TEL AVIV UNIVERSITY

The Iby and Aladar Fleischman Faculty of Engineering

The Zandman-Slaner School of Graduate Studies

A Column Generation Approach for Public Transit Enhanced Robotic Delivery Services

A thesis submitted toward the degree of Master of Science in Industrial Engineering

by

Yishay Shapira

December 2023

TEL AVIV UNIVERSITY

The Iby and Aladar Fleischman Faculty of Engineering

The Zandman-Slaner School of Graduate Studies

A Column Generation Approach for Public Transit Enhanced Robotic Delivery Services

A thesis submitted toward the degree of Master of Science in Industrial Engineering

by

Yishay Shapira

This research was carried out in the Department of Industrial Engineering

Under the supervision of Dr. Mor Kaspi

December 2023

Abstract

Many new technology initiatives that rely on vehicle autonomy capabilities have emerged in recent years. One prominent concept is based on Autonomous Mobile Robots (AMRs). In such systems, small wheeled robots provide point-to-point deliveries on sidewalks at pedestrian speed. They are powered by small batteries, limiting their service range to around 3 km.

The system considered in this work consists of a fleet of vehicles distributed in multiple mini depots along the service area. These robots are assigned to service requests characterized by pick-up and delivery locations and corresponding service time windows. We examine the potential of enhancing the service by public transit. That is, allowing the robots to fulfill parts of their journey on board public transit vehicles. As the robots do not discharge while traveling on board the public transit vehicles, this extension comprises multiple opportunities. First, the service range can be extended, and, in some cases, service durations can be shortened. Second, the overall energy consumption can be reduced. This work focuses on the operational planning problem in the studied AMR based services, consisting of assignment, routing and timing decisions. The problem represents a special case of the well-known Pick-up and Delivery Problem (PDP), with full truck load and multiple modes of transport. We develop two mixed integer programming formulations for the problem: an arc-based formulation and a route-based formulation. The arc-based formulation explicitly represents each robot's potential leg and decides upon the legs to be travelled. The latter considers complete feasible routes with the aim of selecting the best route (and robot) for each request. While the route-based formulation has a more compact structure, the number of routes that may be considered grows exponentially with the network size. To overcome this, we develop a column generation approach. Specifically, we define an initial set of potentially good routes for each robot-request pair and then formulate the underlying sub-problem of finding new promising routes as a resource constrained shortest path problem. We develop a four-stage dynamic programming algorithm to solve the sub-problem. Subsequently, by exploiting robot-request symmetries we can reduce significantly the number of sub-problems solved at each iteration. In addition, as the robot-request sub-problems are independent, we apply parallel computing to solve multiple sub-problems simultaneously. These actions allow us to reduce the computing time required for each column generation iteration.

The numerical experiment results show that the column generation approach can solve instances with up to 150 requests in a few seconds, while the arc-based formulation only enables solving instances with up to 15 requests. Furthermore, we conducted a case study

utilizing real-world data from the city of Tel Aviv. The outcomes of this study demonstrate how the integration of public transit extends the service range of the robots, enabling them to handle a greater number of service requests while conserving their energy. To conclude, this study highlights the potential benefits of enhancing AMR-based delivery services by public transit and provides a practical approach to solving the operational planning problem.

Table of Contents

List of Notations

n	The number of passenger requests
$\mathcal{P}$	Set of pick-up nodes ($i = 0, \dots, n-1$)
$\mathcal{D}$	Set of drop-off nodes ($i = n, \dots, 2n-1$), the drop-off node of pick-up node i is $n+i$
$\mathcal{R}$	Set of request nodes $\mathcal{R} = \mathcal{P} \cup \mathcal{D}$
$\mathcal{T}$	Set of transfer nodes
$\mathcal{C}$	Set of depot nodes
$\mathcal{N}$	Set of all nodes $\mathcal{N} = \mathcal{P} \cup \mathcal{D} \cup \mathcal{T} \cup \mathcal{C}$
$\mathcal{V}$	Set of robots
$\mathcal{W}$	Set of service lines
λ_w	Ordered set of transfer nodes representing service line $w \in \mathcal{W}$ $(\lambda_w \subseteq \mathcal{T})$
J	Robot battery capacity
K	Capacity of the depot nodes
Q	Capacity of the transfer nodes
F_i	Penalty cost of unserved request $i \in \mathcal{P}$
δ_{ij}	Direct travel time between nodes i and j
μ_{ij}	The cost of traveling directly between nodes i and j
ρ_{ij}	Battery discharge while traveling directly between nodes i and j
e_i	Earliest time window of request $i \in \mathcal{R}$
l_i	Latest time window of request $i \in \mathcal{R}$
g_i	Service time at node $i \in \mathcal{N}$
p_i	Departure time of the service line from transfer node $i \in \mathcal{T}$
o_v	Origin depot of robot $v \in \mathcal{V}$
S_i	The number robots initially located at depot node $i \in \mathcal{C}$ ($S_i = \sum_{v \in V: o_v = i} 1$)
γ_i	The shortest path from pickup node $i \in \mathcal{P}$ to drop-off node $i+n \in \mathcal{D}$
μ_p	Cost of path $p \in \Delta$
Δ	Set of all paths
Δ_r	Set of all paths for request r $\in \mathcal{R}$
Δ_v	Set of all paths for robot v $\in \mathcal{V}$

Δ_c	Set of all paths that end in depot node $c \in \mathcal{C}$
$\Delta_{c'}$	Set of all paths that start in depot node $c \in \mathcal{C}$
Δ_t	Set of all paths that use transfer node $t \in \mathcal{T}$
$r(p)$	The request associated with path $p \in \Delta$
$v(p)$	The robot associated with path $p \in \Delta$
$c(p)$	The start depo associated with path $p \in \Delta$
$c'(p)$	The end depo associated with path $p \in \Delta$
θ_p	The set of transfer nodes associated with path $p \in \Delta$
S	The current step in the shortest path sub-problem
T	The current time in the shortest path sub-problem
B	The current battery state in the shortest path sub-problem
N	The current node in the shortest path sub-problem
θ	The cost function in the shortest path sub-problem
σ	The time function in the shortest path sub-problem
ϵ	The energy function in the shortest path sub-problem
ρ	The node function in the shortest path sub-problem
$\mu_{N,c}$	The travel cost consumed while the robot travels from node N to return depot c in the shortest path sub-problem
$\tau_{N,c}$	The travel time consumed while the robot travels from node N to return depot c in the shortest path sub-problem
$\beta_{N,c}$	The battery consumed while the robot travels from node N to return depot c in the shortest path sub-problem
NS	Sets the dimensions of the network area in the experiment
H	The planning horizon in the experiment
TW	Defines the time window size of the request nodes in the experiment
η	Fixed factor represents a different average speed of a public transit vehicles in the experiment
E	Cost factor in the experiment
ψ	Public transit cost factor in the experiment
τ	The center of the time window of a pick-up node in the experiment
τ'	The center of the time window of a drop-off node in the experiment

List of Figures

List of Tables

1. Introduction

Small package delivery services are becoming more and more popular with the rise of on-demand economy services, in particular, e-commerce and fast-food deliveries (Allen et al., 2021). To meet the growing need for home delivery within urban areas, new innovative delivery concepts have been developed, involving cargo bikes, robots, lockers, and more (Boysen et al., 2021; Chen et al., 2021). Autonomous transportation may have a key role in the development of such services, and many new technology initiatives that rely on vehicle autonomy capabilities have emerged in recent years. These initiatives seek to improve urban logistics by reducing costs and greenhouse gas emissions while improving the quality of service (Zhang et al., 2021). One prominent concept is based on Autonomous Mobile Robots (AMRs). In such systems, small wheel-based robots travel on sidewalks at pedestrian speed, providing point-to-point deliveries (Jennings and Figliozzi, 2019). Typically, the robots are powered by small batteries. This feature coupled with the relatively slow traveling speed limits the service range to approximately three kilometres. Notable large-scale implementations of AMRs include Starship (2022), FedEx Roxo (2022), and Amazon Scout (2022).

The development of AMRs and their deployment has seen a rapid growth in the last couple of years, according to Fortune Business Insights, the global delivery robots market size is projected to grow from 306.3 million dollars in 2023 to 2,143.1 million dollars by 2030. Growing demand for contactless services, labor shortage, and increased e-commerce activity boost the growth of the delivery robots market globally. Alongside, AMR services are constantly evolving, introducing new operational problems to the research community, which have already highlighted the potential of AMRs in various areas. In the AMR service at the focus of this study, a set of delivery requests is given, where each request is characterized by a pair of pick-up and delivery locations as well as time windows during which service can begin in these locations. A fleet of robots is distributed among several depots in the service area. To extend the reach of the service, the robots are allowed to perform parts of their journey on board public transit vehicles. Specifically, the robots are loaded in a way that does not decrease passenger capacity and do not impact passenger service times at the public transit stations. AMR boarding and disembarking are performed using a designated ramp. A set of fixed public transit lines that operate in the service area are given. Each fixed line is characterized by a sequence of stations, the traveling times between them, and the frequency of the service. When a robot is assigned to a task, it needs to travel from its origin depot to a pickup point, then travel to the associated delivery point, and finally return to one of the depots for recharging. Each leg

of this journey can be performed directly by the robot (using battery power) or be performed partially on board a public transit vehicle, which travels at a higher speed and does not require battery discharging.

