single bingo game. Five vertices are generated in each of the leisure areas to allow for potential bingo reminders for each of the users in these locations.

6.6.1 Graph Vertices

The set of vertices, $\bar{F}$, augments the recharging station set, F, to allow multiple visits to each recharging station across the robot fleet. Following previous work [183], each recharge station is duplicated in the graph n_f times such that $|\bar{F}| = n_f \cdot |F|$. Thus, the vertices in the augmented recharge station set are given by $\bar{F} = \{v_1, v_2, \ldots, v_{n_f \cdot |F|}\}$. The parameter n_f must be carefully selected so as to avoid inhibiting the performance of the approach and maintain model completeness; we discuss our selection for this value in Section 6.9.2. The physical recharge station associated with a given vertex $i \in \bar{F}$ is given by F^i. The set of vertices associated with a physical recharge station, $f \in F$, is given by $\bar{F}^f$.

The set of graph vertices $\bar{A}$ augments the HRI activity set, A, to account for the candidate set of locations for each HRI activity. An auxiliary vertex is introduced for each candidate location of each HRI activity, $a \in A$, and added to the graph. The set of augmented vertices corresponding to an HRI activity, $a \in A$, is given by $\bar{A}^a$. The HRI activity associated with a vertex $i \in \bar{A}$ is given by A^i. Since bingo game and telepresence session HRIs each have a single candidate location, only one vertex is added to the graph for these activities. Bingo game reminders have a set of candidate locations (i.e., leisure areas or the user personal rooms) and thus a set of vertices, with cardinality equal to the number of candidate locations, is added to the graph for these activities.

Parameter	Description
v_0, v_{N+1}	Start and end depot vertices, respectively
$i \in \bar{F} = \{v_1, \ldots, v_{n_f \cdot \lvert F \rvert}\}$	Set of augmented vertices for recharge stations
$i \in \bar{A} = \{v_{\lvert \bar{F} \rvert+1}, \ldots, v_{\lvert \bar{F} \rvert + \lvert \bar{A} \rvert}\}$	Set of augmented vertices for HRI activities
$i \in \bar{A}^a \subseteq \bar{A}$	Task vertices associated with HRI a
$i \in \bar{A}^{a,\ell} \subseteq \bar{A}^a$	Task vertex associated with HRI a and candidate location ℓ
$i \in \bar{A}^k \subseteq \bar{A}$	Task vertices that robot type k can potentially visit
$i \in \bar{A}^u \subseteq \bar{A}$	Task vertices that user u can potentially visit
$i \in \bar{\Lambda} = \{v_{N-\lvert \bar{\Lambda} \rvert+1}, \ldots, v_N\}$	Set of vertices representing user commitments
$i \in \bar{\Lambda}^u \subseteq \bar{\Lambda}$	Set of vertices for user u commitments
$i \in \mathcal{V} = \{\bar{F} \cup \bar{A} \cup \bar{\Lambda}\}$	Total set of N non-depot vertices in the graph
$\mathcal{V}_0 = \mathcal{V} \cup \{v_0\}$, $\mathcal{V}_{N+1} = \mathcal{V} \cup \{v_{N+1}\}$	Vertex set containing the start and end depot instance, resp.
$\mathcal{V}_{0,N+1} = \mathcal{V} \cup \{v_0, v_{N+1}\}$	Vertex set containing all vertices (including depots)
$i \in \mathcal{V}^k \subseteq \mathcal{V}$	Vertices that robot type k can potentially visit
$i \in \mathcal{V}^u \subseteq \mathcal{V}$	Vertices that user u can potentially visit
$\mathcal{A} = \{(i,j) \mid i,j \in \mathcal{V}_{0,N+1}, i \neq j\}$	Total set of $(N+2)^2 - (N+2)$ arcs in the graph
$\mathcal{A}^k = \{(i,j) \mid i,j \in \mathcal{V}^k_{0,N+1}, i \neq j\}$	Arc set relevant to robot type $k \in K$
$\mathcal{A}^u = \{(i,j) \mid i,j \in \mathcal{V}^u, i \neq j\}$	Arc set relevant to user $u \in U$
s_i	Duration of visit to non-recharge vertex $i \in \mathcal{V} \setminus \bar{F}$
d_{ij}	Distance between vertices $i \neq j,$ where $d_{ii} = 0$
δ^k_{ij}	Travel time between vertices $i \neq j$ for robot type $k \in K$
δ^u_{ij}	Estimated user travel time between vertices $i \neq j$ for user $u \in U$

Table 6.4: Augmented graph representation notation.

Lastly, the vertex set $\bar{\Lambda}$ represents all of the user calendar intervals for which a user is unavailable for interaction, $\{\lambda \in \Lambda \mid \mu(\lambda) = 0\}$. This set does not need to be augmented as each vertex is unique to a user. A vertex is included for each such commitment where the location of the vertex is given by ℓ_λ. These vertices are given by $\bar{\Lambda} = \{v_{N-\lvert \bar{\Lambda} \rvert+1}, \ldots, v_N\}$. The set of calendar commitment vertices for a particular user, $u \in U$, is given by $\bar{\Lambda}^u \subseteq \bar{\Lambda}$. The first element in a user's calendar specifies their starting location for the planning period and the last element specifies their ending location, both in the personal room of the user.

The total set of non-depot vertices in the graph, $\mathcal{V}$, is given by the union of each of these sets. Specifically, $\mathcal{V} = \bar{F} \cup \bar{A} \cup \bar{\Lambda}$. Notation $\mathcal{V}_0$, $\mathcal{V}_{N+1}$, and $\mathcal{V}_{0,N+1}$, is used to define vertex sets that include the first, last, and both depot instances, respectively. We denote $\mathcal{V}^k$ to be the set of vertices relevant to a robot of type $k \in K$, such that $\mathcal{V}^k = \bar{F} \cup \bar{A}^k$. We also denote $\mathcal{V}^u$ as the set of vertices relevant to a user $u \in U$, given by $\mathcal{V}^u = \bar{A}^u \cup \bar{\Lambda}^u$.

6.6.2 Graph Arcs

The arc set, $\mathcal{A}$, is defined between pairs of vertices, specifically $\mathcal{A} = \{(i,j) \mid i,j \in \mathcal{V}_{0,N+1}, i \neq j\}$. Given that there are many arcs in $\mathcal{G}$ that may be irrelevant to a given user or robot, we also define relevant arc sets for each. We denote $\mathcal{A}^k = \{(i,j) \mid i,j \in \mathcal{V}^k_{0,N+1}, i \neq j\}$ as the set of arcs relevant to robot type $k \in K$. We also define $\mathcal{A}^u = \{(i,j) \mid i,j \in \mathcal{V}^u, i \neq j\}$ as the set of arcs relevant to user $u \in U$. The travel time for a robot or user across a given arc (i,j) relevant to them is then given by δ^k_{ij} and δ^u_{ij}, respectively.

Variable	Type	Description
x_{ij}^k	Binary	1 if a robot of type k traverses directed arc (i, j), 0 otherwise
e_i^k	Continuous	Remaining energy of robot type k upon *arriving* at vertex i
E_i^k	Continuous	Remaining energy of robot type k upon *departing* vertex i
y_{ij}^u	Binary	1 if user u traverses directed arc (i, j), 0 otherwise
τ_i	Continuous	Arrival time at vertex i (for both robots and users)
η_{ap}	Binary	1 if HRI activity $a \in S \cup G$ held in time window $p \in P_a$, 0 otherwise
$\epsilon_{a,\ell,\lambda}$	Binary	1 if reminder $a \in M$ held in location ℓ, interval λ, 0 otherwise

Table 6.5: Decision variables for augmented graph MILP formulation

6.7 Augmented Graph MILP Model

In this section we detail the proposed MILP model for the SRRP-RH. The formulation is based on the augmented graph formulation for electric vehicle routing problems [183] using modeling techniques for fleets with heterogeneous vehicles [99].

6.7.1 Decision Variables

Table 6.5 summarizes the decision variables used in the augmented graph MILP model. While Chapter 4 dealt with homogeneous vehicles, this chapter involves a set of heterogeneous social robots and users. In this problem, each user is unique but some symmetry exists in the robot fleet according to robot type. The primary decision variable for this model is x_{ij}^k, indicating whether a robot of type k traverses directed arc $(i, j) \in \mathcal{A}^k$ or not. This variable leverages robot symmetry and does not track each robot (vehicle) explicitly, as the previously presented time-indexed MILP model does, but tracks each robot type explicitly to facilitate the required graph labeling. Decision variable y_{ij}^u indicates whether user u traverses directed arc $(i, j) \in \mathcal{A}^u$ or not. Users are explicitly tracked due to their heterogeneity (unique calendars, bingo preferences, etc.). Variable τ_i represents the arrival time, of both robots and users, at vertex i, and variable e_i^k represents the remaining energy of a robot of type k upon arrival at vertex i. Variable E_i^k represents the remaining energy of a robot upon departing recharge vertex i. The difference between e_i^k and E_i^k can be used to determine the time spent recharging at vertex i. Binary variable η_{ap} is introduced to indicate whether HRI activity $a \in A$ is held in time window $p \in P_a$ or not. Variable $\epsilon_{a,\ell,}$ identifies the time interval, λ, and location ℓ for reminder HRIs. The augmented graph MILP formulation is summarized by Eqn. (6.85) through (6.134).

6.7.2 Objective Function

In our models, we use the same complex objective function expressed in the previously proposed models. The objective function for our MILP formulation is given by Eqn. (6.85).

$$\min \sum_{a \in G, i \in \bar{A}^a} \left(1000 \cdot \left(|U| - \sum_{u \in U} \sum_{j \in \mathcal{V}^u \setminus \bar{A}^a} y_{ij}^u \right) + 500 \cdot \left(1 - \sum_{k \in \gamma(a)} \sum_{j \in \mathcal{V}_{N+1}^k \setminus \bar{A}^a} x_{ij}^k \right) \right)$$
$$+ \sum_{a \in A, i \in \bar{A}^a} \sum_{k \in \gamma(a)} \sum_{j \in \mathcal{V}_{N+1}^k, \setminus \bar{A}^a} s_a o_a^k x_{ij}^k + \sum_{k \in K} \sum_{i \in V_0^k, j \in \mathcal{V}_{N+1}^k, i \neq j} d_{ij} h^k x_{ij}^k$$

$$+\sum_{a\in G}\sum_{u\in U}\xi_{ua} \tag{6.85}$$

s.t. Constraints (6.86) through (6.134)

The first line of the objective function represents the cost associated with bingo game facilitation and participation. The second line represents the cost associated with the total energy expended by the social robot fleet (both HRI activity consumption and travel consumption). The final line represents the cost associated with bingo game reminder proximity; the closer reminders are delivered to their respective bingo games, the better. To express this portion of the objective, continuous variable ξ_{ua} is introduced and bounded by:

$$\Big(\tau_i - (\tau_j + s_{a'}) - |T|\cdot(1-\sum_{k\in\mathcal{V}^u} y^u_{jk})\Big) \le \xi_{ua} \le |T|, \quad \forall a\in G, i\in \bar{A}^a, u\in U, a'\in M_{ua}, j\in \bar{A}^{a'} \tag{6.86}$$

$$\xi_{ua} \ge 0 \qquad \forall a\in G, u\in U \tag{6.87}$$

The variable ξ_{ua} functions very similarly to its counterpart in the time-indexed MILP model previously proposed. Eqns. (6.86) and (6.87) essentially ensure that if user $u\in U$ participates in reminder a' for game a, the difference between the start of the bingo game and end of the associated reminders is added to the objective function. In the event that the user does not participate in the reminder, the value ξ_{ua} is bounded below by 0.

6.7.3 Constraints

We summarize the MILP model constraints in Eqns. (6.88) through (6.134). These constraints identify the key characteristics of the problem, including the flow through the graph, temporal relationships, energy consumption/replenishment, side constraints, and symmetry breaking constraints.

6.7.3.1 General Flow Constraints

The first family of constraints, the general flow constraints, dictate the permissible arc flow through the graph representation, including recharge tasks and user calendar commitments. Constraint (6.88) limits the number of outgoing arcs from the start depot for a robot type k to be limited by the number of robots of type k in the fleet (subsequently ensuring that the number of robots used is limited by the number of robots in the fleet). In the case of unlimited robots, this constraint would be relaxed. Constraint (6.89) places a lower bound on the number of robots routed. Constraint (6.90) ensures each augmented recharge vertex is visited at most once across all robot types. Constraint (6.91) ensures each HRI vertex is visited at most once by a robot. Constraint (6.92) ensures flow is preserved (i.e., in-flow equal to out-flow) for all vertices visited by a robot. Constraint (6.93) ensures each user visits all of their mandatory calendar commitments. Constraint (6.94) dictates that each HRI vertex can be visited at most once by a user. Finally, Constraint (6.95) ensures that the flow of each vertex in the graph is preserved for user routes.

$$\sum_{j\in\mathcal{V}^k} x^k_{0j} \le \Upsilon(k) \qquad \forall k\in K \tag{6.88}$$

$$\sum_{k\in K}\sum_{j\in\mathcal{V}^k} x^k_{0j} \ge 1 \tag{6.89}$$

$$\sum_{k\in K}\sum_{j\in\mathcal{V}^k_{N+1},i\neq j} x^k_{ij} \leq 1 \qquad \forall i\in\bar{F} \tag{6.90}$$

$$\sum_{j\in\mathcal{V}^k_{N+1},i\neq j} x^k_{ij} \leq 1 \qquad \forall i\in\bar{A}^k, k\in K \tag{6.91}$$

$$\sum_{i\in\mathcal{V}^k_{N+1},i\neq j} x^k_{ji} - \sum_{i\in\mathcal{V}^k_{0},i\neq j} x^k_{ij} = 0 \qquad \forall j\in\mathcal{V}^k, k\in K \tag{6.92}$$

$$\sum_{j\in\mathcal{V}^u_{N+1},i\neq j} y^u_{ij} = 1 \qquad \forall i\in\bar{\Lambda}^u, u\in U \tag{6.93}$$

$$\sum_{j\in\mathcal{V}^u_{N+1},i\neq j} y^u_{ij} \leq 1 \qquad \forall i\in\bar{A}^u, u\in U \tag{6.94}$$

$$\sum_{i\in\mathcal{V}^u_{N+1},i\neq j} y^u_{ji} - \sum_{i\in\mathcal{V}^u_{0},i\neq j} y^u_{ij} = 0 \qquad \forall j\in\mathcal{V}^u, u\in U \tag{6.95}$$

6.7.3.2 HRI Activity Constraints

The family of HRI activity constraints ensure that the flow routed through HRI activity vertices corresponds to their facilitation and participation by robots and users, respectively. Constraint (6.96) ensures that a telepresence activity is facilitated by exactly one social robot. Constraint (6.97) dictates that a user involved in a telepresence activity is present for that activity. Constraint (6.98) allows up to a maximum of one social robot to facilitate an optional activity, while Constraint (6.99) bounds user participation in optional activities by whether or not a robot in the fleet facilitates them. Finally, Constraint (6.100) ensures that, for HRIs with multiple candidate locations (i.e., reminder tasks), the flow between participating users and facilitating robots is synchronized to the same location. Constraint (6.101) ensures that if a user participates in a bingo game activity, $a \in G$, they receive the associated reminder HRI, $a' \in M_{ua}$.

$$\sum_{k\in\gamma(a)}\sum_{i\in\bar{A}^a,j\in\mathcal{V}^k_{N+1}\setminus\bar{A}^a} x^k_{ij} = 1 \qquad \forall a\in S \tag{6.96}$$

$$\sum_{i\in\bar{A}^a,j\in\mathcal{V}^u_{N+1}\setminus\bar{A}^a} y^u_{ij} = 1 \qquad \forall u\in\alpha(a), a\in S \tag{6.97}$$

$$\sum_{k\in\gamma(a)}\sum_{i\in\bar{A}^a,j\in\mathcal{V}^k_{N+1}\setminus\bar{A}^a} x^k_{ij} \leq 1 \qquad \forall a\in G\cup M \tag{6.98}$$

$$\sum_{i\in\bar{A}^a,j\in\mathcal{V}^u_{N+1}\setminus\bar{A}^a} y^u_{ij} \leq \sum_{k\in\gamma(a)}\sum_{i\in\bar{A}^a,j\in\mathcal{V}^k_{N+1}\setminus\bar{A}^a} x^k_{ij} \qquad \forall u\in\alpha(a), a\in G\cup M \tag{6.99}$$

$$\sum_{j\in\mathcal{V}^u_{N+1}\setminus\bar{T}^t} y^u_{ij} \leq \sum_{k\in\gamma(a)}\sum_{j\in\mathcal{V}^k_{N+1}\setminus\bar{A}^a} x^k_{ij} \qquad \forall i\in\bar{A}^{a,\ell}, a\in M, \ell\in\zeta(a), u\in\alpha(a) \tag{6.100}$$

$$\sum_{i\in\bar{A}^a,j\in\mathcal{V}^u_{N+1}\setminus\bar{A}^a} y^u_{ij} = \sum_{i\in\bar{A}^{a'},j\in\mathcal{V}^u_{N+1}\setminus\bar{A}^{a'}} y^u_{ij} \qquad \forall a\in G, a'\in M_{ua}, u\in\alpha(a') \tag{6.101}$$

6.7.3.3 Temporal Constraints

The family of temporal constraints ensures that the arrival times assigned to each vertex are feasible with respect to time. Constraint (6.102) dictates robot arrival time requirements for nodes when the preceding visit is of static duration (i.e., HRI activities), while Constraint (6.103) enforces the arrival time requirement of robots for vertices where the previous visit is of variable duration (i.e., recharge station visits). Constraint (6.104) dictates the arrival time requirements for vertices associated with human user visits. Constraint (6.105) ensures that all of the vertices pertaining to a given HRI activity are synchronized. Constraint (6.106) ensures that vertices for a given recharge station are visited in sequence (i.e., the stations are unary capacity). Constraints (6.107) and (6.108) enforce the arrival times at the depot vertices. Using a slight abuse of notation, we let s_i represent the duration of a visit to non-recharge vertex i (whether it be the duration of an HRI activity, $a \in A$, or user calendar interval, $\lambda \in \Lambda$).

$$\tau_i + (\delta^k_{ij} + s_i)x^k_{ij} - H(1 - x^k_{ij}) \leq \tau_j \qquad \forall i \in \mathcal{V}^k_0 \setminus \bar{F}, j \in \mathcal{V}^k_{N+1}, i \neq j, k \in K \tag{6.102}$$

$$\tau_i + \delta_{ij}x^k_{ij} + \frac{(E^k_i - e^k_i)}{g^k} - H(1 - x^k_{ij}) \leq \tau_j \qquad \forall i \in \bar{F}, j \in \mathcal{V}^k_{N+1}, i \neq j, k \in K \tag{6.103}$$

$$\tau_i + (\delta^u_{ij} + s_i)y^u_{ij} - H(1 - y^u_{ij}) \leq \tau_j \qquad \forall i \in V^u_0, j \in \mathcal{V}^u_{N+1}, i \neq j, u \in U \tag{6.104}$$

$$\tau_i = \tau_j \qquad \forall i, j \in \bar{A}^a, i \neq j, a \in A \tag{6.105}$$

$$\tau_i + \sum_{k \in K} \frac{(E^k_i - e^k_i)}{g^k} \leq \tau_j \qquad \forall i, j \in \bar{F}^f, i < j, f \in F \tag{6.106}$$

$$\tau_0 = 0 \tag{6.107}$$

$$\tau_{N+1} = H \tag{6.108}$$

Whereas the time-indexed MILP model presented earlier in this chapter did not account for user transition in the environment, our proposed model does so through the inclusion of Constraint (6.104). This constraint ensures if a user traverses arc $(i, j) \in \mathcal{A}^u$, at least δ^u_{ij}, the user transition time estimate, elapses between the departure from i and the arrival at j. In our empirical evaluation we relax the travel component from this constraint to facilitate a direct comparison to the previously proposed MILP model.