Enhancing AMR delivery services by public transit comprises multiple opportunities. First, the service range can be extended, allowing the robots to serve requests that were infeasible to serve without the public transit, and at the same time, shortening the service times in some cases. Second, reducing the overall energy consumption due to the use of transit transportation for portions of the journeys. Several research studies have already explored related aspects, although with differences from the system examined in this study.

We consider the static version of the operational problem, i.e., the case where information regarding the requests is known long enough in advance for operational decision-making. In our setting, all given requests must be served by the AMRs or by an alternative service (outsourcing) which is represented by a penalty at a greater cost. For each request, the problem is to decide upon the means it will be served. Specifically, for requests served by the AMRs, we need to decide which robot will be assigned to each request, the robots' routes, and the depots to which they will return. This needs to be determined while respecting the maximal number of robots that can be carried simultaneously by a public transit service and the capacities of the robot depots. The goal of the decision problem is to minimize the total operational cost which is composed of the AMR's operational costs and costs associated with the alternative service. This set of attributes can be described by a graph with four types of nodes – depot, pickup, delivery, and transportation nodes, and with two types of arcs – electric arcs representing robot movements using battery power and public transit arcs representing movements on board the public transit lines.

The operational planning problem in AMR based services represents a special case of the well-known Pick-up and Delivery Problem (PDP), which concerns the efficient planning of the transport of objects between given origins and destinations (Berbeglia et al. 2007). In particular, as the robots' carrying capacity is small, in many cases they are limited to serve one delivery at a time, this restriction is also imposed to ensure the security of the deliveries. That is, the operational problem under consideration is a generalization of the Full Truck Load PDP (Gendreau et al., 2015). In addition, while in the classic PDP, transportation is provided by a single mode of transportation, here different parts of the journey may be performed by different modes, namely, robots or public transit.

In conclusion, the potential of enhancing AMR services with public transit is an approach that is only beginning to be explored by the research community. The AMR operational

problem presented in this study differs from previous studies by considering together the following characteristics: (1) fixed AMR capacity on the public transit vehicles; (2) representation of the AMR battery capacity and discharging rate; (3) AMR depot capacities; and (4) AMR's may return to any available depot. By formulating this operational problem and proposing an efficient solution method, we wish to close this gap in the literature.

The contribution of this thesis is as follows. First, we define the operational problem of a public transit enhanced AMR delivery service. Second, we present two MILP formulations of the problem: arc-based and path-based. Third, we devise a column-generation approach tailored to the characteristics of the problem. Lastly, we perform an extensive numerical experiment using synthetic problem instances as well as problem instances derived from a case study in Tel Aviv.

The remainder of this thesis is organized as follows: in section 2 we provide a literature review. In Section 3 we formally define the problem using the arc-based formulation and present the path-based formulation which is the basis for the column generation approach. In Section 4, we present the column generation framework. In particular, we define the sub-problem of finding new columns with negative reduced costs and formulate it as four-step dynamic program. In Section 5, we present the numerical experiment we have conducted to test the proposed approach. Finally, in Section 6, we provide our conclusions and suggest directions for further research.

2. Literature Review

This study introduces a column generation approach designed for addressing the operational challenges inherent in a public transit-enhanced Autonomous Mobile Robot (AMR) delivery service an application-specific variant of the Pickup and Delivery Problem (PDP). Subsequently, the following section is dedicated to an in-depth examination of three critical topics. Section 2.1 provides a comprehensive exploration of the existing PDP literature. In Section 2.2, we conduct a review of recent literature on AMRs, with a specific focus on similar applications. Section 2.3 delves into a critical review of column-generation approaches employed to tackle transportation problems closely related to our study.

2.1. Pickup and Delivery Problems

The literature on Pickup and Delivery problems (PDP) has grown significantly in recent years, with a range of different problem formulations and solution methods being proposed. In the general Pickup and Delivery problems (GPDP), a set of routes must be considered in order to satisfy transportation requests. A fleet of vehicles is available to operate the routes, each vehicle has a given capacity, a start point, and an end point. Each request specifies the size of the load to be transported, the location where it is to be picked up (the origins), and the location where it is to be delivered (the destinations) (Savelsbergh and Sol. 1995).

Parragh et al. (2007) presented a survey of PDPs, describing them as a special class of GPDPs, with transportation requests, each associated with an origin and a destination, resulting in paired pickup and delivery points. In contrast, in GPDP a single pickup point can be connected to multiple delivery points or vice versa. The main context in which PDPs are raised is ground vehicle routing, but novel approaches have also been proposed for PDPs in maritime and aerial domains. For example, Christiansen and Nygreen (1998) presented a method for solving ship routing problems with inventory constraints, while Choudhury et al. (2019) presented a PDP Multi-Layered system with drones that can either fly or ride on vehicles for segments of their routes.

PDP can be further classified into static and dynamic problems. A routing problem is said to be static when all the input data of the problem are known before routes are constructed. There are three solution methods for the static PDP: exact methods, heuristics, and metaheuristics. A detailed benchmark of the main solution methods is described in the survey of Parragh et al. (2007). Brouer et al. (2004) develop a branch and cut and price algorithm that is capable of solving to optimality problem instances with up to 205 requests.

The dynamic routing problem is one in which some of the input data are revealed or updated during the period of time in which operations take place. The input data which are revealed over time in PDPs are generally the user requests (Berbeglia et al., 2010). Several heuristics and metaheuristic solution methods have been proposed for dynamic PDPs. Surveys on dynamic routing can be found in Ghiani et al. (2003) and Berbeglia et al. (2010).

The problem of focus in this study can be seen as a variant of the Pickup and Delivery Problem with Time Windows (PDPTW). In this problem, the set of requests (pickup-delivery pairs) is subject to time windows, i.e. the vehicle needs to load the parcel from the pickup point within a given time frame and unload it within another time frame. Baldacci et al. (2011) presented an exact algorithm for the PDPTW based on a set-partitioning like integer formulation. They also described a bounding procedure that finds a near-optimal dual solution of the LP-relaxation of the formulation by combining two dual ascent heuristics and a cut-and-column generation procedure.

Due to their small capacity, robots often serve one delivery at a time. This case is known as the one-commodity Full Truck Load PDP (1-FT-PDP). The term Full Truck Load implies the unit capacity of the vehicle and the unit supply/demand of the requests. Gendreau et al. (2015) introduce solution approaches that are based on branch-and-cut algorithms.

A combination of the characteristics described above represents the special case of the PDP that is called Full Truck Load PDP with time windows (FT-PDPTW). Gronalt et al. (2003) are the first to consider this problem, while focusing on minimizing empty vehicle movements. They propose different heuristic algorithms for the problem. Caris and Janssens (2009) deal with the problem of Pre- and end-haulage of intermodal container terminals, the drayage of containers in the service area of an intermodal terminal is modeled as an FTPDPTW. A two-phase insertion heuristic is proposed to construct an initial solution. This solution is improved with a local search heuristic based on three neighborhoods. The context of the study of Janssens and Braekers (2015) is transporting goods to the customer, which is more similar to our study, they present an exact solution based on a set partitioning approach.

2.2. Autonomous Mobile Robots

The development of AMRs and their deployment has seen a rapid growth in the last couple of years. Alongside, AMR services are constantly evolving, introducing new operational problems to the research community, which have already highlighted the potential of AMRs in various areas. Boysen et al. (2018) and Ostermeier et al. (2022), study a truck-and-robot delivery concept in which the AMR's are loaded on trucks and are deployed at several locations

in the city from which they move to the customers. As compared to the traditional truck based services, this approach is shown to reduce considerably the delivery costs and the number of trucks required to perform the deliveries. Alfandari et al. (2022) study an AMR last-mile delivery service with the goal of minimizing tardiness based on customer delivery deadlines. They consider three major tardiness indicators, formulate the problem as a Mixed-Integer Programming (MIP), and apply an efficient branch-and-Benders-cut scheme to handle realistic instances. They analyze the impact of various factors such as the number of available facilities, the coverage radius of autonomous robots, and their speed on the quality of service and environmental costs. Liu et al. (2021) examined how online retailers could use AMR sharing capability in order to cope with demand surges during shopping festivals. The study examined an online retailer's AMR capacity sharing strategies with a logistics firm under a demand surge. Sharing models were developed under unilateral and bidirectional option contracts, and the optimal ex-ante and ex-post sharing strategies, as well as the corresponding initial, option, and total sharing quantities, were derived.

Additionally, alongside the ground-based AMRs, autonomous drones have gained prominence in the realm of autonomous transportation. Benarbia and Kyamakya (2021) survey a set of relevant research issues and highlight representative solutions and concepts that have been proposed in the design and modeling of the logistics of drone delivery systems. Prominent examples incude: VRP with drones (Ham, 2018; Wang and Sheu, 2019; Jeon et al., 2021), drone/task assignment (Grippa et al. 2018) and flight range issue (Huang et al. 2021).

In a closely related work to ours, Mourad et al. (2021) studied a combined service in which autonomous delivery services are integrated into a passenger transportation system. The operational problem studied in this work differs in three main aspects: (1) the capacity of the public transit vehicles is assumed to be constant, i.e., uncertainty in capacity is not considered. (2) vehicle range is represented by the energy consumed rather than by the total distance traveled, as the robot path may include non-battery consuming public transit arcs. (3) the capacity of the robot recharging depots is explicitly considered.

Recently, De Maio et al. (2023) examined a last-mile delivery service employing ground drones that can use public transit for parts of their routes. They developed a tailored destroy-and-repair mechanism and embedded it into a neighbourhood search algorithm. Results of a case study in Rome, Italy, shows a reduction of up to 7.5% of the costs as compared to traditional services. The problem considered in De Maio et al. (2023) differs from our problem in the following aspects. First, each request is represented by a single location, i.e., they study a vehicle routing problem, rather than a PDP. Second, time windows for the service requests

are not enforced. Lastly, the ground drones are constrained to returning solely to the originating depot point. In contrast, we study a PDP with time windows in which robots are allowed to return to any non-fully occupied depot.

2.3. Column Generation

Column Generation was initially introduced by Dantzig and Wolfe (1960) as a decomposition method for solving large linear programming models, typically having an exponential number of decision variables. The main idea behind this approach is to start with a small subset of the variables and to incrementally add only variables that may improve the tentative solution. The framework consists of a variable restricted version of the original problem (*master problem*) and a pricing problem used to identify new variables that should be added to the restricted set in order to improve the solution (*pricing subproblem*). This framework iteratively solves the two problems, until no new columns can be found, that is, the algorithm converges to the optimal solution (Lübbecke and Desrosiers, 2005; Desaulniers et al., 2006).

Column Generation has been widely applied to solve problems arising in the fields of logistics and scheduling, in particular, Vehicle Routing Problems (VRP). Recent examples include: multi-trip VRP (Paradiso et al., 2020), VRP with synchronization constraints (Fink et al., 2019), VRP with drones (Wang and Sheu, 2019; Xia et al., 2023), electric VRP (Zang et al., 2022) and the selective Dial-a-Ride Problem (Rist and Forbes, 2022).

The column generation procedure can be computationally expensive, especially when solving large-scale problems. To overcome this limitation, several studies have proposed the use of parallel computing to speed up the column generation procedure. Yu et al. (2022) propose two algorithms to solve the “pricing subproblem” which corresponds to the resource-constrained shortest path problem and used parallel computing to improve the column generation for solving the linear programming relaxations and can obtain heuristic integer solutions with small optimality gaps.

In conclusion, Column Generation is a powerful optimization technique that is particularly useful in solving large-scale transportation problems. Its ability to handle a large number of variables and constraints, as well as its ability to handle multiple objectives make it a valuable tool in the field of transportation. There are several ways to implement column generation in transportation problems, such as Dantzig-Wolfe decomposition and Benders decomposition, and the choice of the method depends on the specific problem and the available data.

3. Problem Definition

In this section, we present two Mixed Integer Linear Programming (MILP) formulations for the operational problem, arc-based and path-based. The former serves as means to define in detail the problem, it represents explicitly each potential leg of the robots and decides upon the legs to be travelled. The latter considers complete feasible paths to select the best path (and robot) for each request.

3.1. The arc-based model

The input of the model consists of information regarding the service requests, the fleet of robots, the transportation network, and the public transit lines. In particular, each service request is characterized by a pickup location, a drop-off location, corresponding time windows, and a penalty cost. The robots are characterized by their battery range and initial locations. The transportation network consists of arcs connecting all system nodes (pickups and drop-offs, public transit stations and depots). Each arc is associated with a travel time (robot or public transit), battery consumption and traveling cost. In addition, each depot is characterized by a capacity, representing the maximum number of robots that can be parked at the end of the planning period. Replications of the public transit lines, namely, service lines, are represented as ordered sequences of transfer nodes and the periods of time during which they are served. Furthermore, the maximum number of robots that can simultaneously board a public transit vehicle is given.

The objective of the model is to minimize operational costs, combining robot movement related costs and the penalty costs. Due to the limited capacity of the robots, a noteworthy assumption we make in this model is that during the planning period, each robot can serve at most one request. Additional sets of constraints are used to enforce: routing and timing feasibility, battery range, public transit service capacity, and the depot capacities.

Indices:

i, j, k	node
v	robot
w	service line

Parameters:

n	the number of passenger requests
$\mathcal{P}$	set of pick-up nodes ($i = 0, \dots, n-1$)
$\mathcal{D}$	set of drop-off nodes ($i = n, \dots, 2n-1$), the drop-off node of pick-up node i is $n+i$
$\mathcal{R}$	set of request nodes $\mathcal{R} = \mathcal{P} \cup \mathcal{D}$
$\mathcal{T}$	set of transfer nodes
$\mathcal{C}$	set of depot nodes
$\mathcal{N}$	set of all nodes $\mathcal{N} = \mathcal{P} \cup \mathcal{D} \cup \mathcal{T} \cup \mathcal{C}$
$\mathcal{V}$	set of robots
$\mathcal{W}$	set of service lines
λ_w	ordered set of transfer nodes representing service line $w \in \mathcal{W}$ $(\lambda_w \subseteq \mathcal{T})$
J	robot battery capacity
K	capacity of the depot nodes
Q	capacity of the transfer nodes
F_i	Penalty cost of an unserved request $i \in \mathcal{P}$
δ_{ij}	direct travel time between nodes i and j
μ_{ij}	the cost of traveling directly between nodes i and j
ρ_{ij}	battery discharge while traveling directly between nodes i and j
e_i	earliest time window of request $i \in \mathcal{R}$
l_i	latest time window of request $i \in \mathcal{R}$
g_i	service time at node $i \in \mathcal{N}$
p_i	departure time of the service line from transfer node $i \in \mathcal{T}$
o_v	origin depot of robot $v \in \mathcal{V}$
S_i	the number robots initially located at depot node $i \in \mathcal{C}$ ($S_i = \sum_{v \in V: o_v = i} 1$)
γ_i	the shortest path from pickup node $i \in \mathcal{P}$to drop-off node $i + n \in \mathcal{D}$

Decision Variables:

x_{ij}^v	1 if robot $v \in \mathcal{V}$ travels from node $i \in N$ to node $j \in N$, 0 otherwise
u_i	1 if pickup $i \in \mathcal{P}$ is unserved, indicating a penalty cost, 0 otherwise

h_i^v The departure time of robot $v \in \mathcal{V}$ from depot node $i \in \mathcal{C}$

$\bar{h}_i^v$ The arrival time of robot $v \in \mathcal{V}$ to depot node $i \in \mathcal{C}$

b_i The service start time of a robot at request node $i \in \mathcal{R}$

$$Minimize \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}\sum_{v\in\mathcal{V}} \mu_{ij}x_{ij}^v + \sum_{i\in\mathcal{P}} F_i u_i \quad (1)$$

Subject to

$$\sum_{i\in\mathcal{P}}\sum_{j\in\mathcal{D}\cup\mathcal{T}} x_{ij}^v \le 1 \qquad \forall\, v \in \mathcal{V} \quad (2)$$

$$\sum_{v\in\mathcal{V}}\sum_{j\in\mathcal{D}\cup\mathcal{T}} x_{ij}^v + u_i = 1 \qquad \forall\, i \in \mathcal{P} \quad (3)$$

$$\sum_{j\in\mathcal{N}} x_{ij}^v - \sum_{j\in\mathcal{N}} x_{n+i,j}^v = 0 \qquad \forall\, i \in \mathcal{P},\ \forall\, v \in \mathcal{V} \quad (4)$$

$$\sum_{i\in\mathcal{N}} x_{ij}^v - \sum_{i\in\mathcal{N}} x_{ji}^v = 0 \qquad \forall\, j \in (\mathcal{N}\backslash\mathcal{C}),\ \forall\, v \in \mathcal{V} \quad (5)$$

$$\sum_{j\in\mathcal{N}} x_{o_v j}^v - \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{C}} x_{ij}^v = 0 \qquad \forall\, v \in \mathcal{V} \quad (6)$$

$$\sum_{j\in\mathcal{N}} x_{o_v j}^v \le 1 \qquad \forall\, v \in \mathcal{V} \quad (7)$$