6.7.3.4 Robot Energy Constraints

The robot energy constraints are similar to the temporal constraints and ensure that each vertex with a potential robot visit is labeled with the remaining energy of the visiting robot. Constraint (6.109) dictates the energy level of a vertex given the preceding visit was a non-recharge vertex, while Constraint (6.110) dictates the energy level of a vertex given that the preceding visit was a recharge vertex. Constraint (6.111) ensures that, at a recharge station vertex, a social robot departs with more energy than it arrived with. We note that this modeling approach allows for partial recharges (i.e., those that do not fill their battery capacity). Constraint (6.112) states that robot energy at the starting depot is the maximum energy capacity of the robot, according to its type.

$$0 \leq e^k_j \leq e^k_i - (h^k d_{ij} + o^k_i s_i)x^k_{ij} + Q^k(1 - x^k_{ij}) \qquad \forall i \in \mathcal{V}^k_0 \setminus \bar{F}, j \in \mathcal{V}^k_{N+1}, i \neq j, k \in K \tag{6.109}$$

$$0 \leq e^k_j \leq E^k_i - (h^k d_{ij})x^k_{ij} + Q^k(1 - x^k_{ij}) \qquad \forall i \in \bar{F}, j \in \mathcal{V}^k_{N+1}, i \neq j, k \in K \tag{6.110}$$

$$e^k_i \leq E^k_i \leq Q^k \qquad \forall i \in \bar{F}, k \in K \tag{6.111}$$

$$e^k_0 = Q^k \qquad \forall k \in K \tag{6.112}$$

6.7.3.5 Side Constraints

We also summarize a set of side constraints in Eqns. (6.113) to (6.124). Constraints (6.113) and (6.114) dictate bounds on the number of participants in bingo games. Constraints (6.115) and (6.116) enforce bounds on a given user's participation in bingo games, based on their stated activity preferences. Constraints (6.117) through (6.119) ensure that each HRI activity is assigned to, and temporally satisfies, one of the prespecified time windows for the activity. Constraints (6.120) and (6.121) ensure that bingo game reminder tasks are delivered with the required proximity to the bingo game task itself.

$$g_{min} \sum_{k \in \gamma(a)} \sum_{i \in \bar{A}^a, j \in \mathcal{V}^k_{N+1}} x^k_{ij} \leq \sum_{u \in U} \sum_{i \in \bar{A}^a, j \in \mathcal{V}^u_{N+1}} y^u_{ij} \qquad \forall a \in G \tag{6.113}$$

$$\sum_{u \in U} \sum_{i \in \bar{A}^a, j \in \mathcal{V}^u_{N+1}} y^u_{ij} \leq g_{max} \sum_{k \in \gamma(a)} \sum_{i \in \bar{A}^a, j \in \mathcal{V}^k_{N+1}} x^k_{ij} \qquad \forall a \in G \tag{6.114}$$

$$g^u_{min} \leq \sum_{a \in G} \sum_{i \in \bar{A}^a, j \in \mathcal{V}^k_{N+1} \setminus \bar{A}^a} y^u_{ij} \qquad \forall u \in U \tag{6.115}$$

$$\sum_{a \in G} \sum_{i \in \bar{A}^a, j \in \mathcal{V}^k_{N+1} \setminus \bar{A}^a} y^u_{ij} \leq g^u_{max} \qquad \forall u \in U \tag{6.116}$$

$$\tau_i \geq \underline{p} - H(1 - \eta_{ap}) \qquad \forall p \in P_a, i \in \bar{A}^a, a \in S \cup G \tag{6.117}$$

$$\tau_i \leq \overline{p} - s_a + H(1 - \eta_{ap}) \qquad \forall p \in P_a, i \in \bar{A}^a, a \in S \cup G \tag{6.118}$$

$$\sum_{p \in P_a} \eta_{ap} = 1 \qquad \forall a \in S \cup G \tag{6.119}$$

$$\tau_j \leq \tau_i - \omega_{min} - s_j \qquad \forall j \in \bar{A}^{a'}, a' \in M_{ua}, i \in \bar{A}^a, a \in G, u \in U \tag{6.120}$$

$$\tau_j \geq \tau_i - \omega_{max} - s_j - H(1 - \sum_{j \in \bar{A}^{a'}, k \in \mathcal{V}^u} y^u_{jk}) \qquad \forall j \in \bar{A}^{a'}, a' \in M_{ua}, i \in \bar{A}^a, a \in G, u \in U \tag{6.121}$$

The final set of side constraints ensures that reminder HRIs are delivered by robots to locations that the user is both: i) located in during that time, and ii) available for an interaction (i.e., $\{\lambda \in \Lambda^u \mid \mu(\lambda) = 1\}$). This accomplished with the following set of constraints:

$$\tau_i \geq \theta_\lambda - H(1 - \epsilon_{a,\ell,\lambda}) \qquad \forall i \in \bar{A}^{a,\ell}, \lambda \in \Lambda^{\alpha(a)} \mid \ell_\lambda = \ell \wedge \mu(\lambda) = 1, \ell \in \zeta(a), a \in M \tag{6.122}$$

$$\tau_i \leq \theta_\lambda + s_\lambda - s_a + H(1 - \epsilon_{a,\ell,\lambda}) \qquad \forall i \in \bar{A}^{a,\ell}, \lambda \in \Lambda^{\alpha(a)} \mid \ell_\lambda = \ell \wedge \mu(\lambda) = 1, \ell \in \zeta(a), a \in M \tag{6.123}$$

$$\sum_{k \in \gamma(a)} \sum_{i \in \bar{A}^{a,\ell}, j \in \mathcal{V}^k \setminus \bar{A}^a} x^k_{ij} = \epsilon_{a,\ell,\lambda} \qquad \forall \lambda \in \Lambda^{\alpha(a)} \mid \ell_\lambda = \ell \wedge \mu(\lambda) = 1, \ell \in \zeta(a), a \in M \tag{6.124}$$

Constraints (6.122) and (6.123) ensure that the reminder is delivered to the vertex associated with the activity and location, $\bar{A}^{a,\ell}$, at a time bounded by the beginning of the available interval, θ_λ, and the end of the available interval minus task duration, $\theta_\lambda + s_\lambda - s_a$. Constraint (6.124) links robot facilitation flow to the reminder location variables, $\epsilon_{a,\ell,\lambda}$.

6.7.3.6 Symmetry Breaking

We include a series of symmetry breaking constraints for recharge station visits and bingo game activities. Constraint (6.125) ensures that the recharge visits at any given station, $f \in F$, across the robot fleet are consumed in a prespecified order. Similarly, Constraint (6.126) ensures that symmetric bingo game activities are facilitated in a prespecified order.

$$\sum_{k\in K}\sum_{l\in\mathcal{V}^k_{N+1}} x^k_{il} \geq \sum_{k\in K}\sum_{l\in\mathcal{V}^k_{N+1}} x^k_{jl} \qquad \forall i,j\in\bar{F}^f \mid i<j, f\in F \tag{6.125}$$

$$\sum_{k\in K}\sum_{i\in\bar{A}^a, j\in\mathcal{V}^k_{N+1}} x^k_{ij} \geq \sum_{k\in K}\sum_{i\in\bar{A}^{a'}, j\in\mathcal{V}^k_{N+1}} x^k_{ij} \qquad \forall a,a'\in G \mid a<a' \tag{6.126}$$

6.7.3.7 Variable Domains

The specific domains of the variables in our proposed MILP formulation are given as follows:

$$x^k_{ij}\in\{0,1\} \qquad \forall i,j\in\mathcal{V}^k_{0,N+1}, i\neq j, i\neq N+1, j\neq 0 \tag{6.127}$$

$$x^k_{ij}=0 \qquad \forall i,j\in\bar{F}^f, f\in F, k\in K \tag{6.128}$$

$$0\leq e^k_i\leq Q^k \qquad \forall i\in\mathcal{V}^k_{N+1}, k\in K \tag{6.129}$$

$$0\leq E^k_i\leq Q^k \qquad \forall i\in\bar{F}^f, f\in F, k\in K \tag{6.130}$$

$$y^u_{ij}\in\{0,1\} \qquad \forall i,j\in\mathcal{V}^u, i\neq j \tag{6.131}$$

$$0\leq\tau_i\leq|T| \qquad \forall i\in\mathcal{V}_{0,N+1} \tag{6.132}$$

$$\eta_{ap}\in\{0,1\} \qquad \forall p\in P_a, a\in S\cup G \tag{6.133}$$

$$\epsilon_{a,\ell,\lambda}\in\{0,1\} \qquad \forall\lambda\in\Lambda^{\alpha(a)} \mid \ell_\lambda=\ell\wedge\mu(\lambda)=1, \ell\in\zeta(a), a\in M \tag{6.134}$$

6.7.4 Valid Inequalities

We can tighten the linear relaxation of our MILP formulation through the inclusion of a number of valid inequalities. Often the model uses disjunctive labeling constraints which are known to have poor linear relaxations [110]. We endeavor to strengthen them through a known lifting procedure.

Proposition 6.7.1. *For any pair of non-recharge, non-depot vertices, $i,j\in\mathcal{V}^k\setminus\bar{F}$, Constraint* (6.109) *can be lifted to produce the following inequality:*

$$e^k_j\leq e^k_i-(h^kd_{ij}+o^k_is_i)x^k_{ij}+Q^k(1-x^k_{ij})-(Q^k-(h^kd_{ji}+o^k_js_j))x^k_{ji} \qquad \forall i,j\in\mathcal{V}^k\setminus\bar{F} \tag{6.135}$$

Proof. We follow the lifting procedure presented in Kara et al. [110]. Consider the constraint:

$$e^k_j\leq e^k_i-(h^kd_{ij}+o^k_is_i)x^k_{ij}+Q^k(1-x^k_{ij})-\beta x^k_{ji} \tag{6.136}$$

where $\beta=0$ currently. We would like to compute the largest β such that the constraint is still satisfied. There are three cases to consider: i) $x^k_{ij}=0, x^k_{ji}=0$, ii) $x^k_{ij}=1, x^k_{ji}=0$ and ii) $x^k_{ij}=0, x_{ji}=1$ ($x^k_{ij}=x^k_{ji}=1$ is not permitted).

Case 1. $x_{ij}=0$ and $x^k_{ji}=0$ results in a redundant constraint.

Case 2. $x^k_{ij}=1, x^k_{ji}=0$ results in Eqn. (6.136) being satisfied for any β.

Case 3. $x^k_{ij}=0, x^k_{ji}=1$ indicates vertex $i\in\mathcal{V}\setminus\bar{F}$ is visited immediately after $j\in\mathcal{V}^k\setminus\bar{F}$ by robot type k, yielding $e^k_i=e^k_j-(h^kd_{ji}+o^k_js_j)$. Substituting this with $x^k_{ij}=0$ into Eqn. (6.136) yields:

$$0\leq e^k_j-(h^kd_{ji}+o^k_js_j)-e^k_j+Q^k-\beta \tag{6.137}$$

which simplifies to:

$$\beta \leq Q^k - (h^k d_{ji} + o_j^k s_j) \tag{6.138}$$

which, upon substitution into Eqn. (6.136) results in the proposed inequality Constraint (6.135). $\square$

While it would be desirable to apply the same lifting procedure to the temporal constraints involving the τ_i variables, the numerous side constraints in the problem result in solutions where social robots, and users, spend time waiting until they can participate in the next activity. This precludes a straightforward implementation of the lifting procedure as the labeling constraint needs to afford some slack in the values.

We are also able to add the following family of inequalities to the formulation:

$$x_{ij}^k + x_{ji}^k \leq 1 \qquad \forall i \in \mathcal{V}_0^k, j \in \mathcal{V}_{N+1}^k, i \neq j, k \in K \tag{6.139}$$

Constraints (6.139) state that at most one of the two arcs, (i, j) or (j, i), must be selected. While these constraints are fairly straightforward, they improve the quality of the linear relaxation and can potentially decrease search effort.

As described in Section 6.9.1, our preliminary experiments indicated that Constraint (6.135) slightly weakened the augmented graph MILP model's performance. Thus, Constraint (6.139) is included in our experiments but Constraint (6.135) is not.

6.7.5 Model Discussion

The proposed MILP model is significantly different than the previously proposed time-indexed model presented in Section 6.4.1. First, it takes advantage of symmetric robots by encoding routing between vertices into the x_{ij}^k variable which is indexed by robot *type*, $k \in K$. This is in contrast to the time-indexed model which uses sequencing variable $z_{raa'}$, indexing each explicit robot $r \in R$. Second, the model does not use any time-indexed reasoning. Arrival time variables, τ_i, are constrained by appropriate task time windows, and temporal relationships are enforced via disjunctive constraints with appropriate big-M values. Furthermore, where possible, these constraints are lifted to produce even stronger inequalities. Finally, the proposed MILP models heterogeneous robots, a characteristic not properly captured by the time-indexed model as discussed in Section 6.4.1.

6.8 Augmented Graph CP Model

In this section, we detail the proposed CP model for the SRRP-RH. Due to the large number of synchronization constraints between social robot and user routes, and the relatively low number of robots in the fleet, the single resource transformation, as investigated in Chapter 4, was found, through initial experimentation, to be ineffective for this particular problem. Activity synchronization required excessive use of the Mod constraint (which takes the modulo of a given integer variable value) to align start times between augmented robot and non-augmented user horizons, which, in on our testing, was found to have poor propagation/performance when combined with the numerous other side constraints.

As a result, the proposed model is based on the augmented graph and formulated following the alternative resource model for electric vehicle routing [23], as summarized in Chapter 4. The model is a

Variable	**Type**	**Description**
x_i^r	Interval	Robot $r \in R$ visit at vertex $i \in \mathcal{V}_{0,N+1}^{\psi(r)}$
π^r	Sequence	Sequence variable for robot $r \in R$
$\mathcal{Q}^r$	Cumul. function expr.	Energy level for robot $r \in R$
y_i^u	Interval	User u visit at vertex $i \in \mathcal{V}_{0,N+1}^u$
ρ^u	Sequence	Sequence variable for user u

Table 6.6: Decision variables for augmented graph CP formulation

revised version of an earlier proposed model [25] and bears many similarities to the previously proposed CP model presented earlier in this chapter [199].

6.8.1 Decision Variables

The decision variables used in the formulation are defined in Table 6.6. Recall that the function $\psi(r)$ returns the robot type, $k \in K$, of robot $r \in R$. The formulation uses interval, cumulative function expression, and sequence variables. Interval variable x_i^r represents a visit by robot $r \in R$ to vertex $i \in \mathcal{V}_{0,N+1}^{\psi(r)}$. The sequence variable, π^r, is defined over all the potential visits robot $r \in R$ may make, and represents the permutation of these visits. Cumulative function expression $\mathcal{Q}^r$ represents the energy level of robot $r \in R$ throughout the day. Interval variable y_i^u indicates a visit by user $u \in U$ at vertex $i \in \mathcal{V}_{0,N+1}^u$. Finally, sequence variable ρ^u is defined over the potential visits of a user and represents the permutation of these visits.

6.8.2 Objective Function

The objective function is given by (6.140), expressing the same multiobjective function that was used for the MILP model.

$$
\begin{aligned}
\min \quad & \sum_{a \in G, i \in \bar{A}^a} \left(1000 \cdot \Big(|U| - \sum_{u \in U} \textsc{Pres}(y_i^u) \Big) + 500 \cdot \Big(1 - \sum_{r \in R} \textsc{Pres}(x_i^r) \Big) \right) \qquad (6.140) \\
& + \sum_{a \in A, i \in \bar{A}^a} \sum_{r \in R} \textsc{Length}(x_i^r) o_a^{\psi(r)} + \sum_{r \in R} \sum_{i \in \mathcal{V}_{N+1}^{\psi(r)}} \textsc{Pres}(x_i^r) \cdot h^{\psi(r)} d_{\textsc{Prev}_{\pi^r}(i), i} \\
& + \sum_{a \in G, i \in \bar{A}^a} \sum_{u \in U} \sum_{a' \in M_{ua}, j \in \bar{A}^{a'}} \textsc{Pres}(y_j^u) \cdot (\textsc{Start}(y_i^u) - \textsc{End}(y_j^u)) \\
& \text{s.t. Constraints (6.141) through (6.176)}
\end{aligned}
$$

In the objective function, following the structure presented in the MILP, the first line represents the cost associated with bingo game facilitation and participation. The second line represents the cost associated with the total energy expended by the social robot fleet, and the final line the cost associated with bingo game reminder proximity.

6.8.3 Constraints

We summarize the main model constraints in Eqn. (6.141) through (6.167). These constraints capture the key elements of the problem: temporal constraints, HRI activity requirements, robot energy, and side constraints.

6.8.3.1 Temporal Constraints

As before, the temporal constraints make sure task participation and facilitation are feasible with respect to time. Constraint (6.141) ensures robot activity facilitation and recharge visits do not overlap, including transition times between locations. Interval variables associated with start and end dummy tasks are constrained in Constraint (6.142). Similarly, Constraint (6.143) ensures user tasks do not interfere temporally, including user transition times, while Constraint (6.145) ensures that each recharge station is used once at a time (i.e., a unary capacity resource).

$$\textsc{NoOverlap}(\pi^r, \{\delta_{ij}^{(\psi(r)} : (i,j) \in \mathcal{A}^{\psi(r)}\}) \qquad \forall r \in R \tag{6.141}$$

$$\textsc{First}(\pi^r, x_0^r), \quad \textsc{Last}(\pi^r, x_{N+1}^r) \qquad \forall r \in R \tag{6.142}$$

$$\textsc{NoOverlap}(\rho^u, \{\delta_{ij}^u : (i,j) \in \mathcal{A}^u\}) \qquad \forall u \in U \tag{6.143}$$

$$\textsc{First}(\rho^u, x_0^u), \quad \textsc{Last}(\rho^u, y_{N+1}^u) \qquad \forall u \in U \tag{6.144}$$

$$\textsc{NoOverlap}(\{x_i^r : i \in \bar{F}^f, r \in R\}) \qquad \forall f \in F \tag{6.145}$$

The previously proposed alternative resource CP model [200] did not account for user transition in the environment, while our model does through the inclusion of Constraint (6.143). In a portion of our empirical evaluation we relax the travel component from this constraint to facilitate a direct comparison to the previously proposed CP model.