$$\sum_{i\in\mathcal{C}/o_v}\sum_{j\in\mathcal{N}} x_{ij}^v + \sum_{j\in\mathcal{D}} x_{o_v j}^v + \sum_{i\in\mathcal{P}}\sum_{j\in\mathcal{C}} x_{ij}^v + \sum_{i\in\mathcal{D}}\sum_{j\in\mathcal{P}} x_{ij}^v = 0 \qquad \forall\, v \in \mathcal{V} \quad (8)$$

$$\sum_{i\in\lambda_w}\sum_{j\in\mathcal{T}\backslash\lambda_w} x_{ij}^v + \sum_{i\in\lambda_w}\sum_{\substack{j\in\lambda_w \\ j\le i}} x_{ij}^v = 0 \qquad \forall\, v \in \mathcal{V} \quad (9)$$

$$\sum_{\substack{j\in\lambda_w \\ j\le i}}\sum_{\substack{k\in\lambda_w \\ k>i}} x_{jk}^v \le Q \qquad \forall\, w \in W,\ \forall\, i \in \lambda_w, \forall\, v \in \mathcal{V} \quad (10)$$

$$\sum_{i\in\mathcal{D}\cup\mathcal{T}}\sum_{v\in\mathcal{V}} x_{ij}^v - \sum_{i\in\mathcal{P}\cup\mathcal{T}}\sum_{v\in\mathcal{V}} x_{ji}^v \le K - S_j \qquad \forall\, j \in C \quad (11)$$

$$\sum_{i\in\mathcal{N}\backslash\mathcal{T}}\sum_{j\in\mathcal{N}} \rho_{ij}x_{ij}^v + \sum_{i\in\mathcal{T}}\sum_{j\in\mathcal{N}\backslash\mathcal{T}} \rho_{ij}x_{ij}^v \le J \qquad \forall\, v \in \mathcal{V} \quad (12)$$

$$\alpha_j \ge \beta_i + \delta_{ij} + g_i - M\left(1 - x_{ij}^v\right) \qquad \forall\, (i,j) \in (\mathcal{N}\times\mathcal{N}), \forall\, v \in \mathcal{V} \quad (13)$$

where:

$$\alpha_i^v = \begin{cases} b_i, & i \in \mathcal{R} \\ p_i, & i \in \mathcal{T} \\ \bar{h}_i^v, & i \in \mathcal{C} \end{cases} \quad \beta_i^v = \begin{cases} b_i, & i \in \mathcal{R} \\ p_i, & i \in \mathcal{T} \\ h_i^v, & i \in \mathcal{C} \end{cases}$$

$$e_i \geq b_i \geq l_i \qquad \forall\, i \in \mathcal{R} \qquad (14)$$

$$b_{n+i} \geq b_i + \gamma_i + g_i \qquad \forall i \in \mathcal{P} \qquad (15)$$

$$x_{ij}^v \in \{0,1\} \qquad (i,j) \in N,\ \forall\, v \in \mathcal{V} \qquad (16)$$

$$u_i \in \{0,1\} \qquad \forall\, i \in \mathcal{P} \qquad (17)$$

$$b_i \geq 0 \qquad \forall\, i \in \mathcal{R} \qquad (18)$$

$$h_i^v \geq 0, \bar{h}_i^v \geq 0 \qquad \forall\, i \in \mathcal{C},\ \forall\, v \in \mathcal{V} \qquad (19)$$

The objective function (1) minimizes the overall cost, which consists of the travel costs of the robots and the penalty costs of unserved requests. Constraints (2) stipulate that each robot serves up to one request. Constraints (3) ensure that every request is served by a robot or outsourced. Constraints (4) ensure that the pickup and drop-off of a request are served by the same robot. Constraints (5) are flow conservation constraints, applied on all nodes except the depot nodes. Constraints (6) ensure that every robot that departs from its origin location returns to a depot node. Constraints (7) asserts that a robot can depart from their origin location at most once. Constraints (8)-(9) eliminate arcs that cannot be traveled directly by the robots (on their own or onboard a service line). Constraints (10)-(11) enforce the capacity limits of the transfer nodes and the depot nodes, respectively. Constraints (12) ensure that the total robot battery discharge does not exceed the battery capacity. Constraints (13) ensure that the time between the service start times at two nodes that are visited consecutively by the same robot, respect the service time and the direct travel time. Constraints (14) ensure the time windows of the request's nodes are respected. Constraints (15) ensure the pickup node is visited before the drop-off node. Finally, variable integrality and non-negativity are set in Constraints (16) -(19).

3.2. The path-based model

In what follows, we present the MILP formulation of the path-based model. The model explicitly represents the travel costs of the robots, the penalty costs for unserved requests as well as the capacity constraints of the transfer and depot nodes. A set of feasible paths for each robot-request pair is calculated in a preprocessing phase. The feasibility of a path is verified during this preprocessing phase, validating the departure time of the transfer nodes, time

windows of the request nodes and battery capacity. The following indices and parameters are used in the path-based model:

Indices:

p	Path
v	Robot
r	Request
c	Depot node
t	Transfer node

Parameters:

K	Capacity of each depot node
Q	Capacity of each transfer node
μ_p	Cost of path $p \in \Delta$
S_c	The initial number of robots in depot node $c \in \mathcal{C}$
Δ	Set of all paths
Δ_r	Set of all paths for request r $\in \mathcal{R}$
Δ_v	Set of all paths for robot v $\in \mathcal{V}$
Δ_c	Set of all paths that end in depot node $c \in \mathcal{C}$
$\Delta_{c'}$	Set of all paths that start in depot node $c' \in \mathcal{C}$
Δ_t	Set of all paths that use transfer node $t \in \mathcal{T}$

Decision Variables:

w_p	1 if path p $\in \Delta$ is in use, 0 otherwise

$$Minimize \sum_{p \in \Delta} w_p * \mu_p \quad (20)$$

Subject to

$$\sum_{p \in \Delta_r} w_p = 1 \qquad \forall\, r \in \mathcal{R} \quad (21)$$

$$\sum_{p \in \Delta_v} w_p = 1 \qquad \forall\, v \in \mathcal{V} \quad (22)$$

$$\sum_{p \in \Delta_t} w_p \leq Q \qquad \forall\, t \in \mathcal{T} \quad (23)$$

$$\sum_{p \in \Delta_c} w_p - \sum_{p \in \Delta_{c\prime}} w_p \leq K - S_c \qquad \forall\, c \in \mathcal{C} \quad (24)$$

$$w_p \in \{0,1\} \qquad \forall\, p \in \Delta \quad (25)$$

To simplify the formulation, we introduce two types of special paths that are included in the set Δ:

1. "Penalty paths" represent the cases where requests are not served by a robot, but instead are outsourced at a penalty cost.
2. "Dummy paths" represent the cases where robots are not in use and have zero costs.

The objective function (20) aims to minimize the overall cost, which consists of the travel costs for the robots and the penalty costs for unserved requests. Constraints (21) ensure that each request is assigned to a single path, either an actual path or an penalty path. Constraints (22) ensure that each robot is assigned to a single path, either an actual path or a dummy path. Constraints (23) ensure that the capacity of the transfer nodes is not exceeded. Constraints (24) ensure that the capacity of the depot nodes is not exceeded, by requiring that the number of paths ending at a depot minus the number of paths starting at a depot must be less than or equal to the depot capacity minus the number of robots initially assigned to the depot. Constraint (25) defines the decision variables as binary. In the column generation procedure, we use the LP relaxation of the problem, that is, constraints (25) are relaxed.

While the path-based formulation has a more compact structure, the number of paths that may be considered grows exponentially with the number of requests, thus this formulation can be solved directly by fully enumerating all potential paths only for very small instances of the problem. To handle medium to large instances we develop a column generation approach that will be discussed in following section.

4. Solution Approach

In the following subsections we present the main components of the column generation framework we have devised in order to solve the operational problem in the public transit enhanced AMR service. In Section 3.1, we formulate the dual of the linear relaxation of the path-based model. The pricing problem is to identify primal non-basic columns with negative reduced costs. This is equivalent to identifying constraints that are violated by the solution of the restricted path-model. As in most column generation approaches for VRP's this is essentially a Resource Constrained Shortest Path (RCSP) problem (Irnich and Desaulniers, 2015; Pugliese and Guerriero, 2013). In Section 3.2, we develop a four-stage dynamic programming algorithm to solve it.

An overview of the framework is presented in Figure 1. We initialize the restricted set of paths and solve the path-based model. Then, in order to identify new columns, we solve for each robot-request pair RCSP problem. Notably, the sub-problems of different robot-request pairs are independent and can be solved simultaneously. If new columns are identified, we add them to the restricted set and re-solve the master problem. This process continues iteratively until no new improving paths can be found. In cases where the resulting solution is non-integral, a feasible integral solution is obtained by directly solving problem (20)-(25) with the path set generated by applying the column generation.

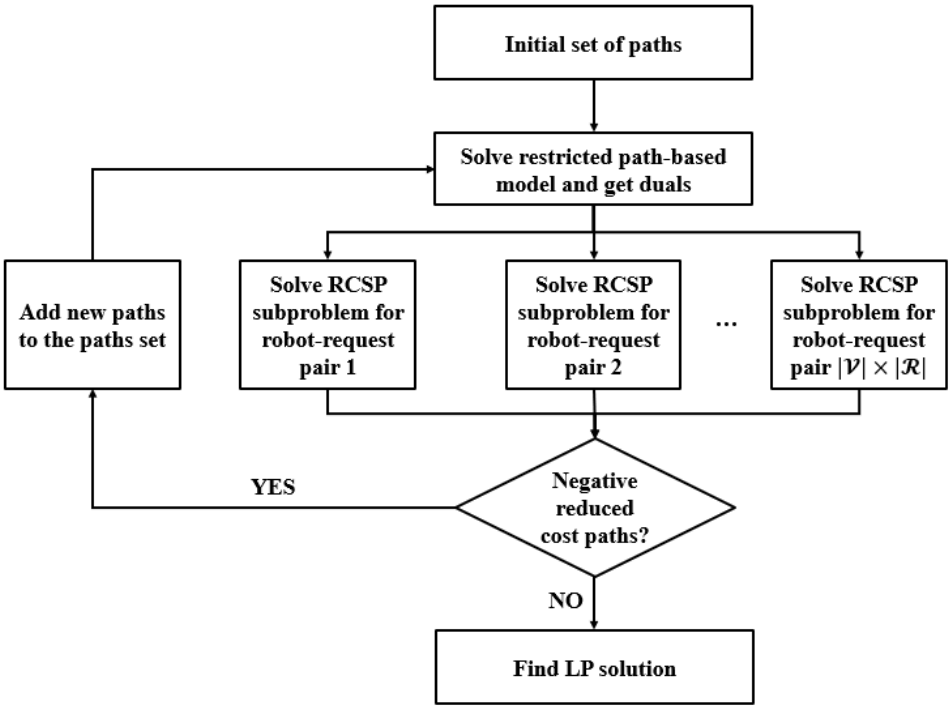


Figure 1: Flow chart of the column generation procedure

4.1. The path-based dual model

The dual model of the restricted LP relaxation version of the path-based problem is defined as follows:

Parameters:

$r(p)$ The request associated with path $p \in \Delta$

$v(p)$ The robot associated with path $p \in \Delta$

$c(p)$ The start depo associated with path $p \in \Delta$

$c'(p)$ The end depo associated with path $p \in \Delta$

θ_p The set of transfer nodes associated with path $p \in \Delta$

Decision Variables:

x_r The dual variables associated with Constraints (21)

y_v The dual variables associated with Constraints (22)

z_t The dual variables associated with Constraints (23)

u_c The dual variables associated with Constraints (24)

$$Maximize \sum_{r \in R} x_r + \sum_{v \in V} y_v - Q * \sum_{t \in \mathcal{T}} z_t - \sum_{c \in C} (K - s_c) * u_c \quad (26)$$

Subject to

$$x_{r(p)} + y_{v(p)} - \sum_{t \in \theta_p} z_t + u_{c(p)} - u_{c'(p)} \leq \mu_p \qquad \forall\, p \in \Delta \quad (27)$$

$$x_r \ free \qquad \forall\, r \in \mathcal{R} \quad (28)$$

$$y_v \ free \qquad \forall\, v \in \mathcal{V} \quad (29)$$

$$zt \leq 0 \qquad \forall\, t \in \mathcal{T} \quad (30)$$

$$uc \leq 0 \qquad \forall\, c \in \mathcal{C} \quad (31)$$

The decisions variables of the dual problem (26)-(31) represent the request, robot, transfer nodes and the depot nodes of a considered path in the restricted primal problem. The variable x_r (28) represents Constraints (21) in the primal problem, which ensure that each service request is served. The variable y_v represents Constraints (22) in the primal problem, which

ensure that each robot is used on only a single path, either an actual path or a dummy path. The variable z_t represent Constraints (23) in the primal problem that ensure that the capacity of the transfer nodes is not exceeded. If z_t has a negative value, it indicates that transfer node t has reached its capacity limit. Similarly, the variable u_c represents Constraints (24) that ensure the capacity of the depot nodes is respected. A negative value of u_c indicates that the amount of robots at depot c at the end of the planning horizon equals to the capacity of the depot. The values of z_t and u_c reflect the penalty incurred when resources are consumed. Considering a given robot-request pair (v, r), we seek to identify a path p that violates constraint (27), that is, a path that satisfies the following:

$$\mu_p - (\bar{x}_{r(p)} + \bar{y}_{v(p)} - \sum_{t \in \theta_p} \bar{z}_t + \bar{u}_{c(p)} - \bar{u}_{c'(p)}) < 0 \tag{32}$$

Noting that the values of the variables y_v, x_r and u_c are fixed for all paths of the robot-request pair (v, r) the objective of the RCSP sub-problem is to find a path p that minimizes the following expression:

$$\mu_p - (- \sum_{t \in \theta_p} \bar{z}_t - \bar{u}_{c'(p)})$$

4.2. The shortest path sub-problem with resource constraints

We formulate the RCSP sub-problem as a four-step dynamic program to account for limitations related to battery usage and the time windows of request nodes. Recall that we assume that a robot can serve at most one request during the planning horizon. As a result, the problem can be decomposed to four main decision steps, as illustrated in Figure 2: the path from the initial depot to the pickup point, the path from the pickup point to the drop-off point, the path from the drop-off point to the depot, and the depot where the robot should return for recharging. Note that the first decision steps reduce to deciding if public transit is to be used, and if so, to selecting the start and end transfer nodes. We denote the set of transfer node pairs that may be selected by $\mathcal{B}$, and a pair $(i, j) \in \mathcal{B}$ is denoted as arc A, for shortness of the presentation. The fourth decision is restricted to selecting a return depot from the set of depots $\mathcal{C}$. A state of the robot is denoted by the quartet (S, T, B, N), representing the decision step S, the current time T, the current battery state B, and the current node N. Specifically, the decision step S takes

one of the following values: 1 – at the origin depot, 2 – at pick-up location, 3 – at drop-off location, 4 – selecting the return depot.

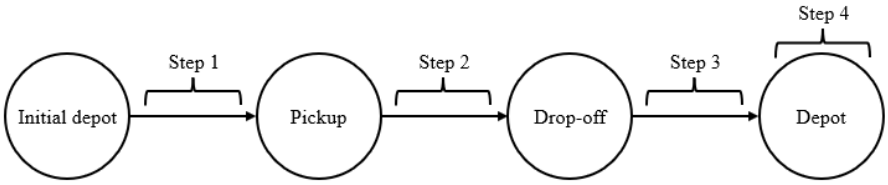


Figure 2: The four-step path diagram

Given that the robot is in state (S, T, B, N), the function $F(S, T, B, N)$ represents the minimal path cost from the node of step S to a return depot. This function can be defined recursively using the following Bellman equation:

$$F(S,T,B,N) = \begin{cases} \min\limits_{A \in \mathcal{B}} \left[\theta(\mathrm{S},\mathrm{A}) + \mathrm{F}\left(\mathrm{S}+1,\ \sigma(S,A,T), \epsilon(S,A,B), \rho(S,A)\right)\right], & S \in \{1,2,3\} \\ \min\limits_{\mathrm{c} \in \mathcal{C}} [\mu_{N,c} + \mathrm{F}(\mathrm{S}+1,\ \mathrm{T} + \tau_{N,c}, B\ -\ \beta_{N,c}, \mathrm{c})], & S \in \{4\} \end{cases}$$

Where θ represent the cost component, σ is a time function, ϵ is the energy function, and ρ is the node function. The travel cost, travel time, and battery consumed while the robot travels from node N to return depot c, are denoted by $\mu_{N,c}$, $\tau_{N,c}$, $\beta_{N,c}$, respectively.

An optimal solution of the sub-problem is obtained by calculating the cost $F(1,\ 0,\ J, o_v)$, which represents the shortest path of the robot departing from origin depot o_v with a battery state J at the beginning of the planning period. An end condition represents the cost at the return depot:

$$F(5,T,B,N) = 0 \quad \forall T, \forall B \geq 0, \forall N$$

In addition, the following boundary conditions are applied to prevent exploration of infeasible solutions:

$$F(S,T,B,N) = \infty \quad \forall S \in \{1,2,3,4\}, \forall B < 0, \forall T, \forall N$$

To ensure that the battery of the robot is not exceeded.

$$F(S,T,B,N) = \infty \quad \forall S \in \{2,3\}, \forall B, \forall T > l_N, \forall N$$

To ensure that the time windows of request nodes at steps 2 and 3 are respected.