6.8.3.2 HRI Activity Constraints

The set of HRI activity constraints can be concisely expressed in CP. Constraint (6.146) uses the Alternative global constraint to enforce that if a user participates in an activity, $a \in S \cup M$, at a particular location, represented by vertex $i \in \bar{A}^a$, exactly one social robot facilitates this activity. This works since both telepresence sessions and reminders are one-to-one activities. Since bingo game activities are one-to-many, we use Constraints (6.147) and (6.148) to constrain activity start time and presence. Constraint (6.150) then ensures that if a user participates in a bingo activity, they also receive the associated reminder activity.

$$\textsc{Alternative}(y_i^u, \{x_i^r : \forall r \in R\}) \qquad \forall i \in \bar{A}^a, a \in S \cup M \tag{6.146}$$

$$\textsc{StartAtStart}(x_i^r, y_i^u, 0) \qquad \forall i \in \bar{A}^a, \forall r \in R, \forall u \in U, a \in G \tag{6.147}$$

$$\textsc{Pres}(y_i^u) \leq \sum_{r \in R} \textsc{Pres}(x_i^r) \leq 1 \qquad \forall i \in \bar{A}^a, u \in U, a \in G \tag{6.148}$$

$$\tag{6.149}$$

$$\sum_{i \in \bar{A}^a} \textsc{Pres}(y_i^u) = \sum_{i \in \bar{A}^{a'}} \textsc{Pres}(y_i^u) \qquad \forall a \in G, a' \in M_{ua}, u \in U \tag{6.150}$$

6.8.3.3 Robot Energy Constraints

Robot energy modeling is accomplished via the use of cumulative function expression variables and constraints that enforce step impacts on those variables. Constraint (6.151) defines the various step impacts to cumulative function expression variable $\mathcal{Q}^r$ for robot $r \in R$ throughout the course of the day. Each present interval variable has a decrement on vehicle energy with magnitude equal to the travel consumption to get to that task. Recharge interval variables have an increment at the end of the associated interval variable with magnitude equal to the amount recharged. Constraint (6.152) ensures that the energy level of a robot always stays within permissible bounds. Constraint (6.153) limits the number of times a recharge at a particular recharge station can be used across the social robot fleet.

$$
\begin{aligned}
\mathcal{Q}^r = {} & \textsc{StepAtStart}(x_0^r, Q^{\psi(r)}) \\
& - \sum_{i \in V_{N+1}^k} \textsc{StepAtStart}(x_i^r, h^{\psi(r)} \cdot d_{\textsc{Prev}_{\pi^r}(i),i}) \\
& - \sum_{a \in A, i \in \bar{A}^a} \textsc{StepAtStart}(x_i^r, o_a^{\psi(r)} \cdot s_a) \\
& + \sum_{i \in \bar{F}} \textsc{StepAtEnd}(x_i^r, g^{\psi(r)} \cdot \textsc{Length}(x_i^r)) && (6.151)
\end{aligned}
$$

$$\textsc{AlwaysIn}(\mathcal{Q}^r, [0, H], [0, Q^{\psi(r)}]) \qquad \forall r \in R \qquad (6.152)$$

$$\sum_{r \in R} \sum_{i \in \bar{F}^f} \textsc{Pres}(x_i^r) \leq n_f \qquad \forall f \in F \qquad (6.153)$$

6.8.3.4 Side Constraints

The side constraints of our SRRP-RH are detailed as follows. Constraints (6.156) and (6.157) define the lower and upper bounds on user participation for a given bingo game, $a \in G$. Constraints (6.156) and (6.157) define lower and upper bounds on a particular user's bingo game participation throughout the day. Constraint (6.158) ensures that if a user participates in a bingo game HRI, its associated reminder ends at least ω_{min} time units prior the bingo game. Similarly, Constraint (6.159) ensures the associated reminder ends no more than ω_{max} time units before the game. Constraint (6.160) enforces activity-level time windows on the execution of user activities. The set $\mathcal{T}_a = \{t \mid t \in \bigcup_{p \in P_a} [\underline{p}, \overline{p}]\}$ identifies the time points that an activity $a \in S \cup G$ can potentially execute during. Constraint (6.161) ensures that a reminder activity is only executed at a time and location that agree with user calendars, where the set $\mathcal{T}_{a,\ell} = \{t \mid t \in \bigcup_{\lambda \in \Lambda^{\alpha(a)} | \ell_\lambda = \ell \wedge \mu(\lambda)=1} [\theta_\lambda, \theta_\lambda + s_\lambda]\}$ identifies the time points that a reminder in a given location can be executed.

$$g_{min} \sum_{r \in R} \sum_{i \in \bar{A}^a} \textsc{Pres}(x_i^r) \leq \sum_{u \in U} \sum_{i \in \bar{A}^a} \textsc{Pres}(y_i^u) \qquad \forall a \in G \qquad (6.154)$$

$$\sum_{u \in U} \sum_{i \in \bar{A}^a} \textsc{Pres}(y_i^u) \leq g_{max} \sum_{r \in R} \sum_{i \in \bar{A}^a} \textsc{Pres}(x_i^r) \qquad \forall a \in G \qquad (6.155)$$

$$g_{min}^u \leq \sum_{a \in G} \sum_{i \in \bar{A}^a} \textsc{Pres}(y_i^u) \qquad \forall u \in U \qquad (6.156)$$

$$\sum_{a \in G} \sum_{i \in \bar{A}^a} \textsc{Pres}(y_{ij}^u) \leq g_{max}^u \qquad \forall u \in U \qquad (6.157)$$

$$\textsc{EndBeforeStart}(y_j^u, y_i^u, \omega_{min}) \qquad \forall j \in \bar{A}^{a'}, a' \in M_{ua}, i \in \bar{A}^a, a \in G, u \in U \quad (6.158)$$
$$\textsc{StartBeforeEnd}(y_i^u, y_j^u, -\omega_{max}) \qquad \forall j \in \bar{A}^{a'}, a' \in M_{ua}, i \in \bar{A}^a, a \in G, u \in U \quad (6.159)$$
$$\textsc{ForbidExtent}(y_i^u, \{t : t \notin \mathcal{T}_a\}) \qquad \forall i \in \bar{A}^a, a \in S \cup G, u \in \alpha(a) \quad (6.160)$$
$$\textsc{ForbidExtent}(y_i^u, \{t : t \notin \mathcal{T}_{a,\ell}\}) \qquad \forall i \in \bar{A}^{a,\ell}, \ell \in \zeta(a), a \in M \quad (6.161)$$

6.8.3.5 Model Strengthening

In order to improve the formulation, we also include a number of redundant strengthening constraints. These constraints are primarily aimed at informing the robot facilitation variables, x_i^r, about relationships stated over the user participation variables, y_i^u.

$$g_{min} \sum_{r \in R} \sum_{i \in \bar{A}^a} \textsc{Pres}(x_i^r) \leq \sum_{u \in U} \sum_{r \in R} \sum_{j \in \bar{A}^{a'}, a' \in M_{ua}} \textsc{Pres}(x_j^r) \qquad \forall a \in G \quad (6.162)$$
$$\sum_{u \in U} \sum_{r \in R} \sum_{j \in \bar{A}^{a'}, a' \in M_{ua}} \textsc{Pres}(x_j^r) \leq g_{max} \sum_{r \in R} \sum_{i \in \bar{A}^a} \textsc{Pres}(x_i^r) \qquad \forall a \in G \quad (6.163)$$
$$\sum_{u \in u} \Big(\sum_{a \in A, i \in \bar{A}^a} \textsc{Pulse}(y_i^u) + \sum_{i \in \bar{\Lambda}^u} \textsc{Pulse}(y_i^u) \Big) \leq |U| \quad (6.164)$$
$$\sum_{r \in R} \sum_{a \in A, i \in \bar{A}^a} \textsc{Pulse}(x_i^r) + \sum_{u \in U} \sum_{i \in \bar{\Lambda}^u} \text{Pulse}(y_i^u) \leq |U| \quad (6.165)$$

$$\textsc{EndBeforeStart}(x_j^r, x_i^r, \omega_{min}) \qquad \forall j \in \bar{A}^{a'}, a' \in M_{ua}, i \in \bar{A}^a, r \in R, a \in G, u \in U \quad (6.166)$$
$$\textsc{StartBeforeEnd}(x_i^r, x_j^r, \omega_{-max}) \qquad \forall j \in \bar{A}^{a'}, a' \in M_{ua}, i \in \bar{A}^a, r \in R, a \in G, u \in U \quad (6.167)$$

Constraints (6.162) and (6.163) enforce the same relationship as Constraints (6.154) and (6.155), except in terms of robot facilitation variables instead of user variables. Constraint (6.164) ensures that, across all users, at most $|U|$ events (i.e., HRIs or calendar commitments) are happening at once. Constraint (6.165) enforces a similar constraint, stating at most $|U|$ robot HRI facilitations or user calendar commitments can execute at once. These constraints represent a more global view of the problem. Finally, Constaints (6.166) and (6.167) constrain reminder proximity to associated bingo games over the robot facilitaton variables.

6.8.3.6 Symmetry Breaking

It was found, through initial experimentation, that the inclusion of symmetry breaking constraints eroded the CP solver's ability to quickly find initial solutions. This finding is in-line with the performance degradation sometimes noted with symmetry breaking in previous work [25]. Due to these observations, we elected to forgo symmetry breaking constraints in our proposed CP formulation.

6.8.3.7 Variable Domains

The specific domains of the variables in our proposed CP formulation are given as follows:

$$x_0^r : \textsc{IntervalVar}(0, 0), \;\; x_{N+1}^r : \textsc{IntervalVar}(0, |T|) \qquad \forall r \in R \quad (6.168)$$
$$x_i^r : \textsc{OptIntervalVar}(s_a, [0, |T| - s_a]) \qquad \forall i \in \bar{A}^a, a \in A, r \in R \quad (6.169)$$

$$x_i^r : \textsc{OptIntervalVar}([0, \frac{Q^{\psi(r)}}{g^{\psi(r)}}], [0, |T|]) \qquad \forall i \in \bar{F}, r \in R \tag{6.170}$$

$$\pi^r : \textsc{SequenceVar}(\{x_i^r : i \in \mathcal{V}_{0,N+1}^k\}) \qquad \forall r \in R \tag{6.171}$$

$$\mathcal{Q}^r : \textsc{CumulFunctionExpr}([0, Q^{\psi(r)}]) \qquad \forall r \in R \tag{6.172}$$

$$y_i^u : \textsc{IntervalVar}(\theta_i, s_i) \qquad \forall i \in \bar{\Lambda}^u, u \in U \tag{6.173}$$

$$y_i^u : \textsc{IntervalVar}(s_a, [0, |T| - s_a]) \qquad \forall i \in \bar{A}^a, a \in S, u \in \alpha(a) \tag{6.174}$$

$$y_i^u : \textsc{OptIntervalVar}(s_a, [0, |T| - s_a]) \qquad \forall i \in \bar{A}^a, a \in A, u \in U \tag{6.175}$$

$$\rho^u : \textsc{SequenceVar}(\{y_i^u : i \in \mathcal{V}^u\}) \qquad \forall u \in U \tag{6.176}$$

6.8.4 Model Discussion

The proposed CP model is similar to the previously proposed model [199] in the sense that both utilize alternative resource strategies. Our model is different, however, in how it captures user transition in the environment, and through the method that it links user and robot variables. Instead of modeling user unavailability with the ForbidExtent constraint, we represent calendar commitments as fixed, mandatory interval variables and use the NoOverlap constraint in Eqn. (6.143). This constraint reasons over the permutation of interval variables in sequence variable ρ^u, in addition to user transition times between adjacent pairs of these variables. Additionally, the previously proposed approach introduces an interval variable for each bingo activity and subsequently links all of the user participation and robot facilitation options through clone tasks. Our formulation, however, binds robot and user variables directly together with no intermediate HRI variable. We propose a number of strengthening constraints to help improve propagation from user schedules to robot schedules and vice-versa.

6.9 Experimental Results and Analysis

In this section we present an empirical evaluation of our proposed MILP and CP formulations, comparing them to our implementation of those from the literature. We conduct two classes of experiments: those with *relaxed* user travel, and those with *enforced* user travel. The first class of experiments assumes that resident users are able to move between locations instantly; all formulations are applied to these problems. The second class of experiments requires user travel throughout the environment to be considered. For these experiments we only test the new models proposed in this chapter.

We evaluate our methods in terms of their ability to produce feasible solutions, their ability to find and prove optimality, and their ability to produce solutions that involve at least some bingo game HRIs.

6.9.1 Setup

All experiments are implemented on in C++ and run on the Compute Canada Niagara computing cluster operated by SciNet (http://www.scinetpc.ca). The cluster runs the Linux CentOS 7 operating system and uses Skylake cores at 2.4 GHz. We use CP Optimizer for the CP models and CPLEX for the MILP model from the IBM ILOG CPLEX Optimization Studio version 12.9. All experiments are single-threaded with default inference settings and use a time limit of one hour. Experimentation indicated that the valid inequalities given by Constraint (6.135) slightly eroded the augmented graph

Scenario	**Users**	**Robots**	**Telepresence**	**Bingo**	**Recharge**	**Leisure**
S1	5	2	2	1	2	3
S2	10	2	4	2	3	3
S3	15	3	6	3	3	4
S4	20	3	8	4	4	4

Table 6.7: SRRP-RH experimental scenarios.

MILP model's performance. As such, our experiments include Constraint (6.139) but do not include Constraint (6.135).

6.9.2 Problem Instances

In generating our main benchmark set we follow the parameters introduced in Tran et al. [199]. Aside from meal times, busy intervals in user calendars, Λ^u, are generated randomly and have duration from 30 minutes to 1 hour. Users do not need to be involved in bingo games, and are willing to participate in at most one game per day (i.e., $g^u_{min} = 0, g^u_{max} = 1, \forall u \in U$). The facility-mandated time windows for telepresence sessions and bingo games are 8:00 AM to 7:00 PM (i.e., these HRIs cannot be held during the first hour of the planning horizon). Following Tran et al. [199], we use a homogeneous fleet of social robots, $|K| = 1$, where each robot is capable of facilitating all HRIs. User speed is set to $\nu^u = 60$ meters per minute and robot movement speed in the environment is $\nu^k = 20$ meters per minute. Minimum and maximum battery capacity is defined by $Q^k_{min} = 0, Q^k_{max} = 20$ energy units, where robot movement consumption is $h^k = 0.04$ energy units per meter, and robot HRI facilitation, in energy units per minute, is $o^k_a = 0.1, \forall a \in A$. Finally, robots recharge at a rate of $g^k = 0.5$ energy units per minute, requiring 40 minutes for a full recharge.

We generate problem instances according to four different scenarios as illustrated in Table 6.7. Each scenario varies the number of users, social robots, telepresence HRIs, bingo game HRIs, recharge stations, and leisure locations. Scenario S1 is the smallest and S4 the largest. Due to the augmentation of the graph representation, the number of vertices and arcs in the graph grows dramatically as the scenario size increases. For each scenario size, 20 instances are randomly generated using the aforementioned parameters for a total of 80 instances. The retirement home environment, $\mathcal{L}$, is generated by producing a series of (x, y) coordinates representing the various locations, where the maximum distance between any pair of points is set to 200 meters.

As discussed in previous chapters, the augmentation of the recharge stations is an important step as the value for n_f selected can have significant impact on the underlying models. For these experiments, we wish to preserve the completeness of our approaches. Recall that n_f represents the number of times a recharge station, $f \in F$, can be visited across the entire social robot fleet.

Lemma 6.9.1. *En-route to the robot depot or facilitating an HRI activity, a robot, $r \in R$, does not need to visit a recharge station, $f \in F$, more than once in an optimal solution to the SRRP-RH.*

Proof. Due to the triangle inequality and the positive valued, static energy consumption rate per unit distance, h^k for robot type $k \in K$, the most energy efficient route will always be the shortest. Visiting a recharge station more than once en-route to the depot or a given HRI activity implies that the robot traveled elsewhere in between station visits, expending more energy and consuming more time than required. $\square$

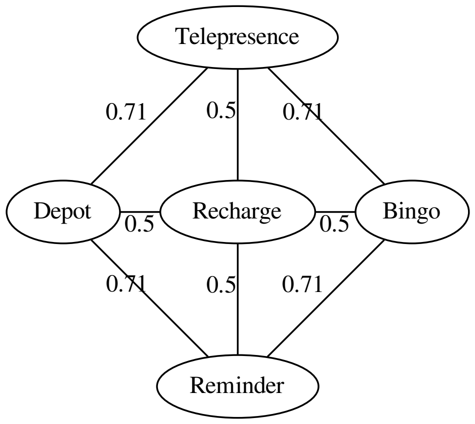


Figure 6.6: Example. SRRP-RH instance, recharge station augmentation.

Proposition 6.9.1. *The robot fleet will not visit a given recharge station more than* $n_f = |R| + |S| + |G| + \sum_{u \in U} g^u_{max}$ *times in an optimal solution to the SRRP-RH.*

Proof. By the definition of the problem, the maximum number of HRI activities the robot fleet can facilitate is $|S| + |G| + \sum_{u \in U} g^u_{max}$, where each user participates in their maximum preferred number of bingo games ($\sum_{u \in U} g^u_{max}$ representing the bingo game reminder HRIs). Since one robot facilitates each activity and, due to Lemma 6.9.1, each recharge station will be visited at most once by a robot en-route to facilitating a given activity, the fleet will visit a given recharge station $f \in F$ at most $|S| + |G| + \sum_{u \in U} g^u_{max}$ times to conduct the HRIs. The fleet also may need to visit a given recharge station one more time en-route back to the depot, adding $|R|$ additional potential visits, and so we get $n_f = |R| + |S| + |G| + \sum_{u \in U} g^u_{max}$. □

Figure 6.6 provides a minimal SRRP-RH example where any solution requires the aforementioned number of recharge visits to the one recharge station.

Example 6.9.1. *Consider an SRRP-RH example where the social robot fleet must facilitate a telepresence HRI, a bingo game HRI involving one user, and its corresponding reminder activity. The HRIs are spatially distributed as shown in Figure 6.6, where the edge weights are the distances between vertices. Consider a homogeneous social robot fleet, starting and ending at the depot, where each robot can travel at most one unit of distance before requiring a recharge. To facilitate the three HRIs in this example the recharge station will be visited, at the least,* $|R| + |S| + |G| + \sum_{u \in U} g^u_{max} = |R| + 1 + 1 + 1$ *times. If* $|R| = 1$*, the recharge station is visited four times, with* $|R| = 2$*, the recharge station is visited five times, and with* $|R| = 3$ *(one robot facilitating each HRI), the recharge station will be visited six times across the fleet.*

We note that the previously proposed time-indexed MILP and both CP models will multiply these recharge stations across all symmetric robots, while the augmented graph MILP model we propose in this chapter does not by taking advantage of robot symmetry.