During each step in the program, there are two options available to the robot to move to the next step node. The first option is a direct movement from node S to node $S+1$, which involves the robot traveling one leg using its battery power without the use of public transit. The second

option involves the use of public transit for part of the travel. In this case, the travel consists of three legs: 1) the robot uses its battery power to travel from node S to the boarding transfer node, 2) a public transport leg that connects two transfer nodes of the same service line, and 3) the robot uses its battery power to travel from the disembarking transfer node to node $S+1$. It is important to note that in each step, we assume that only one transit line is used.

Each step of the algorithm requires calculating the cost accumulated, the time elapsed and the battery used. All of these can be presented as linear functions of the problem input and the dual variable values associated with the transfer nodes and the return depots. Four nodes are considered: i, j, k and l, with i being the node at step S, (j,k) representing a pair of public transit nodes within the set $\mathcal{B}$, and l being the node at step $S+1$. As illustrated in Figure 3, the leg $i \to l$ represents a direct robot movement, whereas the leg $i \to j \to k \to l$ represents the use of public transit for part of the journey between i and l. In what follows, let C, P, and D denote the indices of the initial depot node, the pickup node, and the drop-off node, respectively. We next define the functions used in the Bellman equation.

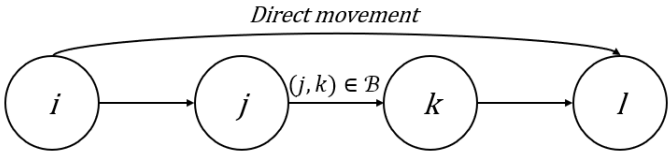


Figure 3: Movement options in the DP algorithm

The cost function $\boldsymbol{\theta(S, A)}$:

The function determines the costs of traveling from the node of step S using public transit arc A to the node of step $S+1$. We differentiate between the case of a direct robot movement and movement using public transit. The costs for each step of the dynamic program are presented in Table 1.

Table 1: The cost function values per state

S	Direct movement $A = 0$	Movement using public transit $A = (j, k)$
1	$\mu_{CP} + \bar{u}_C$	$(\mu_{Cj} + \mu_{jk} + \mu_{kP}) - \sum_{t \in (j,k)} \bar{z}_t$
2	μ_{PD}	$(\mu_{Pj} + \mu_{jk} + \mu_{kD}) - \sum_{t \in (j,k)} \bar{z}_t$
3	0	$(\mu_{Dj} + \mu_{jk}) - \sum_{t \in (j,k)} \bar{z}_t$
4	$\mu_{Dl} - \bar{u}_l$	$\mu_{kl} - \bar{u}_l$

Step 1 involves selecting the path from the initial depot to the pickup point. If the robot moves directly from the depot node C to the pickup node P, then the cost will be the actual travel cost which is represented by μ_{CP}, plus the dual price of depot node C, u_C. However, if the robot uses a public transit arc (j,k), then the actual travel cost will be the sum of three legs: from C to j, from j to k and from k to P. In this case, the cost will also include the dual price of depot node C, u_C, and the transfer nodes included in arc (j,k), which is $-\sum_{t\in(j,k)} z_t$.

In Step 2, the decision is made to select the path from the pickup point to the drop-off point. If the robot moves directly from the pickup node P to the drop-off node D, then the cost will be the actual travel cost represented by μ_{PD}. There is no dual price associated with these nodes. However, if the robot uses a public transit arc (j,k), then the actual travel cost will be the sum of the three legs plus the dual price of the transfer nodes included in arc (j,k).

Step 3 involves selecting the public transit arc to use before returning to the end depot. The robot has the option of not using public transit, in which case the cost is zero. However, if the robot uses public transit, the cost will include the actual travel cost from the drop-off node D to the transfer node j, plus the public transit cost of arc (j,k) represented by $\mu_{Dj} + \mu_{jk}$. Additionally, the cost will include the dual price of the transfer nodes included in the path.

Finally, in Step 4, the decision is made to select the depot node to which the robot should return for recharging. The cost of this route will be the travel cost from the current node to node l, being the end depot the robot returns to, minus the dual price of depot node l, which is $-u_l$. In case public transit is not used in Step 3, the current node is drop-off node D, and the travel cost is μ_{Dl}. Otherwise, the current node is transfer node k and the travel cost is μ_{kl}.

The time function $\sigma(S, A, T)$:

The time function value for the four steps, considering direct movement or movements using public transit, are presented in Table 2. The function determines the time the robot arrives at the node of step $S+1$ given that it departs at time T from the node of step S and uses public transit arc A. σ represents both options of movements and consider the departure time of the service line from transfer node j, p_j. In case a feasible path cannot be found within the given departure time, the function returns an infinite value.

Table 2: The time function values per state

S	Direct movement $A = 0$	Movement using public transit $A = (j, k)$
1	$max(T + \delta_{CP}, e_P) + g_P$	$\begin{cases} max(p_j + \delta_{jk} + \delta_{kP}, e_P) + g_P, & T + \delta_{Cj} \le p_j \\ \infty, & T + \delta_{Cj} > p_j \end{cases}$
2	$max(T + \delta_{PD}, e_D) + g_D$	$\begin{cases} max(p_j + \delta_{jk} + \delta_{kD}, e_D) + g_D, & T + \delta_{Pj} \le p_j \\ \infty, & T + \delta_{Pj} > p_j \end{cases}$
3	T	$\begin{cases} p_j + \delta_{jk} + g_k, & T + \delta_{Dj} \le p_j \\ \infty, & T + \delta_{Dj} > p_j \end{cases}$
4	$T + \mu_{Dl}$	$T + \mu_{kl}$

Step 1 involves determining the updated time for a robot to pick up a parcel. This is done by comparing the time it takes to travel to the pickup point with the earliest time window specified for that pickup. The updated time is set to the later of these two times, as the robot must wait for the beginning of the time window before attempting to pick up the parcel. If the robot moves directly from one point to another, the updated time is the current time T plus the travel time from node C to node P, which is represented as δ_{CP}. However, if public transit is used, the updated travel time is calculated by adding the departure time of transfer node j with the travel time from node j to node k, and from node k to pickup node P. In both cases, the service time at node l is added to the updated time. In case of the robot arrive to transfer node J after the departure time, the function return ∞. Step 2 follows a similar logic, but with node l being the drop-off node D. In step 3, if public transit is not used, the time remains unchanged. If public transit is used, the updated time is the departure time of transfer node j plus the travel time from node j to node k and the service time at node k. The departure time of node j must be respected in this step as well. Finally, in Step 4, the updated time is calculated by adding the travel time from the current node to node l, to the current time T.

The energy function $\epsilon(S, A, B)$:

The function is designed to determine the battery state of a robot that departs from the node of step S with battery state B and uses public transit arc A. The function considers both options of movements and reduces the battery that is consumed in the current state from B. This is achieved by reducing the discharge amount of non-public transit arcs from B. The energy function value for the four steps, considering direct movement or movements using public transit, are presented in Table 3.

Table 3: The energy function values per state

S	Direct movement $A = 0$	Movement using public transit $A = (j, k)$
1	$B - \rho_{CP}$	$B - (\rho_{Cj} + \rho_{kP})$
2	$B - \rho_{PD}$	$B - (\rho_{Pj} + \rho_{kD})$
3	B	$B - \rho_{jk}$
4	$B - \rho_{Dl}$	$B - \rho_{kl}$

The node function $\rho(S, A)$:

The function is designed to return the node of step $S + 1$, considering the use of public transit arc A at step S. The node function value for the four steps, considering direct movement or movements using public transit, are presented in Table 4.

Table 4: The node function values per state

S	Direct movement $A = 0$	Movement using public transit $A = (j, k)$
1	P	P
2	D	D
3	D	k
4	l	l

If the current step is 1, the node at step 2 is the pickup node P. If the current step is 2, the node at step 3 is the drop-off node D. In step 3, the node at step 4 depends on the use of public transit. If public transit is not in use, the node at step 4 is the drop-off node D, as no movement has occurred. However, if public transit is in use, the node at step 4 is the transfer node where the robot exited the public transit, which is represented by node k of the selected arc $(j, k) \in \mathcal{B}$., the node at the end of the program is l, the return depot node of the robot.

As illustrated in Figure 1, each column generation iteration requires solving a total of $|\mathcal{V}| \times |\mathcal{R}|$ sub-problems. However, it's important to note that when two or more robots commence their journeys from the same initial depot, the resource-constrained shortest path problem is fundamentally identical for all these robots. To enhance the efficiency of the column generation procedure, we have applied symmetry reduction considerations to reduce the number of resource-constrained shortest path problems to be solved to $|\mathcal{C}| \times |\mathcal{R}|$. The outcomes of this optimization process are elaborated upon in Section 4.2.