	MILP (Proposed)					**Time-Indexed MILP** [200]				
	# Feas.	**# Best**	**# Opt.**	**TTF**	**Bingo**	**# Feas.**	**# Best**	**# Opt.**	**TTF**	**Bingo**
Fleet Distance (*minimization*)										
S1	**20**	**20**	**18**	0.2	0	**20**	**20**	10	0.5	0
S2	**20**	14	0	58.9	0	**20**	**17**	0	28.5	0
S3	**16**	**16**	0	136.9	0	3	2	0	1791.3	0
S4	**2**	**2**	0	734.9	0	0	0	0	–	0
Bingo Participation (*maximization*)										
S1	**20**	**20**	**19**	0.5	**19**	**20**	**20**	**19**	0.5	**19**
S2	17	15	0	513.0	0	**20**	**20**	0	420.8	**2**
S3	**14**	**14**	0	629.7	0	2	2	0	2243.2	0
S4	0	0	0	–	0	0	0	0	–	0
Complex Objective										
S1	**20**	**19**	**6**	0.5	**19**	**20**	14	1	0.6	18
S2	18	11	0	123.9	**3**	**20**	**16**	0	133.8	**3**
S3	**12**	**12**	0	496.2	0	1	1	0	651.0	0
S4	**3**	**3**	0	939.8	0	0	0	0	–	0

Table 6.8: SRRP-RH results. Relaxed user travel MILP formulations. Bold values indicate the best result out of the two methods.

6.9.3 Experiments: Relaxed User Travel

Our first empirical assessment compares the performance of all of the approaches (both existing and proposed) on the *user travel relaxed* variant of the problem, following the initial problem definition in Tran et al. [199, 200]. In order to compare our proposed augmented graph-based models with those previously proposed, we alter the constraints enforcing user transitions. Constraint (6.104) in the proposed MILP model is relaxed and expressed as follows:

$$\tau_i + s_i y_{ij}^u - H(1 - y_{ij}^u) \leq \tau_j \qquad \forall i \in V_0^u, j \in \mathcal{V}_{N+1}^u, i \neq j, u \in U \tag{6.177}$$

Similarly, Constraint (6.143) in the proposed CP is relaxed and expressed as follows:

$$\textsc{NoOverlap}(\rho^u) \qquad \forall u \in U \tag{6.178}$$

Each of these alterations removes the parameter δ_{ij}^u, representing user transition time from vertex i to j, from the constraint, effectively allowing retirement home residents to immediately perform one HRI after another, regardless of their location in the environment.

6.9.3.1 MILP Approaches

The relaxed user travel results for the MILP approaches, our proposed MILP formulation and the previously proposed time-indexed model [200], are detailed in Table 6.8. For each approach, objective function, and scenario size, we detail the number of feasible solutions found (# Feas.), the number of problems for which the approach had the best solution (out of the approaches in the table) at runtime (# Best), and the number of problems for which the approach found and proved the optimal solution (# Opt.). We also detail the average runtime to first solution found (across instances for which a solution was found) (TTF), and the number of instances that had *at least one* bingo game scheduled (Bingo).

With respect to feasibility, it is clear that the proposed MILP approach matches or outperforms the existing model for nearly all scenario sizes and objectives. While both methods are able to find feasible solutions for all objectives for scenario S1, and nearly all objectives for scenario S2, the proposed MILP is able to find feasible solutions to considerably more of the S3 instances. Specifically, it is able to find feasibility for 16/20 (80%) instances under fleet distance minimization, and 14/20 (70%) instances when maximizing bingo participation, and 12/20 (60%) for the complex objective function. The existing MILP, on the other hand, is only able to find feasibility for 3/20 (15%), 2/20 (10%), and 1/20 (5%) of instances for S3 problems for the fleet distance, bingo participation, and complex objectives, respectively. The proposed MILP is also able to find feasible solutions for a small fraction of S4 instances, whereas the time-indexed MILP cannot find feasible solutions for any of these problems.

Given that it finds feasible solutions to more problems, it follows that the proposed MILP approach would also have more high-quality solutions by the end of the runtime. Looking at Table 6.8, we can see that the proposed MILP finds the best solution more times than the time-indexed model for all scenarios and objectives except for S2. An interesting observation is that, though the time-indexed model finds few feasible solutions, when it does find a feasible solution it is usually of high quality. In terms of finding and proving optimal solutions, it would appear that for small instances, S1, the proposed MILP outperforms the existing model, overall. For larger instances, S2-S4, both methods have difficulty proving optimality.

A direct comparison of time-to-first (TTF) results is difficult to conduct, as the metric is calculated over all instances for which a given approach found a feasible solution. Regardless, we can observe that, for both methods, as the instances get larger, considerably more time is taken by the approach to find a feasible solution. For S1 and S2 scenarios, given that both methods found feasibility to all problems, we note a slight advantage to the time-indexed method. For larger problems, however, the proposed MILP appears to maintain much lower average TTF values.

Finally, the ‘Bingo’ column provides a measure of how many final solutions included any bingo game participation by residents. The results for both approaches under fleet distance minimization make sense: since the only required HRI activities are telepresence sessions, (i.e., $g^u_{min} = 0, \forall u \in U$), minimal distance solutions will not involve any bingo facilitations (and they do not). The results for the other objectives are, however, less encouraging. While both approaches are able to produce bingo participation for nearly all instances in the S1 category,[6] they perform rather poorly for larger problems. Specifically, for instance classes S3 and S4, no bingo games were facilitated in final solutions. For S2, the time-indexed MILP was only able to involve bingo games for a small fraction of instances, while the augmented graph model provided none. Upon reflection, this result makes sense: solutions involving bingo game HRIs require all of the synchronization and proximity side constraints to be satisfied, while those involving only telepresence HRIs do not. Additionally, since user bingo preference lower bounds are zero, a long sequence of the right branching decisions presumably must take place until the linear relaxation involves non-zero bingo facilitation/participation variables.

6.9.3.2 CP Approaches

The relaxed user travel results for the CP approaches, our proposed CP formulation and the previously proposed alternative resource model [199], are detailed in Table 6.9. The table follows the same structure

[6]The one instance that did not have bingo participation was actually bingo infeasible (i.e., could not support even a single bingo game).

	CP (Proposed)					**Alternative Resource CP [199]**				
	# Feas.	**# Best**	**# Opt.**	**TTF**	**Bingo**	**# Feas.**	**# Best**	**# Opt.**	**TTF**	**Bingo**
Fleet Distance (*minimization*)										
S1	19	17	0	137.1	2	**20**	**20**	0	0.1	0
S2	**20**	**20**	0	3.3	0	**20**	14	0	0.4	1
S3	**10**	**10**	0	74.2	0	**10**	8	0	396.0	1
S4	1	1	0	179.6	0	**2**	**2**	0	2.7	0
Bingo Participation (*maximization*)										
S1	**20**	**20**	**19**	0.2	**19**	**20**	18	9	0.4	17
S2	**19**	**18**	**3**	20.7	**19**	17	4	1	125.4	12
S3	7	7	0	31.0	7	**15**	**10**	0	277.4	**12**
S4	0	0	0	–	0	**7**	**7**	0	52.1	**2**
Complex Objective										
S1	**20**	17	0	1.0	**18**	**20**	**18**	0	0.1	17
S2	**17**	**15**	0	206.6	**16**	**14**	3	0	199.1	12
S3	6	**5**	0	248.5	6	**7**	**5**	0	1.4	**7**
S4	0	0	0	–	0	**2**	**2**	0	3.1	**2**

Table 6.9: SRRP-RH results. Relaxed user travel CP formulations. Bold values indicate the best result out of the two methods.

as Table 6.8.

Overall, the two CP models perform similarly on relaxed user travel problems, though the previously proposed model seems to scale slightly better. It would appear that our proposed CP approach outperforms the previously proposed approach for medium-sized (S2-S3) fleet distance minimization problems. For the other objective functions, bingo participation maximization and the complex objective function, the existing model maintains a slight edge for larger problems (i.e., S3 and S4). We note that, for the bingo participation objective on smaller problems (S1 and S2), the proposed formulation is able to prove optimality for considerably more instances (i.e., 19/20 (95%) vs. 9/20 (45%) for S1 and 3/20 (15%) vs. 0/20 (0%) for S2). We attribute this to the strengthening constraints added to the proposed CP model, which help the approach prove optimality through the improved pruning of domain values.

When compared to our MILP approach from Table 6.8, it is clear that the proposed MILP approach is able to find feasible solutions to more of the larger problems (S3 and S4). With respect to solutions that actually contain bingo game HRIs, however, both of the CP approaches outperform our proposed MILP, indicating that they are more suitable for participation-based objective functions (as opposed to routing). Additionally, the MILP approach is able to prove optimality for more S1 instances than either of the CP approaches. This is a fairly common finding; the use of the linear relaxation in MILP provides an effective means of proving optimality, whereas optimality is proven in CP through the complete pruning of domain stores, which can be more difficult for problems with large amounts of symmetry.

6.9.3.3 Search Manipulations

Following findings from previous work [199, 25], we explore the use of simple search manipulations in an effort to improve the baseline performance of the methods, without sacrificing their completeness. For the MILP models, we accomplish this through the use of *branching priorities* (BP), while in the CP

	MILP (Proposed) + BP					**Time-Indexed MILP [200] + BP**				
	# Feas.	**# Best**	**# Opt.**	**TTF**	**Bingo**	**# Feas.**	**# Best**	**# Opt.**	**TTF**	**Bingo**
Fleet Distance (*minimization*)										
S1	**20**	**20**	**18**	0.2	0	**20**	**20**	10	0.5	0
S2	$19^{(-1)}$	14	0	49.9	0	**20**	$\mathbf{18}^{(+1)}$	0	52.1	0
S3	$\mathbf{17}^{(+1)}$	**16**	0	378.5	0	3	$1^{(-1)}$	0	2010.0	0
S4	**2**	**2**	0	775.9	0	0	0	0	–	0
Bingo Participation (*maximization*)										
S1	**20**	**20**	**19**	0.7	**19**	**20**	**20**	**19**	0.5	**19**
S2	$\mathbf{20}^{(+3)}$	$\mathbf{19}^{(+4)}$	0	456.6	0	$17^{(-3)}$	$17^{(-3)}$	0	272.8	$\mathbf{1}^{(-1)}$
S3	$\mathbf{12}^{(-2)}$	$\mathbf{12}^{(-2)}$	0	160.2	0	2	2	0	2337.7	0
S4	0	0	0	–	0	0	0	0	–	0
Complex Objective										
S1	**20**	$\mathbf{20}^{(+1)}$	**6**	0.5	**19**	**20**	$10^{(-4)}$	1	0.5	18
S2	$17^{(-1)}$	11	0	290.8	$\mathbf{2}^{(-1)}$	$\mathbf{19}^{(-1)}$	**16**	0	123.0	$\mathbf{2}^{(-1)}$
S3	$\mathbf{11}^{(-1)}$	$\mathbf{11}^{(-1)}$	0	253.4	0	$3^{(+1)}$	$3^{(+1)}$	0	2166.7	0
S4	**3**	**3**	0	855.7	0	0	0	0	–	0

Table 6.10: SRRP-RH results. Relaxed user travel MILP formulations with branching priorities (BP) heuristic. Bold values indicate the best result out of the two methods. Superscript indicates performance difference against baseline version.

models use *variable ordering heuristics* (VOH) via search phases. Setting a non-zero branching priority for a set of variables within CPLEX ensures that the solver branches on those variables earlier in the search. Similarly, establishing a search phase over a set of variables in CP Optimizer ensures that the solver branches on those variables earlier.

Tran et al. [199] were successful in CP approaches to SRRP-RH using a problem decomposition approach that first solved the problem with a bingo game maximization objective function. Once a feasible solution was found, the variables participating in that solution (i.e., present interval variables) were used to construct a secondary optimization with the original complex objective function which was then solved. The authors noted that using this simplified objective function to identify high-potential variables had significant impact on their ability to produce solutions to reasonably sized instances. Our own work [25] found that the use of search phases over the robot bingo facilitation variables had significant impact on the performance of the CP model. Both of these works carry a consistent finding: prioritizing the variables associated with bingo games can have significant impact on CP model performance. Motivated by this, we investigate the use of similar search manipulations for our models.

Search Manipulation in MILP: Branching Priorities. For the MILP approaches, we manipulate the search using branching priorities. Following the finding that bingo game decisions are important to the success of the models, we set branching priorities over binary flow variables associated with robots faciliting bingo game HRIs. We note that, since continuous arrival time variables are not branched on in the search, it did not make sense to conduct branching priorities over them. For the previously proposed time-indexed MILP, we set:

$$\textsc{SetPriority}(x_{ra}) = 1, \forall r \in R, a \in G \tag{6.179}$$

	CP (Proposed) + VOH					**Alternative Resource CP [199] + VOH**				
	# Feas.	**# Best**	**# Opt.**	**TTF**	**Bingo**	**# Feas.**	**# Best**	**# Opt.**	**TTF**	**Bingo**
Fleet Distance (*minimization*)										
S1	**20**$^{(+1)}$	**20**$^{(+3)}$	0	0.1	0$^{(-2)}$	**20**	**20**	0	0.1	0
S2	19$^{(-1)}$	**19**$^{(-1)}$	0	0.4	0	**20**	15$^{(+1)}$	0	0.3	0$^{(-1)}$
S3	17$^{(+7)}$	**17**$^{(+7)}$	0	1.5	0	**19**$^{(+9)}$	11$^{(+3)}$	0	1.4	0$^{(-1)}$
S4	**14**$^{(+13)}$	**14**$^{(+13)}$	0	4.0	0	**14**$^{(+12)}$	13$^{(+11)}$	0	4.6	0
Bingo Participation (*maximization*)										
S1	**20**	**20**	**18**$^{(-1)}$	0.1	**18**$^{(-1)}$	**20**	18	8$^{(-1)}$	0.1	17
S2	**20**$^{(+1)}$	**19**$^{(+1)}$	**2**$^{(-1)}$	0.4	**20**$^{(+1)}$	**20**$^{(+3)}$	5$^{(+1)}$	0$^{(-1)}$	0.3	14$^{(+2)}$
S3	**19**$^{(+12)}$	**16**$^{(+9)}$	**1**$^{(+1)}$	1.4	**19**$^{(+12)}$	**19**$^{(+4)}$	4$^{(-6)}$	**1**$^{(+1)}$	1.4	16$^{(+4)}$
S4	**14**$^{(+14)}$	**14**$^{(+14)}$	0	3.1	**14**$^{(+14)}$	**14**$^{(+7)}$	1$^{(-6)}$	0	4.2	3$^{(+1)}$
Complex Objective										
S1	**20**	7$^{(-10)}$	0	0.1	**17**$^{(-1)}$	**20**	**17**$^{(-1)}$	0	0.1	**17**
S2	**20**$^{(+3)}$	**16**$^{(+1)}$	0	0.4	**20**$^{(+4)}$	**20**$^{(+6)}$	4$^{(-1)}$	0	0.4	14$^{(+2)}$
S3	**19**$^{(+13)}$	**14**$^{(+9)}$	0	1.4	**19**$^{(+13)}$	**19**$^{(+12)}$	5	0	1.6	15$^{(+8)}$
S4	**14**$^{(+14)}$	**10**$^{(+10)}$	0	3.4	**13**$^{(+13)}$	**14**$^{(+12)}$	4$^{(+2)}$	0	4.5	7$^{(+5)}$

Table 6.11: SRRP-RH results. Relaxed user travel CP formulations with variable ordering heuristic (VOH). Bold values indicate the best result out of the two methods. Superscript indicates performance difference against baseline version.

Eqn. (6.179) effectively ensures that the variables associated with robot bingo game HRI assignments are prioritized in the search (non-zero values are prioritized over those without set priorities). For our proposed augmented graph MILP model, we set priorites such that:

$$\textsc{SetPriority}(x_{ij}^{k}) = 1, \forall i \in \bar{A}^{a}, j \in \mathcal{V}^{k} \setminus \bar{A}^{a}, a \in G \tag{6.180}$$

where Eqn. (6.180) ensures that routing variables associated with robot bingo game HRIs are prioritized in the search. The results for the MILP models with these priorities are illustrated in Table 6.10.

MILP Search Manipulation Impact. In the table, values without a superscript indicate there was no performance difference (in terms of the displayed metrics) compared to the baseline MILP models. Values with superscripts indicate the number of instances for which an improvement (or degradation) was observed. It is fairly evident that the use of search priorities in MILP over the bingo game HRI variables has little effect on model performance. For the proposed MILP model, the use of priority branching maintains the same model performance as the baseline, or reduces it, on nearly all instance classes, with minor improvement for S2 instances with a bingo participation objective function. The existing time-indexed MILP model has primarily reduced performance, with some minor improvements on S3 instances with the complex objective function. The number of instances solved to optimality, for both methods, remains largely unchanged, and the time-to-first values for both methods do not change substantially. In general, we can conclude that search priorities have negligible impact on the MILP methods.

Search Manipulation in CP: Search Phases. We manipulate the CP search using search phases in CP Optimizer. The strategy here is a little different than in the MILP models, as start time decisions in the CP approaches are discrete valued and can be prioritized in the search. Specifically, we instantiate the following search phase for the previously proposed CP model:

$$0 \leq w_a^{start} \leq |T| - s_a \qquad \forall a \in G \tag{6.181}$$

$$\textsc{Start}(w_a) = w_a^{start} \qquad \forall a \in G \tag{6.182}$$

$$\textsc{SearchPhase}(\{w_a^{start} : a \in G\}) \tag{6.183}$$

Eqns. (6.181) through (6.183) introduce a new integer variable, w_a^{start}, for each bingo game HRI interval variable, w_a, and constrain the value of it to be the start time of the bingo HRI variable. A search phase is then placed over these bingo start time variables ensuring they are prioritized in the search.

Then, the search phase for our proposed CP model is constructed as follows:

$$0 \leq \bar{x}_i^r \leq |T| - s_a \qquad \forall i \in \bar{A}^a, a \in G, r \in R \tag{6.184}$$

$$\textsc{Start}(x_i^r) = \bar{x}_i^r \qquad \forall i \in \bar{A}^a, a \in G, r \in R \tag{6.185}$$

$$\textsc{SearchPhase}(\{\bar{x}_i^r : i \in \bar{A}^a, a \in G, r \in R\}) \tag{6.186}$$

where Eqns. (6.184) through (6.186) create and bind the integer variables, $\bar{x}_i^r$, to the bingo HRI interval variables and establish the search phase over them. We note that interval variables can still be set as absent during the search; for an absent interval variable implemented in CP Optimizer, $\textsc{Start}(var) = 0$, and the associated start time variable is constrained to a value of 0.

CP Search Manipulation Impact. The performance of the search manipulations for the CP approaches with relaxed user travel are presented in Table 6.11. While the addition of search phases has minor impact on the ability to prove optimality, they have a significant impact on the feasibility and quality of solutions. For both of the CP approaches the inclusion of search phases improves algorithm performance on nearly all problem classes. For the proposed CP approach, we see drastic improvements to the number of feasible solutions found and the number of solutions containing HRI activities for medium-to-large instances (S2 to S4). A similar trend is observed for the previously proposed CP approach as well. Notably, with search phases, both approaches are able to find feasible solutions to over half of the largest problem instances (those in S4), whereas the baseline implementation of these approaches struggled to find any. Furthermore, the time-to-first (TTF) values when using search phases are remarkably lower than those without. The ability to produce fast, feasible solutions to the problem is a highly desirable property in real-world robot applications, indicating that the use of search phases greatly improves the viability of these approaches for said applications.

6.9.3.4 Time-based Analysis: All Methods

Figure 6.7 provides detail for model performance throughout the course of the 1 hour runtime. The plots track the number of instances for which a given method has the best solution at various time points. From the figure, it is evident that the proposed CP approach with variable ordering heuristics outperforms all other methods across all objective functions. We can see that the previously proposed CP model with variable ordering exhibits similar performance for fleet distance minimization, however is outperformed significantly for the other objective functions. The rapid surge near the beginning of the runtime represents the various methods converging on the best solutions for the 20 small instances in S1. Overall, it would seem that the MILP approaches are the worst performing, with standalone CP

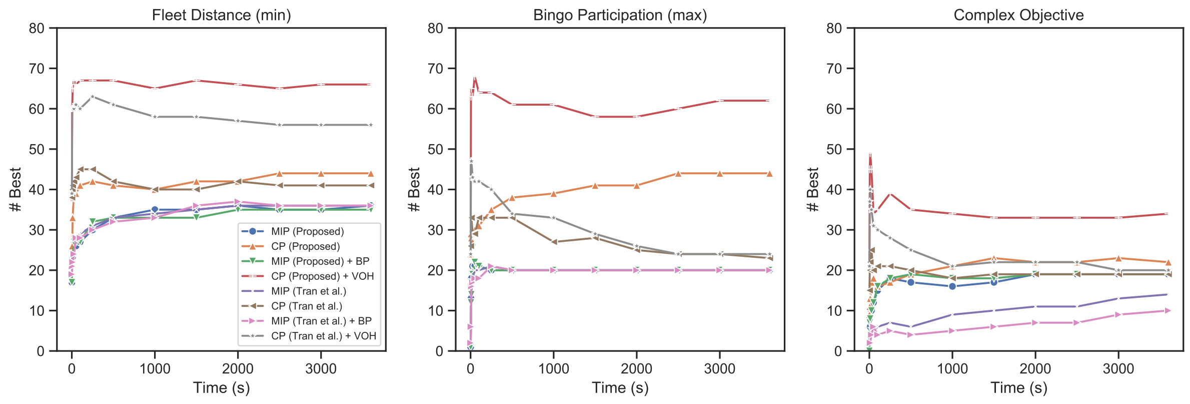


Figure 6.7: Time-based analysis, relaxed user travel. All methods.

providing middling results.

6.9.4 Experiments: Enforced User Travel

For these experiments, Constraints (6.104) and (6.178) are included as they were originally posed. Since the models in the literature do not capture user transition in the environment, we compare only our proposed MILP and CP approaches to each other, with and without search manipulation.

6.9.4.1 Homogeneous Robot Fleet

Table 6.12 summarizes the performance of the proposed methods (MILP, CP, and CP with search manipulation) with enforced user travel using the same homogeneous social robot instances as before. As search manipulation on the MILP model seemed to have little impact, and often degraded performance, we do not include those results in our assessment.

From the table we can see that the MILP model excels at small instances from S1, solving more instances to optimality than any other approach. For experiments in medium-to-large instance classes, however, the MILP approach begins to struggle with respect to finding good quality solutions and, for instances from S4, even finding feasible solutions. These findings are in-line with the findings for relaxed user travel experiments. Additionally, for the bingo participation and complex objective functions, we can see that the MILP approach is only rarely successful in incorporating bingo game HRIs into solutions for larger problems.

The proposed CP approach, standalone, is competitive with MILP for small instances, but lacks the ability to prove optimality for objective functions other than bingo participation maximization. The model, however demonstrates the ability to incorporate many more bingo game HRI activities into solutions than the MILP. An interesting observation is that standalone CP is able to find more feasible solutions to S4 instances with enforced user travel than it was with relaxed user travel. We suspect this is because the additional constraints enforcing user travel actually allow for stronger propagation within CP, allowing the solver to rule out areas of the search that it could not have previously.

The final approach detailed is CP with search manipulation (i.e., variable ordering heuristics). As with the user travel relaxed experiments, this approach is the strongest of the group with respect to feasibility and solution quality by a wide margin. CP with VOH finds the most feasible solutions on

	MILP (Proposed)					**CP (Proposed)**					**CP (Proposed) + VOH**				
	Feas.	**Best**	**Opt.**	**TTF**	**Bingo**	**Feas.**	**Best**	**Opt.**	**TTF**	**Bingo**	**Feas.**	**Best**	**Opt.**	**TTF**	**Bingo**
Fleet Distance (*minimization*)															
S1	**20**	**20**	**18**	0.3	0	**20**	19	0	45.0	1	**20**	**20**	0	0.1	0
S2	**20**	16	0	90.0	0	**20**	**19**	0	1.8	0	19	17	0	0.4	0
S3	17	5	0	261.7	0	10	8	0	208.6	0	**17**	**16**	0	1.6	0
S4	3	0	0	1147.0	0	2	2	0	3.4	0	**14**	**14**	0	4.8	0
Bingo Participation (*maximization*)															
S1	**20**	**20**	**19**	0.4	**18**	**20**	**20**	**19**	0.1	**18**	**20**	19	18	0.1	17
S2	**20**	0	0	941.0	0	**20**	**15**	**2**	3.5	**20**	**20**	13	**2**	0.4	**20**
S3	11	0	0	308.3	0	14	11	**1**	116.8	14	**19**	**15**	**1**	1.6	**19**
S4	1	0	0	2081.8	0	3	2	0	24.8	3	**14**	**13**	0	3.8	**14**
Complex Objective															
S1	**20**	**20**	**7**	0.4	**18**	**20**	15	0	0.1	17	**20**	17	0	0.1	17
S2	18	1	0	220.4	1	16	9	0	1.5	16	**20**	**11**	0	0.4	**17**
S3	16	0	0	521.5	0	6	5	0	1.3	6	**19**	**14**	0	1.6	**19**
S4	2	0	0	1395.3	0	2	1	0	3.2	2	**14**	**13**	0	5.0	**13**

Table 6.12: SRRP-RH results. Enforced user travel, homogeneous fleet. Proposed MILP and CP approaches. Bold values indicate the best result out of the three methods.

all problem classes with the exception of S2 under fleet distance minimization, where it finds one fewer than MILP. It provides the largest number of high-quality solutions in all of the larger problem instance classes (S3 and S4) across all objective functions, and provides solutions containing bingo game HRIs the most often for the latter two objective functions. Further, its average time to first solution (TTF) is considerably lower than the competing methods. Overall, the results indicate that the CP approach with search manipulation is the most suitable candidate for solving these problems, in agreement with the results for user travel relaxed experiments.

Time-based Analysis: All Methods Figure 6.8 provides a time-based analysis of the algorithms for user travel enforced problems. The figure, in the same fashion as Figure 6.7, provides an indication of the number of instances for which an approach has found the best solution at a given point in time. In agreement with the findings detailed for Table 6.12, the figure depicts the strong performance of CP with search manipulation, offering the best performance across all objective functions and at each measured point within the runtime. The MILP approach, while flat-lining quite early on bingo participation and complex objective functions, appears to be steadily improving for the fleet distance minimization objective.

Solution Degradation vs. Relaxed User Travel Figure 6.9 details the average degradation in solution quality seen when solving a given instance while enforcing user travel versus relaxing it. For each objective and each approach, the figure provides a summary of average objective value degradation (in %) for a given instance size. For a given instance, we calculate degradation as follows:

$$\%\text{ degradation } = \begin{cases} \frac{(sol^{*} - sol^{+})}{sol^{*}} \cdot 100, & \text{for bingo participation objective} \\ \frac{(sol^{+} - sol^{*})}{sol^{*}} \cdot 100, & \text{for complex objective} \end{cases}$$

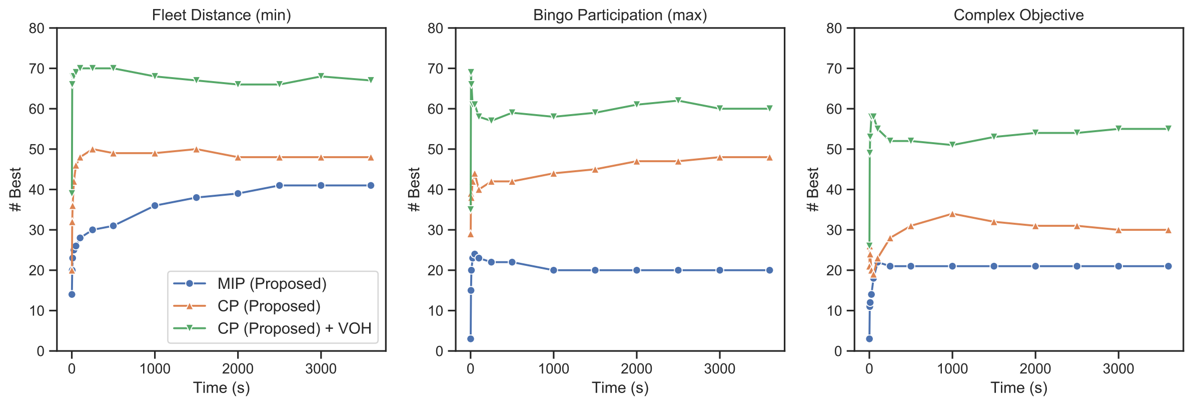


Figure 6.8: Experiment. Time-based analysis, enforced user travel with homogeneous fleets. Proposed methods.

where sol^* is the objective value for the travel relaxed formulation and sol^+ for the travel enforced formulation. Degradation is only calculated for instances that were found to be feasible for all formulations, and is represented by a bar with positive-valued height in the figure. We illustrate results for bingo game participation and the complex objective function only, as enforcing user travel was found to have negligible effect on the fleet distance objective function.

We expect solution quality to worsen when enforcing user travel constraints. The figure illustrates that degradation in solution quality is seen across the majority of objectives, algorithms and instance sizes, although with some exceptions. For smaller instances (S1 and S2), where final solutions are often the globally optimal value, it is evident that there is a significant degradation in solution quality when enforcing user travel constraints. The average degradation for the bingo participation objective is typically around 5%, while it can be in excess of 10% for the complex objective function. For the bingo participation objective, a 5% degradation for a solution that has a total user involvement in bingo games of 20 would indicate one less game played. These results for these smaller problems indicate that enforcing user travel can have a significant impact on the quality of produced schedules.

There are some examples of solution quality *improvement* when user travel constraints are enforced (i.e., negative degradation values), notably for larger bingo participation problem instances (S3 and S4) solved with the CP approaches. Since, for larger problems, the returned solutions are often far from the global optimum, it is entirely possible for the tree search with the more constrained formulation to discover better solutions by the end of the invoked run-time. As previously mentioned, these could potentially be cases where the addition of user travel constraints improves the propagation that the CP solver is able to implement.

6.9.4.2 Heterogeneous Robot Fleet

As a final experiment, we alter our benchmark set such that the social robot fleet is heterogeneous. As illustrated in Figure 6.10, social robot fleets are often heterogeneous in their ability to facilitate specific activities. In our recent work with a real-world implementation of our social robots [24], a fleet with two robot types $|K| = 2$ was used. The first robot type (center in the figure) was capable of facilitating all activity types, while the second robot type was only capable of facilitating reminder HRIs and telepresence sessions. For these experiments we adjust our benchmark set such that the first robot in

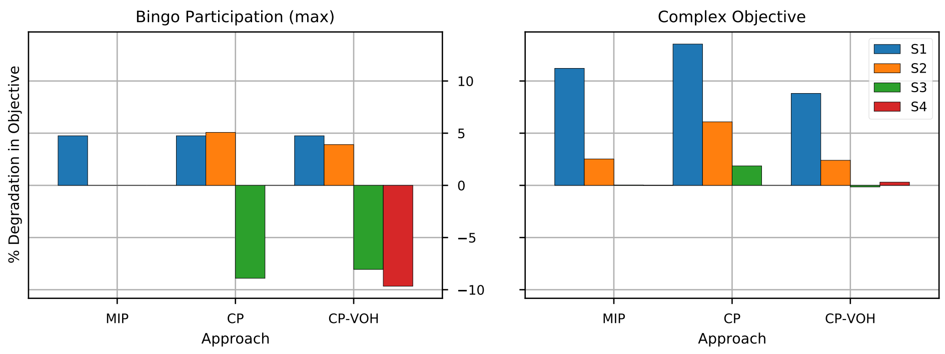


Figure 6.9: Solution degradation when enforcing user travel constraints (against relaxed user travel baseline). Average computed for instances where both relaxed and non-relaxed user travel experiments returned a feasible solution for a particular method.

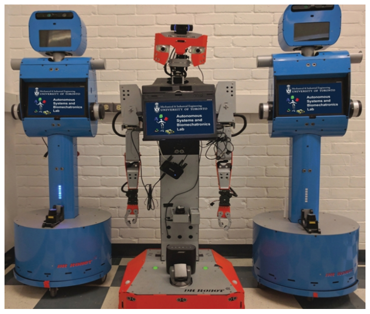


Figure 6.10: Heterogeneous social robot fleet [24]. Center robot capable of facilitating bingo game activities (and all other activities), while other robots only capable of facilitating reminder HRIs and telepresence sessions.

the fleet is of the first type, and all other robots are of the second type. For example, in the S1 problems, one robot would be of the first type and one of the second. The remainder of the parameters remain the same (i.e., all robot speeds, consumption rates and replenishment rates remain homogeneous).

Table 6.13 illustrates the results of our proposed approaches on these heterogeneous instances. The results, overall, follow a similar trend to the homogeneous experiments, with the CP approach with search manipulation providing the best overall results. The MILP approach offers strong performance for small problems (i.e., scenario S1), however, quickly struggles as instance size is increased. The CP approach without search manipulations provides better performance as instances are scaled, but is significantly outperformed by the CP approach with search manipulations.

Time-based Analysis: All Methods Figure 6.11 illustrates the overall performance of the proposed algorithms as the runtime progresses. The line-plot tracks the number of instances, at a given time point, for which the method has found the best solution (higher is better). We observe that the relative algorithm performance follows the homogeneous results with the CP approach using search manipulation performing the strongest by a considerable margin.

	MILP (Proposed)					CP (Proposed)					CP (Proposed) + VOH				
	Feas.	**Best**	**Opt.**	**TTF**	**Bingo**	**Feas.**	**Best**	**Opt.**	**TTF**	**Bingo**	**Feas.**	**Best**	**Opt.**	**TTF**	**Bingo**
Fleet Distance (*minimization*)															
S1	**20**	**20**	**18**	0.3	0	**20**	**20**	0	0.1	0	**20**	**20**	0	0.1	0
S2	**20**	**17**	0	91.1	0	9	9	0	0.3	0	19	16	0	0.3	0
S3	17	4	0	262.1	0	16	14	0	36.2	0	**19**	**16**	0	1.4	0
S4	3	0	0	1206.0	0	7	6	0	407.2	0	**14**	**13**	0	5.7	0
Bingo Participation (*maximization*)															
S1	**20**	**20**	**19**	0.4	18	**20**	19	18	0.1	17	**20**	19	18	0.1	17
S2	**20**	0	0	874.5	0	19	**14**	**1**	106.3	18	**20**	13	**1**	0.4	**20**
S3	11	0	0	295.3	0	15	11	**3**	9.3	15	**19**	**14**	0	1.4	**19**
S4	1	0	0	1978.8	0	7	7	0	18.1	7	**14**	**12**	0	3.5	**14**
Complex Objective															
S1	**20**	**20**	**7**	0.4	**18**	**20**	5	0	0.1	17	**20**	3	0	0.1	16
S2	18	0	0	203.4	0	16	5	0	50.1	14	**20**	**15**	0	0.4	**20**
S3	16	1	0	490.6	0	11	4	0	12.5	11	**19**	**15**	0	1.5	**18**
S4	2	0	0	1305.0	0	3	1	0	141.7	3	**14**	**13**	0	6.3	**13**

Table 6.13: SRRP-RH results. Enforced user travel, heterogeneous fleet. Proposed MILP and CP approaches. Bold values indicate the best result out of the three methods.

6.9.5 General Discussion

Given the empirical analyses performed, there are a number of observations that can be made. First, the proposed methods based on the augmented routing graph outperform previously proposed formulations for relaxed user travel problems, overall. Specifically, the proposed MILP model outperformed its time-indexed MILP counterpart with respect to the number of feasible solutions found, as well as the number of instances for which optimality was proven (across all objectives and problem sizes) as illustrated in Table 6.8. The proposed CP approach was competitive with the previously proposed CP approach when used standalone (Table 6.9), however, when search manipulations were added, the proposed formulation outperforms the existing model (Table 6.11). In general, the addition of search manipulation ensuring that decisions pertaining to robot facilitation of bingo HRIs are made early in the search seemed to have dramatic impact on solver performance, following trends noted in previous work [199, 25].

Second, the proper modeling of user transition in the environment resulted in a degradation in average solution quality for most scenarios and objectives (Figure 6.9). This indicates that seemingly high-quality solutions found for the travel relaxed variant, as the problem was originally posed, are not always found when user travel is enforced (particularly evident in the degradation shown for small problem instances). Such a finding supports the notion that properly modeling user travel in the environment is important when approaching the SRRP-RH. Furthermore, we were able to conduct heterogeneous robot fleet experiments that were more closely matched to the types of fleets used in real-world scenarios. Experiments suggested that relative algorithm performance remained the same, with the CP approach using search manipulations emerging as the most suitable technology by a substantial margin.

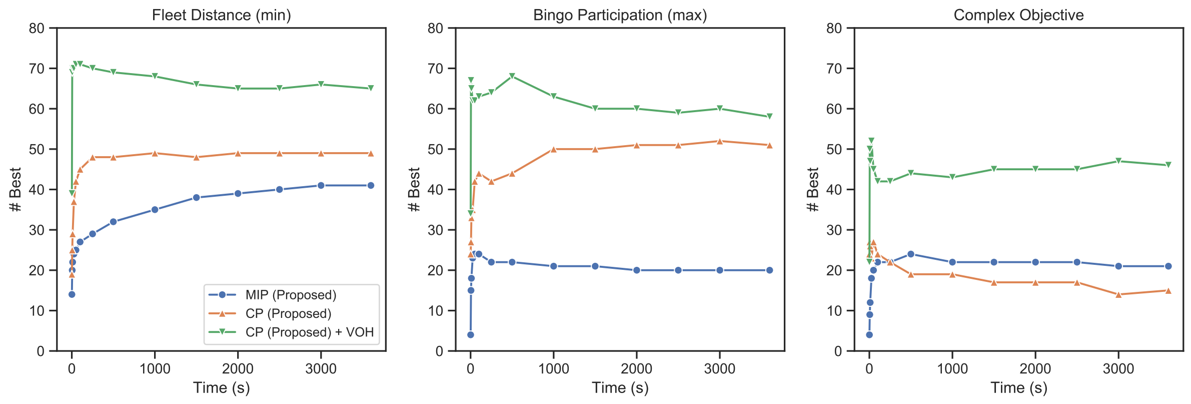


Figure 6.11: Time-based analysis, enforced user travel with heterogeneous fleets. Proposed methods.