5. Numerical Experiments

In this section, we discuss in detail the design of our numerical experiment, as well as the obtained results. The primary objective of our study was to examine the performance of our solution approach and showcase the impact of enhancing AMR services with public transit. For this purpose, we have generated both synthetic problem instances as well as case study instances based on data from Tel Aviv. In the following sections, we will elaborate on the specifics of the data sets, the parameters employed in our model, and the evaluation metrics utilized to assess the effectiveness of the proposed method.

5.1. Synthetic problem instances

We have created a random instance generator, that in addition to the model parameters described in Section 2, uses the following auxiliary parameters:

- NS: Sets the dimensions of the network area.
- H: the planning horizon.
- TW: Defines the time window size of the request nodes.

The network area is defined as a square with a range of $[0, NS]$, and all nodes' locations are located within this area. The sets of robots, depots, requests, service lines, and transfers, are generated as follows. Each robot's depot origin location is randomly sampled from the depot nodes set with respect to the depot capacity parameter. Transfer nodes are distributed among the service lines such that each service line has a size of $\left\lceil \frac{|\mathcal{T}|}{|\mathcal{W}|} \right\rceil$, except for one service line that has a size of $\left\lceil \frac{|\mathcal{T}|}{|\mathcal{W}|} \right\rceil \% \mathcal{T}$. Depot and request nodes are randomly located within the network.

For each service line, the transfer node locations are generated as follows: the first and last node are randomized, ensuring that there is a minimum distance of $0.6 \cdot NS$ between them. The remaining transfer nodes associated with the service line are evenly split on the line between these two nodes. The departure time of the first transfer node is randomized, and the departure times of the rest of the transfer nodes are calculated based on their service time and travel time from the previous transfer node. If the departure time of the last node in the line exceeds the planning horizon, the entire service line schedule is shifted back to ensure it is feasible.

We generate the request time windows using the following process. Let τ be the center of the time window of pick-up node i, we randomly draw it from the uniform distribution $\tau \sim U[TW, \frac{H}{4}]$, and set $e_i = \tau - TW$, $l_i = \tau + TW$. Similarly, let τ' be the center of the time

window of drop-off node $n+i$, we randomly draw it from the following uniform distribution τ'~$[\tau + g_i + \delta_{i,n+i}, 0.9H]$ and set $e_{n+i} = \tau' - TW$, $l_{n+i} = \tau' + TW$.

The travel times of the robots between any pair of nodes (δ_{ij}) is set to the Euclidean distance between the two nodes. For simplicity, we set the discharge rate to be one. That is, for every time unit a robot travels, it discharges one unit of battery. For public transit arcs we further multiply the Euclidian distance by a fixed factor η to represent a different average speed of a public transit vehicles. The cost is calculated as the product of the travel time and a cost factor ε, which is a parameter input of the instance. In the case of public transit arcs, the cost is multiplied by a public transit cost factor ψ, which is a number between 0 and 1, reflecting the lower cost of public transit compared to regular robot movement:

$$\mu_{ij} = \begin{cases} \delta_{ij} * \varepsilon, & (i,j) \in (\mathcal{N} \backslash \mathcal{T}) \\ \delta_{ij} * \varepsilon * \psi, & (i,j) \in \mathcal{T} \end{cases}$$

The parameter values we have used to in order to generate the synthetic problem instances are presented in Table 5.

Table 5: Random instances parameters and values

Parameter	Description	Range of values
NS	the size of the network	15-400
H	the planning horizon of the instance	1500-40000
n	number of service request	2-150
$\mathcal{V}$	set of robots	2-150
$\mathcal{C}$	set depot nodes	1-70
$\mathcal{W}$	set of public transit service lines	1-12
$\mathcal{T}$	set of transfer nodes	3-34
J	robots maximum travel time	10-1000
K	capacitiy on depot node. $K \geq \left\lceil \frac{\lvert\mathcal{V}\rvert}{\lvert\mathcal{C}\rvert} \right\rceil$	2-11
g_i	service time for all node $i \in \mathcal{N}$	2
TW	half length of the time window	200
F_i	penalty cost for unsreved request	1-10000
Q	capacitiy on transfer node	0-10
ε	cost factor	5
ψ	public transit cost factor	0-1
η	public transit travel time factor	1

5.2. Tel Aviv problem instances

To evaluate the efficiency of our proposed method in a real-world setting, we generated a new set of problem instances using the road, sidewalks, and public transit networks of Tel Aviv. Our study focused on a specific sub region of the city, measuring four square kilometres in size. The sub region is composed of ten neighbourhoods defined by Tel Aviv - Yafo Municipality. The city's GIS Portal (gisn.tel-aviv.gov.il) provided data for neighbourhood boundaries and bus stops. We calculated the distance and travel time over each network arc using OpenRouteService (https://github.com/GIScience/openrouteservice). The arc costs and battery discharge were calculated using the same approach as for the synthetic problem instances.

Requests pickup and drop-off locations were randomly selected from passenger trips completed by Tel Aviv's ride-pooling service (Bubble) between April 15, 2019, and September 30, 2020. We collected data on four public transit lines, encompassing a total of 47 stations, and 300 service requests. We randomly created the time window centers of the service request nodes to make them coincide with public transit schedules. The time window widths of the request nodes were set to 2400 seconds, which is typical for food and office equipment deliveries. We also picked locations randomly for 30 depot stations, with some coinciding with public transit stations. The initial distribution of the robots to depots was selected randomly. A map of the studied area is presented in Figure 4. From these data sets, we created five instances, each varying in the number of robots, requests, public transit lines, and depot stations. The name of each instance follows the following convention: <number of robots>_<number of requests>_<number of depots_number of public transit lines>_<number of public transit stations>, to enhance clarity in the subsequent tables and figures, we abbreviate the convention to tlv_<number of requests>

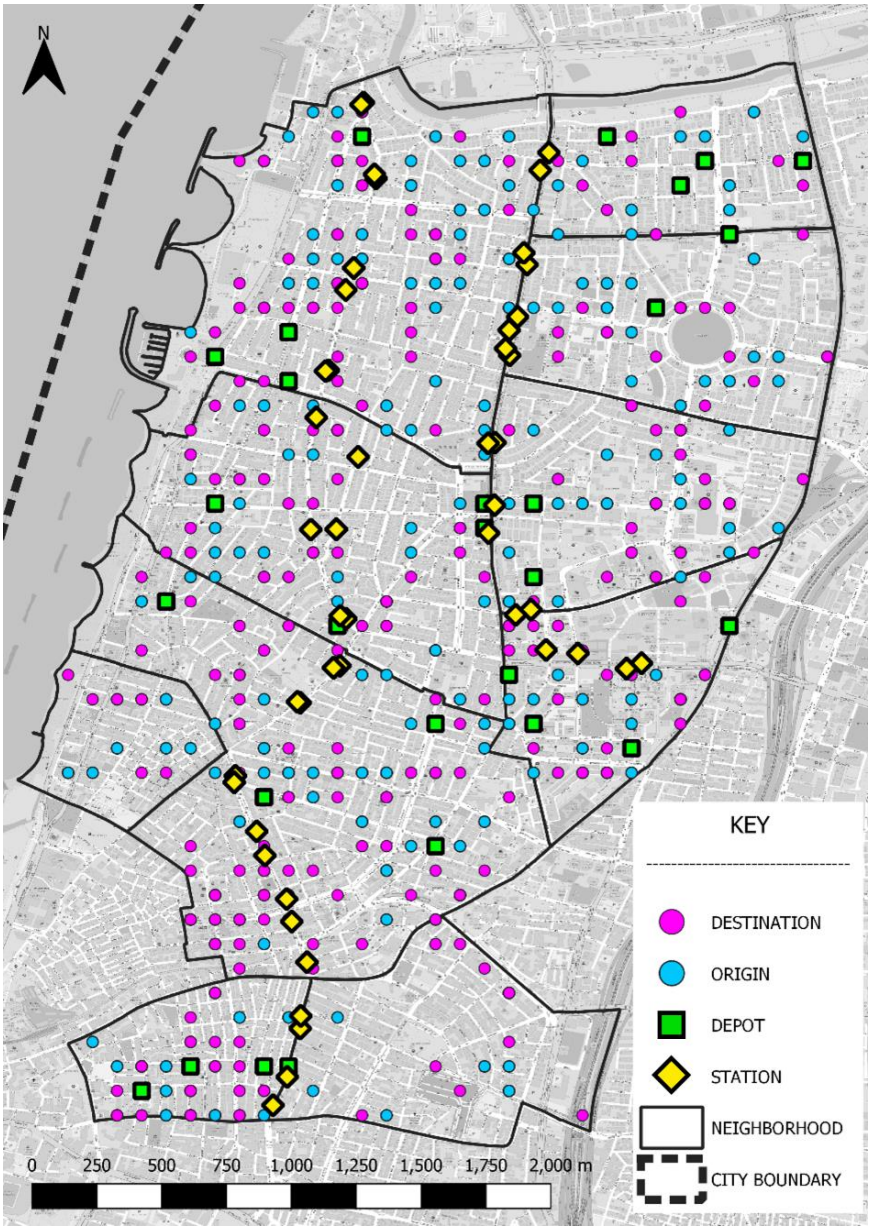


Figure 4: Map of the Tel Aviv sub region by OpenStreetMap

Table 6 provides the parameters of each instance, which were chosen to ensure that each instance was both realistic and feasible. Some parameters were fixed across instances, while others varied. The number of requests varied from 100 to 300 in increments of 50, to test the column generation approach across a range of problem instance sizes. The robot's supply size was kept identical to the number of requests so as to assess the number of requests that could be served by robots. The number of depots increased with the number of robots, and the depot capacity was set to $\left\lceil \frac{|\mathcal{T}|}{|\mathcal{W}|} \right\rceil + 1$ to allow some flexibility in robot routes while keeping the depot capacities limited. The planning horizon was set as the latest departure time of the public transit plus 1500 seconds to ensure that all public transit stations were feasible for all requests. The

penalty cost was set high at 40000 to encourage using robots over outsourcing, except when no other option was available. The battery capacity was set to 3600, representing one hour of robot operation. The capacity in transfer nodes was set to 8 to ensure that public transit could be used with realistic resources. The public transit cost factor was set to zero to test the effectiveness of public transit usage in an extreme case.