6.9.5.1 Alternate Solution Methods

This work focused on the presentation of alternate strategies for approaching SRRP-RH in MILP and CP based on techniques from the electric vehicle routing literature. Indeed, the state-of-the-art for popular vehicle routing variants is often a decomposition of some kind, such as logic-based Benders decomposition (LBBD) [27] or, more frequently, branch-and-price [52].

The structure of the SRRP-RH, however, precludes the straightforward application of either algorithm. First, LBBD typically assumes, and performs well, when there is a clean decoupling of subproblems. The SRRP-RH, due to the cross-schedule dependency of bingo game facilitation on user reminder HRI participation, lacks this structure. Assuming the typical structure of an LBBD model for scheduling (master problem performing task allocation and subproblems performing task sequencing), the subproblems would need to be generated such that a given bingo game and all associated reminders are contained in the same subproblem. This strategy places a fair amount of the complexity on the subproblem. Furthermore, it was recently found that LBBD can be outperformed by standalone CP for specific vehicle routing problems [141].

Branch-and-price, on the other hand, works well when the variables in the routing formulation (columns) represent vehicle routes, and the cost of a route is independent of other routes. Similar to the discussion on LBBD, the cross-schedule dependency of the SRRP-RH means that the cost of a social robot route cannot be calculated without considering other robot routes. A potential solution to this would be to pack even more complexity into the master problem variable, however this has implications on the efficiency of the subproblems, which, in the case of the SRRP-RH, are likely not able to be modeled as resource-constrained shortest path problems.

6.9.5.2 Impact of Assumptions

As was mentioned earlier in the chapter, there were a number of assumptions made when modeling the SRRP-RH such that progress could be made on key challenges. We assumed that the social robots move around the environment unhindered and with deterministic travel times. In real-world use cases, the presence of other humans and robots moving in the environment will likely cause travel time estimates to vary. Indeed, the study of robot task planning in human-populated environments is a growing area [158, 194]. A potential solution to this would be to enforce more conservative robot speeds allowing the fleet more time to get to locations, however, the possibility for schedule failure still remains.

We also assumed linear energy recharging and consumption rates. Real-world robot batteries do not typically follow linear recharge patterns and more sophisticated modeling could attempt to capture these trajectories, following recent work in electric vehicle routing [159]. Once again, conservative estimates can be built into the parameters as a temporary solution. This approach will further erode the solution quality of produced routes, however, as the robot fleet spends more time at each of the recharge stations.

In general, our assumptions regarding static parameters are likely to cause issues in real-world implementations. We have conducted initial efforts to integrate an early version of our CP model, along with an early version of a person search module [158], into a real-world use case with promising levels of success [24]. This study determined that the highest failure rates were typically attributed to lower-level tasks such as user identification, a task that has been completely abstracted away at the level of the SRRP-RH. While the value of high-level route planning may be questionable given the uncertainty inherent in lower level robot function, the reliability of these complex lower level systems is constantly improving. Furthermore, the successful implementation of online re-planning/re-routing systems is a promising area of research that can bolster the utility of such static techniques [97].

6.10 Conclusions

In this chapter we presented the SRRP-RH, a problem that involves routing a team of social robots in a retirement home environment to facilitate cognitively stimulating HRI activities with residents. Given the strong connection between electric vehicle routing and multi-robot task allocation, we classified the SRRP-RH according to taxonomies from both the MRTA and VRP literatures and provided details on the complexity of the fleet distance minimization variant of the problem.

As a means to approach the SRRP-RH modeling challenge, we propose novel MILP and CP models based on augmented graph modeling techniques from the electric vehicle routing literature. These models include consideration for user travel in the environment. Our MILP model leverages vehicle symmetry for robots within a given type via a compact formulation. We compare our approaches to those that have been previously published on a user travel relaxed variant of SRRP-RH, as it was initially modeled, demonstrating, through empirical assessment, that our methods are competitive with, and often outperform, existing approaches to the problem, with respect to solution feasibility and quality for a variety of objective functions.

Motivated by previous findings regarding the importance of making bingo game HRI decisions early in the search, we improve the performance of our CP approaches, and the previously proposed CP model, with variable ordering heuristics for the SRRP-RH. We also investigate similar priority branching heuristics in MILP, motivated by the success observed with the CP models, however, found these branching strategies to be ineffective. We demonstrate, via modeling and experimental analysis, that our proposed approaches are able to capture user travel within the environment, an important aspect of SRRP-RH that ensures produced routes are achievable in practice.

The work in this chapter supports our thesis, demonstrating that the adaptation of approaches to electric vehicle routing problems can yield a high performance and flexible CP approach for this complex, real-world problem.

Chapter 7

Concluding Remarks

In this dissertation, we investigate our thesis that constraint programming (CP) can be an effective and flexible approach for modeling and solving routing problems involving electric vehicles. Informing our approach with modeling strategies from mixed-integer linear programming (MILP), we formulate CP models that provide strong performance on this important and increasingly pervasive class of problems.

In Chapter 4, we investigated the electric vehicle routing problem with time windows (EVRPTW) and the pickup and delivery problem with time windows and electric vehicles (PDPTW-EV). We proposed novel CP models for these problems based on recharging path multigraphs and a single resource transformation that exploits the symmetry in homogeneous vehicle fleets. The latter can be applied to other routing and scheduling problems and results in models that use significantly less memory. For models based on the augmented routing graph, we empirically demonstrated that the single resource transformation modeling technique significantly outperforms the alternative resource CP model for both the EVRPTW and the PDPTW-EV. We also demonstrate that our single resource CP model significantly outperforms a MILP model from the literature on medium-to-large size problems across a variety of objective functions. For models based on recharging path multigraphs, our CP approaches exhibit weaker overall performance relative to their MILP counterparts, however, offer advantages for fleet size minimization problems.

In Chapter 5, we explore our thesis in the context of a range-constrained variant of the UAV target search problem, where we assess the viability of commercially available UAVs for target search missions over large-scale road networks. In this work, we augment the UAV fleet with ground-based mobile recharging vehicles (MRVs) that can meet up with and recharge UAVs via a constant-time battery swap operation, effectively increasing their operational range. To formulate our optimization models, we first propose a novel pipeline for taking road network data and, through a series of discretizations, producing an augmented routing graph. Then, we propose MILP and CP formlations for solving the problem based on the augmented routing graph strategies presented in Chapter 4, with some alterations for heterogeneous fleets. Our empirical results indicate the superiority of CP for this problem over MILP, and support the inclusion of MRVs in the fleet which can significantly boost the accumulated search reward of the UAV fleet.

In Chapter 6, we introduced the social robot routing problem in retirement homes (SRRP-RH). The problem involves routing a team of social robots to facilitate a variety of HRI activities with retirement home residents. In contrast to previous chapters, this problem includes a variety of complex side constraints including those involving user availability calendars, robot-user synchronization for activities, and optional activity precedence. Following our research approach, we develop novel MILP and CP formulations for the SRRP-RH based on an augmented routing graph representation, following techniques adopted in the preceding chapters. We compare these formulations to those that have been previously published on a user travel relaxed variant of the SRRP-RH, and conduct new experiments on a variant of the problem that captures user travel within the environment. We show that our methods are competitive with, and often outperform, existing models for this problem. We also demonstrate that a relatively simple variable ordering scheme for our CP formulation can bolster its performance, increasing the size of problems that can be solved and significantly outperforming the other methods.

7.1 Summary of Contributions

In this section we reiterate the primary contributions of this dissertation. Whereas in the introduction we organized each of the contributions by the chapter in which they appear, here we summarize our primary methodological and application-based contributions to the field of CP followed by those to other areas.

Contributions to Constraint Programming

1. We propose a new CP modeling technique for fuel-constrained routing formulations using interval and sequence variables based on multigraphs, where multiple arcs between graph vertices represent different recharging/refueling path choices. Our technique uses a Table constraint, in addition to fuel/energy tracking constraints and variables, to model the problem. (Chapter 4)

2. We propose a novel single resource transformation for modeling homogeneous routing problems with interval, sequence, and cumulative function expression variables. Following the two-index modeling strategy in MILP, this modeling technique exploits vehicle symmetry to significantly reduce the number of model variables and constraints. We describe a simple enhancement to this modeling strategy based on the use of optional horizon segments and a variable ordering heuristic to improve its performance. (Chapter 4)

3. We extend our single resource transformation to heterogeneous vehicle fleets where an augmented horizon is used for each distinct vehicle type. We detail the implementation of synchronization constraints for this formulation using constraints involving the Mod operator. For completely heterogeneous fleets, the formulation follows the alternative resource technique. (Chapter 5)

4. We present new CP models for a variety of problem domains, namely the EVRPTW and the PDPTW-EV (Chapter 4), a range-constrained joint UAV/MRV target search problem with mobile recharging (Chapter 5), and the SRRP-RH (Chapter 6). These models leverage the novel strategies described above, where appropriate.

5. Throughout this dissertation, we conduct a series of extensive empirical analyses, comparing our proposed CP models to MILP formulations for each of the problem domains. Our results support our thesis, indicating the effectiveness and flexibility of CP for solving electric vehicle routing problems with performance often superior to those of the MILP approaches.

Other Contributions

1. We propose a novel algorithm for constructing a target search routing graph from real-world road network data, over which our optimization models are formulated. The algorithm consists of a series of discretization steps, including a shortest-path sampling of the underlying road network and the evaluation of a hitting set problem to determine the placement of UAV/MRV recharge rendezvous opportunities. (Chapter 5)

2. We classify the SRRP-RH according to taxonomies from both the multi-robot task allocation (MRTA) and vehicle routing problem literature and discuss the complexity of the fleet distance minimization variant of the problem. This work represents, to the best of our knowledge, the first link between the fields of electric vehicle routing and MRTA. (Chapter 6)

3. We present new MILP models for a range-constrained joint UAV/MRV target search problem with mobile recharging (Chapter 5), and the SRRP-RH (Chapter 6). These formulations follow and adapt modeling strategies from the MILP literature for electric vehicle routing problems as detailed in Chapter 3. In the case of the SRRP-RH, our augmented routing graph MILP formulation provides significantly better scaling than a time-index model from the literature.

7.2 Future Work

In this section, we highlight future research opportunities that materialized as a direct consequence of our investigations. These opportunities fall into two categories: the exploration of alternate problem characteristics, and modeling extensions and manipulations.

7.2.1 Alternate Problem Characteristics

A natural avenue of future work is to investigate alternate problem definitions. In this section, we discuss two possible areas for such investigation.

Non-Linear Recharging and Consumption. Following many of the works in electric vehicle routing, all of the models in this dissertation assumed linear vehicle recharging and linear vehicle energy consumption profiles. In practice, however, more complex recharging and energy consumption profiles could be used to more accurately model these processes [11].

As discussed in Chapter 3, the work of Montoya et al. [159] and Froger et al. [76] explored the development of MILP formulations for problems with non-linear battery recharging. Instead of strictly linear recharging profile, batteries tend to exhibit a linear recharge rate for the first 80% of battery capacity, followed by a tapering off in recharge rate [159], as illustrated in Figure 7.1. Referring to the figure, it is evident that more realistic recharging profiles result in longer recharging times, and different recharge rates depending on the vehicle's state of charge when it initiates recharging. Higher voltages can be used to increase the rate of recharge, but the non-linear behavior remains. Therefore, an interesting avenue of future research is to incorporate non-linear recharging profiles into our CP formulations. While MILP modeling approaches typically address this characteristic using piece-wise linear approximations, the more flexible modeling capabilities of CP could potentially be used to model smooth recharge profiles following low-order polynomials, for example.

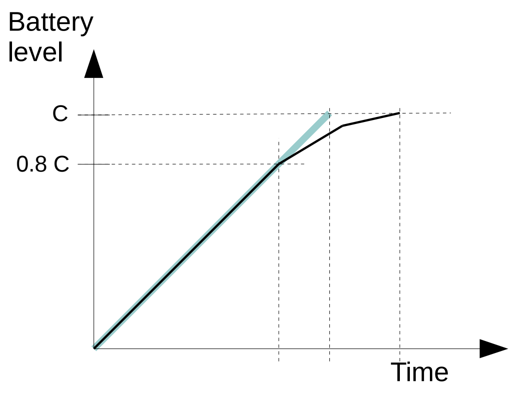


Figure 7.1: Example non-linear electric vehicle recharging profile. C indicates battery capacity. Blue shaded line indicates linear recharge assumption.

The work of Barco et al. [11] explored non-linear energy consumption profiles of electric vehicles driven by the underlying physics. In their work, the authors note that energy consumption on the road depends on the environment as well as the road and vehicle characteristics. They propose an energy consumption model based on the longitudinal dynamics equations of motion that expresses energy consumption as a function of different factors. In this dissertation, we assumed linear energy consumption profiles. Extending our modeling efforts to incorporate the energy consumption models of Barco et al. is an interesting area for future research.

Mobile Recharging On the Move. In Chapter 5, we presented MILP and CP formulations for a joint UAV/MRV fleet routing problem involving a surveillance application. In our formulation, an MRV can meet up with and recharge a UAV via a fixed-location battery swap operation provided the two vehicles are synchronized in time and space. As discussed in the chapter, related work on a similar problem permits the UAV to be loaded on the MRV and initiate recharging while the MRV is traveling on the road network [212]. Given that battery swap-outs are fairly quick procedures, it is unclear how this added capability would impact routes in the context of our surveillance application, but it could be an interesting avenue to explore with CP in the context of other UAV applications.

7.2.2 Modeling Extensions and Manipulations

Another avenue of future work is to explore further extensions to the novel CP modeling techniques we introduced in this dissertation.

Single Resource Transformation. In this dissertation we introduced a novel single resource transformation for routing problems modeled with interval and sequence variables, and involving homogeneous (or partially homogeneous) fleets. A potentially high-impact area of future work is the application of this modeling technique to scheduling problems with homogeneous (or partially homogeneous) machines.

While applying the single resource technique to scheduling problems may seem straightforward, the nature of common scheduling objective functions may be problematic for solver performance using this technique due to solver tendency to 'left-shift' tasks. In this dissertation, we applied the single resource transformation to problems with traditional routing objective functions (e.g., fleet distance

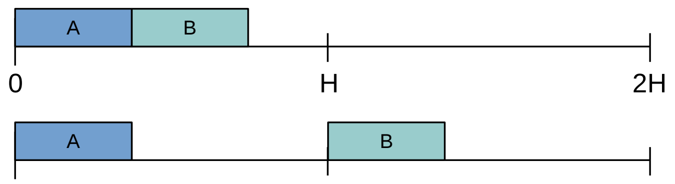


Figure 7.2: Single resource transformation for scheduling problems. Impact of task start time on makespan minimization objective function. Tasks denoted by A and B. Scheduling horizon denoted by H.

minimization, orienteering-style reward maximization). For these objectives, the position of tasks/visits along the augmented horizons does not, intrinsically, erode the objective function value. For scheduling objective functions such as makespan or sum of completion times minimization, left-shifting tasks on augmented horizons may be detrimental to performance. For example, consider the schedules presented in Figure 7.2. Under a makespan minimization objective, the top left-shifted schedule will produce a makespan of $c_B = p_A + p_B$, where c_i is the completion time of task i and p_i the processing time of task i. The bottom schedule, however, will have a makespan of $\max(c_A, c_B \mod H)$. With $c_A = c_B = 1$, the top schedule has a makespan twice that of the bottom.

Under a 'min-value' heuristic, the solver may take considerable time to set the start time of task B to be equal to H. Thus, an interesting avenue of future work would be the investigation of value ordering heuristics that could better-exploit the structure of the single resource transformation's augmented horizons for scheduling problems. This approach could prioritize branching on start times close to each horizon start point, as opposed to simply prioritizing the smallest start times.

Recharging Path Multigraph Formulation. In this dissertation we introduced CP formulations for electric routing problems based on recharging path multigraphs. As discussed in Chapter 4, these formulations require a pre-processing step that identifies energy feasible routes between pairs of customer vertices, and forms a multigraph with these routes. The work of Froger et al. [74] noted that, for problems with capacitated recharging stations, the dominance rules weaken during the generation of the multigraph, resulting in the identification of fewer infeasible paths and a denser multigraph. As a result, they showed that these models for capacitated problems are not necessarily advantageous over augmented routing graph models. A potential avenue of future work is to apply our multigraph-based CP formulations to capacitated recharging station problems.

Bibliography

[1] Tobias Achterberg. SCIP: solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[2] Tobias Achterberg, Thorsten Koch, and Alexander Martin. Branching rules revisited. *Operations Research Letters*, 33(1):42–54, 2005.

[3] Anagnostopoulou Afroditi, Maria Boile, Sotirios Theofanis, Eleftherios Sdoukopoulos, and Dimitrios Margaritis. Electric vehicle routing problem with industry constraints: trends and insights for future research. *Transportation Research Procedia*, 3:452–459, 2014.

[4] Juho Andelmin and Enrico Bartolini. An exact algorithm for the green vehicle routing problem. *Transportation Science*, 51(4):1288–1303, 2017.

[5] David L Applegate, Robert E Bixby, Vasek Chvatal, and William J Cook. *The traveling salesman problem: a computational study.* Princeton university press, 2006.

[6] Tamio Arai, Enrico Pagello, Lynne E Parker, et al. Advances in multi-robot systems. *IEEE Transactions on robotics and automation*, 18(5):655–661, 2002.

[7] Rajesh Arumugam, Vikas Reddy Enti, Liu Bingbing, Wu Xiaojun, Krishnamoorthy Baskaran, Foong Foo Kong, A Senthil Kumar, Kang Dee Meng, and Goh Wai Kit. Davinci: A cloud computing framework for service robots. In *2010 IEEE international conference on robotics and automation*, pages 3084–3089. IEEE, 2010.

[8] Michel L Balinski and Richard E Quandt. On an integer program for a delivery problem. *Operations research*, 12(2):300–304, 1964.

[9] Marian R Banks, Lisa M Willoughby, and William A Banks. Animal-assisted therapy and loneliness in nursing homes: use of robotic versus living dogs. *Journal of the American Medical Directors Association*, 9(3):173–177, 2008.

[10] Philippe Baptiste, Claude Le Pape, and Wim Nuijten. *Constraint-based scheduling: applying constraint programming to scheduling problems*, volume 39. Springer Science & Business Media, 2012.

[11] John Barco, Andres Guerra, Luis Munoz, and Nicanor Quijano. Optimal routing and scheduling of charge for electric vehicles: A case study. *Mathematical Problems in Engineering*, 2017.

[12] Rafael Basso, Balázs Kulcsár, Bo Egardt, Peter Lindroth, and Ivan Sanchez-Diaz. Energy consumption estimation integrated into the electric vehicle routing problem. *Transportation Research Part D: Transport and Environment*, 69:141–167, 2019.

[13] J Christopher Beck and Mark S Fox. Constraint-directed techniques for scheduling alternative activities. *Artificial Intelligence*, 121(1-2):211–250, 2000.

[14] J Christopher Beck, Patrick Prosser, and Evgeny Selensky. Vehicle routing and job shop scheduling: What's the difference? In *International Conference on Automated Planning and Scheduling*, pages 267–276, 2003.

[15] Nicolas Beldiceanu, Mats Carlsson, Sophie Demassey, and Thierry Petit. Global constraint catalogue: Past, present and future. *Constraints*, 12(1):21–62, 2007.

[16] Roger Bemelmans, Gert Jan Gelderblom, Pieter Jonker, and Luc De Witte. Socially assistive robots in elderly care: A systematic review into effects and effectiveness. *Journal of the American Medical Directors Association*, 13(2):114–120, 2012.

[17] Gerardo Berbeglia, Jean-François Cordeau, and Gilbert Laporte. A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem. *INFORMS Journal on Computing*, 24(3):343–355, 2012.

[18] Gerardo Berbeglia, Gilles Pesant, and Louis-Martin Rousseau. Checking the feasibility of dial-a-ride instances using constraint programming. *Transportation Science*, 45(3):399–412, 2011.

[19] Sara Bernardini, Maria Fox, and Derek Long. Combining temporal planning with probabilistic reasoning for autonomous surveillance missions. *Autonomous Robots*, 41(1):181–203, 2017.

[20] Sara Bernardini, Maria Fox, Derek Long, and Chiara Piacentini. Deterministic versus probabilistic methods for searching for an evasive target. In *AAAI*, pages 3709–3715, 2017.

[21] Brett Bethke, Mario Valenti, and Jonathan P How. UAV task assignment. *IEEE robotics & automation magazine*, 15(1):39–44, 2008.

[22] Robert E Bixby. A brief history of linear and mixed-integer programming computation. *Documenta Mathematica*, pages 107–121, 2012.

[23] Kyle E. C. Booth and J Christopher Beck. A constraint programming approach to electric vehicle routing with time windows. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 129–145. Springer, 2019.

[24] Kyle E. C. Booth, Sharaf C Mohamed, Sanjif Rajaratnam, Goldie Nejat, and J Christopher Beck. Robots in retirement homes: Person search and task planning for a group of residents by a team of assistive robots. *IEEE Intelligent Systems*, 32(6):14–21, 2017.

[25] Kyle E. C. Booth, Goldie Nejat, and J Christopher Beck. A constraint programming approach to multi-robot task allocation and scheduling in retirement homes. In *International Conference on Principles and Practice of Constraint Programming*, pages 539–555. Springer, 2016.

[26] Kyle E. C. Booth, Chiara Piacentini, Sara Bernardini, and J. Christopher Beck. Target search on road networks with range-constrained UAVs and ground-based mobile recharging vehicles. *IEEE Robotics and Automation Letters*, 5(4):6702–6709, 2020.

[27] Kyle E. C. Booth, Tony T Tran, and J Christopher Beck. Logic-based decomposition methods for the travelling purchaser problem. In *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 55–64. Springer, 2016.

[28] Kyle E. C. Booth, Tony T Tran, Goldie Nejat, and J Christopher Beck. Mixed-integer and constraint programming techniques for mobile robot task planning. *IEEE Robotics and Automation Letters*, 1(1):500–507, 2016.

[29] Cynthia Breazeal. Toward sociable robots. *Robotics and autonomous systems*, 42(3-4):167–175, 2003.

[30] Elizabeth Broadbent, Rie Tamagawa, Ngaire Kerse, Brett Knock, Anna Patience, and Bruce MacDonald. Retirement home staff and residents' preferences for healthcare robots. In *Robot and human interactive communication, 2009. RO-MAN 2009. The 18th IEEE international symposium on*, pages 645–650. IEEE, 2009.

[31] Joost Broekens, Marcel Heerink, and Henk Rosendal. Assistive social robots in elderly care: a review. *Gerontechnology*, 8(2):94–103, 2009.

[32] Maurizio Bruglieri, Simona Mancini, Ferdinando Pezzella, and Ornella Pisacane. A new mathematical programming model for the green vehicle routing problem. *Electronic Notes in Discrete Mathematics*, 55:89–92, 2016.

[33] Maurizio Bruglieri, Simona Mancini, Ferdinando Pezzella, Ornella Pisacane, and Stefano Suraci. A three-phase matheuristic for the time-effective electric vehicle routing problem with partial recharges. *Electronic Notes in Discrete Mathematics*, 58:95–102, 2017.

[34] Maurizio Bruglieri, Ferdinando Pezzella, Ornella Pisacane, and Stefano Suraci. A variable neighborhood search branching for the electric vehicle routing problem with time windows. *Electronic Notes in Discrete Mathematics*, 47:221–228, 2015.

[35] Quentin Cappart, Charles Thomas, Pierre Schaus, and Louis-Martin Rousseau. A constraint programming approach for solving patient transportation problems. In *International Conference on Principles and Practice of Constraint Programming*, pages 490–506. Springer, 2018.

[36] Darshan Chauhan, Avinash Unnikrishnan, and Miguel Figliozzi. Maximum coverage capacitated facility location problem with range constrained drones. *Transportation Research Part C: Emerging Technologies*, 99:1–18, 2019.

[37] Andre Cire, Elvin Coban, and John N Hooker. Mixed integer programming vs. logic-based Benders decomposition for planning and scheduling. In *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, pages 325–331. Springer, 2013.

[38] Geoff Clarke and John W Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, 12(4):568–581, 1964.

[39] Ryan G Conrad and Miguel Andres Figliozzi. The recharging vehicle routing problem. In *Proceedings of the 2011 industrial engineering research conference*, page 8. IISE Norcross, GA, 2011.

[40] Cristián E Cortés, Michel Gendreau, Louis Martin Rousseau, Sebastián Souyris, and Andrés Weintraub. Branch-and-price and constraint programming for solving a real-life technician dispatching problem. *European Journal of Operational Research*, 238(1):300–312, 2014.

[41] Rosario Cuda, Gianfranco Guastaroba, and Maria Grazia Speranza. A survey on two-echelon routing problems. *Computers & Operations Research*, 55:185–199, 2015.

[42] Giacomo Da Col and Erich C Teppan. Industrial size job shop scheduling tackled by present day CP solvers. In *International Conference on Principles and Practice of Constraint Programming*, pages 144–160. Springer, 2019.

[43] George Dantzig, Ray Fulkerson, and Selmer Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the operations research society of America*, 2(4):393–410, 1954.

[44] George B Dantzig and John H Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

[45] George Bernard Dantzig. *Linear programming and extensions*, volume 48. Princeton university press, 1998.

[46] George Bernard Dantzig, Delbert R Fulkerson, and Selmer Martin Johnson. On a linear-programming, combinatorial approach to the traveling-salesman problem. *Operations Research*, 7(1):58–66, 1959.

[47] Danial Davarnia and John N Hooker. Consistency for 0–1 programming. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 225–240. Springer, 2019.

[48] Bruno De Backer, Vincent Furnon, P Prosser, P Kilby, and Paul Shaw. Local search in constraint programming: Application to the vehicle routing problem. In *Proc. CP-97 Workshop Indust. Constraint-Directed Scheduling*, pages 1–15. Schloss Hagenberg Austria, 1997.

[49] Bruno De Backer, Vincent Furnon, Paul Shaw, Philip Kilby, and Patrick Prosser. Solving vehicle routing problems using constraint programming and metaheuristics. *Journal of Heuristics*, 6(4):501–523, 2000.

[50] Alessandro E De Luca, S Bonacci, and G Giraldi. Aging populations: the health and quality of life of the elderly. *La Clinica Terapeutica*, 162(1):e13–8, 2010.

[51] Rommert Dekker, Jacqueline Bloemhof, and Ioannis Mallidis. Operations research for green logistics–an overview of aspects, issues, contributions and challenges. *European Journal of Operational Research*, 219(3):671–679, 2012.

[52] Guy Desaulniers, Fausto Errico, Stefan Irnich, and Michael Schneider. Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research*, 64(6):1388–1405, 2016.

[53] Martin Desrochers, Jan Karel Lenstra, Martin WP Savelsbergh, and François Soumis. Vehicle routing with time windows: optimization and approximation. Technical report, CWI. Department of Operations Research and System Theory [BS], 1987.

[54] Jacques Desrosiers, François Soumis, and Martin Desrochers. Routing with time windows by column generation. *Networks*, 14(4):545–565, 1984.

[55] Luca Di Gaspero, Andrea Rendl, and Tommaso Urli. Balancing bike sharing systems with constraint programming. *Constraints*, 21(2):318–348, 2016.

[56] US Doe. National algal biofuels technology roadmap. *US Department of Energy, Office of Energy Efficiency and Renewable Energy, Biomass Program*, 2010.

[57] Kevin Dorling, Jordan Heinrichs, Geoffrey G Messier, and Sebastian Magierowski. Vehicle routing problems for drone delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1):70–85, 2016.

[58] Michael Drexl. Synchronization in vehicle routing-a survey of VRPs with multiple synchronization constraints. *Transportation Science*, 46(3):297–316, 2012.

[59] Tomislav Erdelić and Tonči Carić. A survey on the electric vehicle routing problem: variants and solution approaches. *Journal of Advanced Transportation*, 2019, 2019.

[60] Sevgi Erdoğan and Elise Miller-Hooks. A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100–114, 2012.

[61] David Feil-Seifer and Maja J Matarić. Socially assistive robotics. *IEEE Robotics & Automation Magazine*, 18(1):24–31, 2011.

[62] Dominique Feillet. A tutorial on column generation and branch-and-price for vehicle routing problems. *4or*, 8(4):407–424, 2010.

[63] Ángel Felipe, M Teresa Ortuño, Giovanni Righini, and Gregorio Tirado. A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transportation Research Part E: Logistics and Transportation Review*, 71:111–128, 2014.

[64] Wei Feng and Miguel Figliozzi. An economic and technological analysis of the key factors affecting the competitiveness of electric commercial vehicles: A case study from the usa market. *Transportation Research Part C: Emerging Technologies*, 26:135–145, 2013.

[65] Matteo Fischetti and Andrea Lodi. Local branching. *Mathematical programming*, 98(1-3):23–47, 2003.

[66] Matteo Fischetti, Juan José Salazar González, and Paolo Toth. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research*, 45(3):378–394, 1997.

[67] Marshall L Fisher and Ramchandran Jaikumar. A generalized assignment heuristic for vehicle routing. *Networks*, 11(2):109–124, 1981.

[68] Christodoulos A Floudas and Xiaoxia Lin. Mixed integer linear programming in process scheduling: Modeling, algorithms, and applications. *Annals of Operations Research*, 139(1):131–162, 2005.

[69] John Forrest and Robin Lougee-Heimer. CBC user guide. In *Emerging theory, methods, and applications*, pages 257–277. INFORMS, 2005.

[70] Robert Fourer, David M Gay, and Brian W Kernighan. AMPL. A modeling language for mathematical programming. 1993.

[71] Colombo Francesca, Llena-Nozal Ana, Mercier Jérôme, and Tjadens Frits. *OECD Health Policy Studies Help Wanted? Providing and Paying for Long-Term Care: Providing and Paying for Long-Term Care*, volume 2011. OECD Publishing, 2011.

[72] Kathryn Glenn Francis and Peter J Stuckey. Explaining circuit propagation. *Constraints*, 19(1):1–29, 2014.

[73] Aurélien Froger, Jorge E Mendoza, Ola Jabali, and Gilbert Laporte. The electric vehicle routing problem with capacitated charging stations. *Technical report*. `hal.archives-ouvertes.fr/hal-02386167`.

[74] Aurélien Froger, Jorge E Mendoza, Ola Jabali, and Gilbert Laporte. A matheuristic for the electric vehicle routing problem with capacitated charging stations. *Technical report*, 2017. `hal.archives-ouvertes.fr/hal-01559524`.

[75] Aurélien Froger, Jorge E Mendoza, Ola Jabali, and Gilbert Laporte. Modeling and solving the electric vehicle routing problem with nonlinear charging functions and capacitated charging stations. *Technical report*, 2018. `hal.archives-ouvertes.fr/hal-01814645`.

[76] Aurélien Froger, Jorge E Mendoza, Ola Jabali, and Gilbert Laporte. Improved formulations and algorithmic components for the electric vehicle routing problem with nonlinear charging functions. *Computers & Operations Research*, 104:256–294, 2019.

[77] Ridvan Gedik, Emre Kirac, Ashlea Bennet Milburn, and Chase Rainwater. A constraint programming approach for the team orienteering problem with time windows. *Computers & Industrial Engineering*, 107:178–195, 2017.

[78] Bernard Gendron, Paul-Virak Khuong, and Frédéric Semet. Comparison of formulations for the two-level uncapacitated facility location problem with single assignment constraints. *Computers & Operations Research*, 86:86–93, 2017.

[79] Brian P Gerkey and Maja J Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954, 2004.

[80] Carmen Gervet. Constraints over structured domains. In *Foundations of Artificial Intelligence*, volume 2, pages 605–638. Elsevier, 2006.

[81] Thomas J Gleason and Paul G Fahlstrom. Classes and missions of UAVs. *Encyclopedia of Aerospace Engineering*, pages 1–10, 2010.

[82] Dominik Goeke. Granular tabu search for the pickup and delivery problem with time windows and electric vehicles. *European Journal of Operational Research*, 278(3):821–836, 2019.

[83] Dominik Goeke and Michael Schneider. Routing a mixed fleet of electric and conventional vehicles. *European Journal of Operational Research*, 245(1):81–99, 2015.

[84] Bruce L Golden, Thomas L Magnanti, and Hien Q Nguyen. Implementing vehicle routing algorithms. *Networks*, 7(2):113–148, 1977.

[85] Lucio Grandinetti, Francesca Guerriero, Ferdinando Pezzella, and Ornella Pisacane. A pick-up and delivery problem with time windows by electric vehicles. *International Journal of Productivity and Quality Management*, 18(2-3):403–423, 2016.

[86] Peter Gregory and Alan Lindsay. The dimensions of driverlog. *PlanSIG 2007*, page 52, 2007.

[87] GJ Grenzdörffer, A Engel, and B Teichert. The photogrammetric potential of low-cost UAVs in forestry and agriculture. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 31(B3):1207–1214, 2008.

[88] Minh Hoàng Hà, Tat Dat Nguyen, Thinh Nguyen Duy, Hoang Giang Pham, Thuy Do, and Louis-Martin Rousseau. A new constraint programming model and a linear programming-based adaptive large neighborhood search for the vehicle routing problem with synchronization constraints. *Computers & Operations Research*, 124:105085, 2020.

[89] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[90] Andy Ham and Myoung-Ju Park. Electric vehicle route optimization under time-of-use electricity pricing. *IEEE Access*, 9:37220–37228, 2021.

[91] Andy M Ham. Integrated scheduling of m-truck, m-drone, and m-depot constrained by time-window, drop-pickup, and m-visit using constraint programming. *Transportation Research Part C: Emerging Technologies*, 91:1–14, 2018.

[92] Andy M Ham. Drone-based material transfer system in a robotic mobile fulfillment center. *IEEE Transactions on Automation Science and Engineering*, 2019.

[93] William E Hart, Carl D Laird, Jean-Paul Watson, David L Woodruff, Gabriel A Hackebeil, Bethany L Nicholson, and John D Siirola. *Pyomo-optimization modeling in python*, volume 67. Springer, 2017.

[94] Erez Hartuv, Noa Agmon, and Sarit Kraus. Scheduling spare drones for persistent task performance under energy constraints. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 532–540. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[95] Nick Hawes, Christopher Burbridge, Ferdian Jovan, Lars Kunze, Bruno Lacerda, Lenka Mudrova, Jay Young, Jeremy Wyatt, Denise Hebesberger, Tobias Kortner, et al. The strands project: Long-term autonomy in everyday environments. *IEEE Robotics & Automation Magazine*, 24(3):146–156, 2017.

[96] Ruijie He, Abraham Bachrach, and Nicholas Roy. Efficient planning under uncertainty for a target-tracking micro-aerial vehicle. In *2010 IEEE International Conference on Robotics and Automation*, pages 1–8. IEEE, 2010.

[97] Pascal Van Hentenryck and Russell Bent. *Online stochastic combinatorial optimization*. The MIT Press, 2009.

[98] Gerhard Hiermann, Richard F Hartl, Jakob Puchinger, and Thibaut Vidal. Routing a mix of conventional, plug-in hybrid, and electric vehicles. *European Journal of Operational Research*, 272(1):235–248, 2019.

[99] Gerhard Hiermann, Jakob Puchinger, Stefan Ropke, and Richard F Hartl. The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal of Operational Research*, 252(3):995–1018, 2016.

[100] Insu Hong, Michael Kuby, and Alan T Murray. A range-restricted recharging station coverage model for drone delivery service planning. *Transportation Research Part C: Emerging Technologies*, 90:198–212, 2018.

[101] John N Hooker and Greger Ottosson. Logic-based benders decomposition. *Mathematical Programming*, 96(1):33–60, 2003.

[102] Joey Hwang and David G Mitchell. 2-way vs. d-way branching for CSP. In *International Conference on Principles and Practice of Constraint Programming*, pages 343–357. Springer, 2005.

[103] Md Mainul Islam, Hussain Shareef, and Azah Mohamed. A review of techniques for optimal placement and sizing of electric vehicle charging stations. *Elect. Review*, 91:122–126, 2015.

[104] Josef Jablonskỳ et al. Benchmarks for current linear and mixed integer optimization solvers. *Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis*, 63(6):1923–1928, 2015.

[105] Joxan Jaffar and Michael J Maher. Constraint logic programming: A survey. *The journal of logic programming*, 19:503–581, 1994.

[106] Wanchen Jie, Jun Yang, Min Zhang, and Yongxi Huang. The two-echelon capacitated electric vehicle routing problem with battery swapping stations: Formulation and efficient methodology. *European Journal of Operational Research*, 272(3):879–904, 2019.

[107] Narendra Jussien, Guillaume Rochart, and Xavier Lorca. Choco: an open source Java constraint programming library. 2008.

[108] Krishna Kalyanam, David Casbeer, and Meir Pachter. Graph search of a moving ground target by a UAV aided by ground sensors with local information. *Autonomous Robots*, pages 1–13, 2020.

[109] Marisa G Kantor and Moshe B Rosenwein. The orienteering problem with time windows. *Journal of the Operational Research Society*, 43(6):629–635, 1992.

[110] Imdat Kara, Gilbert Laporte, and Tolga Bektas. A note on the lifted miller–tucker–zemlin subtour elimination constraints for the capacitated vehicle routing problem. *European Journal of Operational Research*, 158(3):793–795, 2004.

[111] Aline Karak and Khaled Abdelghany. The hybrid vehicle-drone routing problem for pick-up and delivery services. *Transportation Research Part C: Emerging Technologies*, 102:427–449, 2019.

[112] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311, 1984.

[113] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[114] Merve Keskin and Bülent Çatay. Partial recharge strategies for the electric vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 65:111–127, 2016.

[115] Merve Keskin and Bülent Çatay. A matheuristic method for the electric vehicle routing problem with time windows and fast chargers. *Computers & Operations Research*, 100:172–188, 2018.

[116] Merve Keskin, Gilbert Laporte, and Bülent Çatay. Electric vehicle routing problem with time-dependent waiting times at recharging stations. *Computers & Operations Research*, 107:77–94, 2019.

[117] Cory D Kidd, Will Taggart, and Sherry Turkle. A sociable robot to encourage social interaction among the elderly. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 3972–3976. IEEE, 2006.

[118] Jonghoe Kim and James R Morrison. On the concerted design and scheduling of multiple resources for persistent UAV operations. *Journal of Intelligent & Robotic Systems*, 74(1-2):479–498, 2014.

[119] Joris Kinable, Willem-Jan van Hoeve, and Stephen F Smith. Optimization models for a real-world snow plow routing problem. In *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, pages 229–245. Springer, 2016.

[120] Joris Kinable, Willem-Jan van Hoeve, and Stephen F Smith. Snow plow route optimization: A constraint programming approach. *IISE Transactions*, pages 1–19, 2020.

[121] Maurice W Kirby. *Operational research in war and peace: the British experience from the 1930s to 1970*. Imperial College Press, 2003.

[122] Joonho Ko, Tae-Hyoung Tommy Gim, and Randall Guensler. Locating refuelling stations for alternative fuel vehicles: a review on models and applications. *Transport Reviews*, 37(5):551–570, 2017.

[123] BO Koopman. Search and screening, operations evaluation group report 56. *Center for Naval Analysis, Alexandria, Virginia*, 1946.

[124] G Ayorkor Korsah, Anthony Stentz, and M Bernardine Dias. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12):1495–1512, 2013.

[125] Wen-Yang Ku and J Christopher Beck. Mixed integer programming models for job shop scheduling: A computational analysis. *Computers & Operations Research*, 73:165–173, 2016.

[126] Philippe Laborie. IBM ILOG CP Optimizer for detailed scheduling illustrated on three problems. In *International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems*, pages 148–162. Springer, Springer, 2009.

[127] Philippe Laborie, Jérôme Rogerie, Paul Shaw, and Petr Vilím. IBM ILOG CP Optimizer for scheduling. *Constraints*, 23(2):210–250, 2018.

[128] Edward Lam. *Hybrid optimization of vehicle routing problems*. PhD thesis, 2017.

[129] Edward Lam and Pascal Van Hentenryck. A branch-and-price-and-check model for the vehicle routing problem with location congestion. *Constraints*, 21(3):394–412, 2016.

[130] Edward Lam, Pascal Van Hentenryck, and Phil Kilby. Joint vehicle and crew routing and scheduling. *Transportation Science*, 54(2):488–511, 2020.

[131] Edward Lam, Pascal Van Hentenryck, and Philip Kilby. Joint vehicle and crew routing and scheduling. In *International Conference on Principles and Practice of Constraint Programming*, pages 654–670. Springer, 2015.

[132] Ailsa H Land and Alison G Doig. An automatic method for solving discrete programming problems. In *50 Years of Integer Programming 1958-2008*, pages 105–132. Springer, 2010.

[133] Allison Langer, Ronit Feingold-Polak, Oliver Mueller, Philipp Kellmeyer, and Shelly Levy-Tzedek. Trust in socially assistive robots: Considerations for use in rehabilitation. *Neuroscience & Biobehavioral Reviews*, 2019.

[134] Gilbert Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European journal of operational research*, 59(3):345–358, 1992.

[135] Gilbert Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, 2009.

[136] Gilbert Laporte, Jorge Riera-Ledesma, and Juan-José Salazar-González. A branch-and-cut algorithm for the undirected traveling purchaser problem. *Operations Research*, 51(6):940–951, 2003.

[137] Philippe Lebeau, Cedric De Cauwer, Joeri Van Mierlo, Cathy Macharis, Wouter Verbeke, and Thierry Coosemans. Conventional, hybrid, or electric vehicles: which technology for an urban distribution centre? *The Scientific World Journal*, 2015, 2015.

[138] Chungmok Lee. An exact algorithm for the electric-vehicle routing problem with nonlinear charging time. *Journal of the Operational Research Society*, pages 1–24, 2020.

[139] Jan Karel Lenstra and AHG Rinnooy Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, 1981.

[140] Canhong Lin, King Lun Choy, George TS Ho, Sai Ho Chung, and HY Lam. Survey of green vehicle routing problem: past and future trends. *Expert Systems with Applications*, 41(4):1118–1138, 2014.

[141] Chang Liu, Dionne M Aleman, and J Christopher Beck. Modelling and solving the senior transportation problem. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 412–428. Springer, 2018.

[142] Wing-Yue Louie, Runqi Han, and Goldie Nejat. Did anyone say bingo: A socially assistive robot to promote stimulating recreational activities at long-term care facilities. *Journal of Medical Devices*, 7(3):30944, 2013.

[143] Wing-Yue G Louie, Jacob Li, Chris Mohamed, Francis Despond, Vincent Lee, and Goldie Nejat. Tangy the robot bingo facilitator: A performance review. *Journal of Medical Devices*, 9(2):20936, 2015.

[144] Wing Yue Geoffre Louie, Jacob Li, Tiago Vaquero, and Goldie Nejat. Socially assistive robots for seniors living in residential care homes: User requirements and impressions. In *Human-Robot Interactions: Principles, Technologies and Challenges*. 2015.

[145] Wing-Yue Geoffrey Louie, Tiago Vaquero, Goldie Nejat, and J Christopher Beck. An autonomous assistive robot for planning, scheduling and facilitating multi-user activities. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 5292–5298. IEEE, 2014.

[146] Alessandro Luè, Alberto Colorni, Roberto Nocerino, and Valerio Paruscio. Green move: An innovative electric vehicle-sharing system. *Procedia-Social and Behavioral Sciences*, 48:2978–2987, 2012.

[147] Alan K Mackworth. Consistency in networks of relations. *Artificial intelligence*, 8(1):99–118, 1977.

[148] Parikshit Maini and PB Sujit. On cooperation between a fuel constrained UAV and a refueling ugv for large scale mapping applications. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1370–1377. IEEE, 2015.

[149] Satyanarayana G Manyam, Kaarthik Sundar, and David W Casbeer. Cooperative routing for an air-ground vehicle team–exact algorithm, transformation method, and heuristics. *IEEE Transactions on Automation Science and Engineering*, 2019.

[150] Derek McColl and Goldie Nejat. Recognizing emotional body language displayed by a human-like social robot. *International Journal of Social Robotics*, 6(2):261–280, 2014.

[151] Derek McColl and Goldie Nejat. A socially assistive robot that can monitor affect of the elderly during mealtime assistance. *Journal of Medical Devices*, 8(3):30941, 2014.

[152] Nimrod Megiddo. *Progress in Mathematical Programming: Interior-Point and Related Methods*. Springer Science & Business Media, 2012.

[153] Luc Mercier and Pascal Van Hentenryck. Edge finding for cumulative scheduling. *INFORMS Journal on Computing*, 20(1):143–153, 2008.

[154] Bernard Michini, Tuna Toksoz, Joshua Redding, Matthew Michini, Jonathan How, Matthew Vavrina, and John Vian. Automated battery swap and recharge to enable persistent UAV missions. In *Infotech@ Aerospace 2011*, page 1405. 2011.

[155] Clair E Miller, Albert W Tucker, and Richard A Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.

[156] Stuart Mitchell, Michael OSullivan, and Iain Dunning. PuLP: a linear programming toolkit for Python. *The University of Auckland, Auckland, New Zealand*, 2011.

[157] Reza Moghdani, Khodakaram Salimifard, Emrah Demir, and Abdelkader Benyettou. The green vehicle routing problem: a systematic literature review. *Journal of Cleaner Production*, page 123691, 2020.

[158] Sharaf C Mohamed, Sanjif Rajaratnam, Seung Tae Hong, and Goldie Nejat. Person finding: An autonomous robot search method for finding multiple dynamic users in human-centered environments. *IEEE Transactions on Automation Science and Engineering*, 2019.

[159] Alejandro Montoya, Christelle Guéret, Jorge E Mendoza, and Juan G Villegas. The electric vehicle routing problem with nonlinear charging function. *Transportation Research Part B: Methodological*, 103:87–110, 2017.

[160] Stanislav Murín and Hana Rudová. Scheduling of mobile robots using constraint programming. In *International Conference on Principles and Practice of Constraint Programming*, pages 456–471. Springer, 2019.

[161] Chase C Murray and Amanda G Chu. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies*, 54:86–109, 2015.

[162] Nicholas Nethercote, Peter J Stuckey, Ralph Becket, Sebastian Brand, Gregory J Duck, and Guido Tack. MiniZinc: Towards a standard CP modelling language. In *International Conference on Principles and Practice of Constraint Programming*, pages 529–543. Springer, 2007.

[163] Alena Otto, Niels Agatz, James Campbell, Bruce Golden, and Erwin Pesch. Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey. *Networks*, 72(4):411–458, 2018.

[164] Manfred Padberg and Giovanni Rinaldi. Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters*, 6(1):1–7, 1987.

[165] Laurent Perron. Operations research and constraint programming at Google. In *International Conference on Principles and Practice of Constraint Programming*, pages 2–2. Springer, 2011.

[166] Gilles Pesant, Michel Gendreau, Jean-Yves Potvin, and Jean-Marc Rousseau. An exact constraint logic programming algorithm for the traveling salesman problem with time windows. *Transportation Science*, 32(1):12–29, 1998.

[167] Gilles Pesant, Michel Gendreau, Jean-Yves Potvin, and Jean-Marc Rousseau. On the flexibility of constraint programming models: From single to multiple time windows for the traveling salesman problem. *European Journal of Operational Research*, 117(2):253–263, 1999.

[168] Gilles Pesant, Claude-Guy Quimper, and Alessandro Zanarini. Counting-based search: Branching heuristics for constraint satisfaction problems. *Journal of Artificial Intelligence Research*, 43:173–210, 2012.

[169] Chiara Piacentini, Sara Bernardini, and J Christopher Beck. Autonomous target search with multiple coordinated UAVs. *Journal of Artificial Intelligence Research*, 65:519–568, 2019.

[170] Michael L Pinedo. *Scheduling: theory, algorithms, and systems*. Springer, 2016.

[171] Ryan R Pitre, X Rong Li, and R Delbalzo. UAV route planning for joint search and track missions - An information-value approach. *IEEE Transactions on Aerospace and Electronic Systems*, 48(3):2551–2565, 2012.

[172] Stefan Poikonen and Bruce Golden. The mothership and drone routing problem. *INFORMS Journal on Computing*, 32(2):249–262, 2020.

[173] Grzegorz Radzki, Amila Thibbotuwawa, and Grzegorz Bocewicz. UAVs flight routes optimization in changing weather conditions–constraint programming approach. *Applied Computer Science*, 15(3), 2019.

[174] Ramin Raeesi and Konstantinos G Zografos. The electric vehicle routing problem with time windows and synchronised mobile battery swapping. *Transportation Research Part B: Methodological*, 140:101–129, 2020.

[175] Jean-Charles Régin. A filtering algorithm for constraints of difference in CSPs. In *AAAI*, volume 94, pages 362–367, 1994.

[176] Zeinab Rezvani, Johan Jansson, and Jan Bodin. Advances in consumer electric vehicle adoption research: A review and research agenda. *Transportation research part D: transport and environment*, 34:122–136, 2015.

[177] Roberto Roberti and Min Wen. The electric traveling salesman problem with time windows. *Transportation Research Part E: Logistics and Transportation Review*, 89:32–52, 2016.

[178] Matthias Rogge, Evelien van der Hurk, Allan Larsen, and Dirk Uwe Sauer. Electric bus fleet size and mix problem with optimization of charging infrastructure. *Applied Energy*, 211:282–295, 2018.

[179] Francesca Rossi, Peter Van Beek, and Toby Walsh. *Handbook of constraint programming.* Elsevier, 2006.

[180] Louis-Martin Rousseau, Michel Gendreau, Gilles Pesant, and Filippo Focacci. Solving VRPTWs with constraint programming based column generation. *Annals of Operations Research*, 130(1-4):199–216, 2004.

[181] Francisco Rubio, Francisco Valero, and Carlos Llopis-Albert. A review of mobile robots: Concepts, methods, theoretical framework, and applications. *International Journal of Advanced Robotic Systems*, 16(2):1729881419839596, 2019.

[182] Ons Sassi, Wahiba Ramdane Cherif-Khettaf, and Ammar Oulamara. Iterated tabu search for the mix fleet vehicle routing problem with heterogenous electric vehicles. In *Modelling, Computation and Optimization in Information Systems and Management Sciences*, pages 57–68. Springer, 2015.

[183] Michael Schneider, Andreas Stenger, and Dominik Goeke. The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science*, 48(4):500–520, 2014.

[184] Christian Schulte, Guido Tack, and Mikael Z Lagerkvist. Modeling and programming with Gecode. *Schulte, Christian and Tack, Guido and Lagerkvist, Mikael*, 1, 2010.

[185] Markus Sebastian Schwenk, Tiago Stegun Vaquero, Goldie Nejat, and Kai Oliver Arras. Schedule-based robotic search for multiple residents in a retirement home environment. In *AAAI*, pages 2571–2577, 2014.

[186] Mostafa Setak and Asal Karimpour. A mathematical model for the electric vehicle routing with time windows considering queuing system at charging stations and alternative paths. *Journal of Industrial and Systems Engineering*, 12(1):284–306, 2019.

[187] Sujie Shao, Shaoyong Guo, and Xuesong Qiu. A mobile battery swapping service for electric vehicles based on a battery swapping van. *Energies*, 10(10):1667, 2017.

[188] Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *International Conference on Principles and Practice of Constraint Programming*, pages 417–431. Springer, Springer, 1998.

[189] Lixin Situ. Electric vehicle development: the past, present & future. In *2009 3rd International Conference on Power Electronics Systems and Applications (PESA)*, pages 1–3. IEEE, 2009.

[190] Byung Duk Song, Jonghoe Kim, Jeongwoon Kim, Hyorin Park, James R Morrison, and David Hyunchul Shim. Persistent UAV service: An improved scheduling formulation and prototypes of system components. *Journal of Intelligent & Robotic Systems*, 74(1-2):221–232, 2014.

[191] Jörg Stückler, David Droeschel, Kathrin Gräve, Dirk Holz, Michael Schreiber, Angeliki Topalidou-Kyniazopoulou, Max Schwarz, and Sven Behnke. Increasing flexibility of mobile manipulation and intuitive human-robot interaction in robocup@ home. In *Robot Soccer World Cup*, pages 135–146. Springer, 2013.

[192] Jorg Stuckler, Dirk Holz, and Sven Behnke. Robocup@ home: Demonstrating everyday manipulation skills in robocup@ home. *IEEE Robotics & Automation Magazine*, 19(2):34–42, 2012.

[193] Ltd. SZ DJI Technology Co. Matrice 200 series specs, year = 2020, url = https://www.dji.com/matrice-200-series/info, urldate = 2020-02-11.

[194] Zeynab Talebpour and Alcherio Martinoli. Risk-based human-aware multi-robot coordination in dynamic environments shared with humans. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3365–3372. IEEE, 2018.

[195] Ziye Tang, Willem-Jan van Hoeve, and Paul Shaw. A study on the traveling salesman problem with a drone. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 557–564. Springer, 2019.

[196] Robert Tarjan. Depth-First Search and Linear Graph Algorithms. *SIAM J. Comput.*, 1:146–160, 1972.

[197] Paolo Toth and Daniele Vigo. *The vehicle routing problem.* SIAM, 2002.

[198] Paolo Toth and Daniele Vigo. *Vehicle routing: problems, methods, and applications.* SIAM, 2014.

[199] Tony T Tran, Tiago Vaquero, Goldie Nejat, and J Christopher Beck. Robots in retirement homes: Applying off-the-shelf planning and scheduling to a team of assistive robots. *Journal of Artificial Intelligence Research*, 58:523–590, 2017.

[200] Tuan Tony Tran. *Decomposition Models for Complex Scheduling Applications.* PhD thesis, 2017.

[201] Masoumeh Vali and Khodakaram Salimifard. A constraint programming approach for solving multiple traveling salesman problem. In *The Sixteenth International Workshop on Constraint Modelling and Reformulation*, 2017.

[202] Pascal Van Hentenryck. *The OPL optimization programming language*. Mit Press, 1999.

[203] Willem-Jan Van Hoeve. The alldifferent constraint: A survey. *arXiv preprint cs/0105015*, 2001.

[204] Willem-Jan van Hoeve and Irit Katriel. Global constraints. In *Foundations of Artificial Intelligence*, volume 2, pages 169–208. Elsevier, 2006.

[205] Tiago Vaquero, Sharaf Christopher Mohamed, Goldie Nejat, and J Christopher Beck. The implementation of a planning and scheduling architecture for multiple robots assisting multiple users in a retirement home setting. In *Artificial Intelligence Applied to Assistive Technologies and Smart Environments (AAAI 2015)*, 2015.

[206] Manuela M Veloso, Joydeep Biswas, Brian Coltin, and Stephanie Rosenthal. Cobots: Robust symbiotic autonomous mobile service robots. In *IJCAI*, page 4423. Citeseer, 2015.

[207] Petr Vilím. $O(n \log n)$ filtering algorithms for unary resource constraint. In *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming*, pages 335–347. Springer, 2004.

[208] Petr Vilím. Edge finding filtering algorithm for discrete cumulative resources in $O(kn \log n)$. In *International Conference on Principles and Practice of Constraint Programming*, pages 802–816. Springer, 2009.

[209] Wayne L Winston and Jeffrey B Goldberg. *Operations research: applications and algorithms*, volume 3. Thomson Brooks/Cole Belmont, 2004.

[210] Thomas Wisspeintner, Tijn Van Der Zant, Luca Iocchi, and Stefan Schiffer. Robocup@ home: Scientific competition and benchmarking for domestic service robots. *Interaction Studies*, 10(3):392–426, 2009.

[211] Kevin Yu, Ashish Kumar Budhiraja, Spencer Buebel, and Pratap Tokekar. Algorithms and experiments on routing of unmanned aerial vehicles with mobile recharging stations. *Journal of Field Robotics*, 36(3):602–616, 2019.

[212] Kevin Yu, Ashish Kumar Budhiraja, and Pratap Tokekar. Algorithms for routing of unmanned aerial vehicles with mobile recharging stations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–5. IEEE, 2018.

[213] Yan Zhou, Michael Wang, Han Hao, Larry Johnson, and Hewu Wang. Plug-in electric vehicle market penetration and incentives: a global review. *Mitigation and Adaptation Strategies for Global Change*, 20(5):777–795, 2015.

[214] Xiaorong Zuo, Yiyong Xiao, Meng You, Ikou Kaku, and Yuchun Xu. A new formulation of the electric vehicle routing problem with time windows considering concave nonlinear charging function. *Journal of Cleaner Production*, 236:117687, 2019.