Table 6: Tel Aviv case study instances

Parameter	**Instance**				
	tlv_100	tlv_150	tlv_200	tlv_250	tlv_300
NS	NA	NA	NA	NA	NA
H	6660	6660	6660	6660	6660
n	100	150	200	250	300
$\mathcal{V}$	100	150	200	250	300
$\mathcal{C}$	20	25	30	30	30
$\mathcal{W}$	3	4	4	4	4
$\mathcal{T}$	33	47	47	47	47
J	3600	3600	3600	3600	3600
K	6	7	8	10	11
g_i	2	2	2	2	2
TW	1200	1200	1200	1200	1200
F_i	40000	40000	40000	40000	40000
Q	8	8	8	8	8
ε	5	5	5	5	5
ψ	0	0	0	0	0

The complete set of instances used in this numerical experiment and the results is available online at: https://github.com/Yishay-S/AMRsPy.

5.3. Experimental setup

The experiments were carried out on a Windows server with a 64-bit operating system, two Intel Xeon Gold 6230 CPU@2.10GHz, each with 20 physical cores (40 logical), and 128 GB of usable RAM. We utilized IBM CPLEX 12.10.0.0 solver for the linear and mixed integer programming tasks, while the random instances creation and column generation procedure, including the DP algorithm for the sub-problem, were implemented using Python 3.7.4. Different parallel computing settings were used in the synthetic and Tel Aviv problem instances for the column generation method in sub-problem solving. The synthetic problem instances varied more in size, including small instances, so we used 8 CPU cores, which was beneficial

for both small and large numbers of cores. For the Tel Aviv case study, which involved instances of 100 requests and above, we used all available resources and employed 80 CPU cores for parallel computing.

5.4. Results

We tested our approach on seven-hundred synthetic problem instances we have randomly generated with varying numbers of requests, robots, and public transit nodes. Figure 5 illustrates the average solving time of the arc-based model and column generation approach for problem instances consisting of two to 24 requests. Specifically, the arc-based model was solved using CPLEX with a time limit of one hour. It exceeded the time limit for instances with more than 12 requests. The column generation approach produced optimal solutions for these instances in just a few seconds.

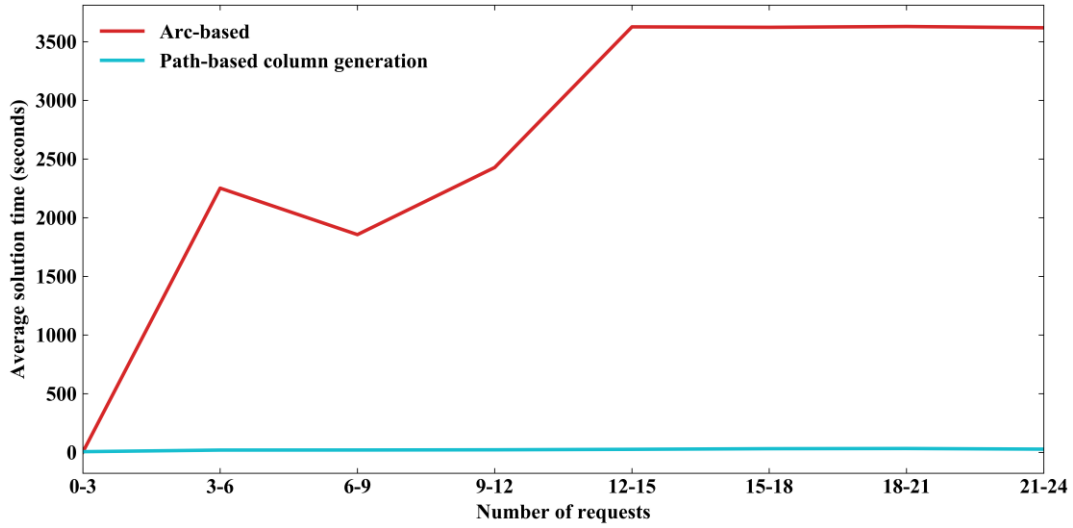


Figure 5: Arc based model vs. column generation approach

In the implementation of the column generation framework, we have applied symmetry reduction considerations to reduce the number of resource-constrained shortest path problems to be solved. In addition, we applied parallel computing and evaluated the impact of the number of sub-problems solved simultaneously on the overall running time of the column generation framework. Figure 6 displays the average solving time of the column generation approach for varying instance sizes, with up to 150 requests. We compare three versions of the algorithm, the baseline straightforward implementation of the model, a version that applies symmetry reduction and a third version that applies symmetry reduction and utilizes 8 CPU's to simultaneously solve the sub-problems. As can be observed, reducing symmetry, and utilizing

parallel computing have significantly improved the performance of the column generation approach.

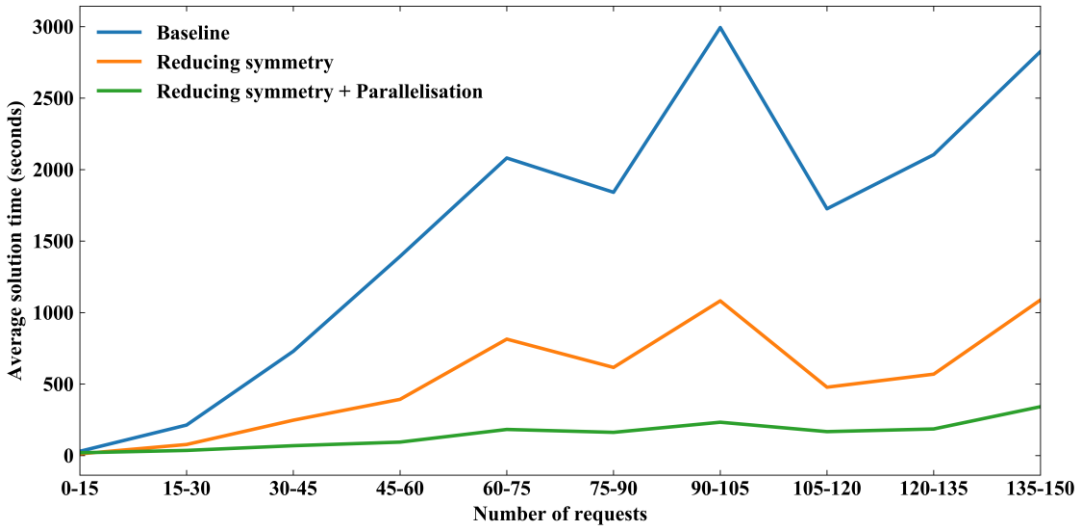


Figure 6: Performance of the column generation versions

Recall that the column generation framework solves the linear relaxation of the path based model. For the synthetic problem instances, less than 2% of the obtained solutions were non-integral. For these specific instances, we have generated a feasible integral solution by directly solving problem (20)-(25) with the path set obtained by applying the column generation. An average optimality gap of 0.13% was obtained for these instances. This demonstrates that for any practical purpose the proposed column generation approach can be used to obtain high quality solutions in reasonable computing time. Lastly, we have measured the extent of the public transit usage in the obtained solutions. In the tested instances, we observed cases in which none of the requests were served using public transit and up to cases in which all of the requests were served using public transit, for parts of the robot paths. This depends on the structure of the network, the available public transit lines, and the relative cost of using these lines. As can be expected, we observed that as the cost of public transit decreases, its usage increases.

For the Tel Aviv problem instances, we ran the model twice for each instance, once enabling the option of using public transit and once when this option is disabled. We analyzed various metrics, including the overall cost of the service, the number of requests that could not be served by robots and were thus outsourced, and the average energy consumption of the served requests. The results of each run of the algorithm are displayed in Table 7.

Table 7: Tel Aviv case study results

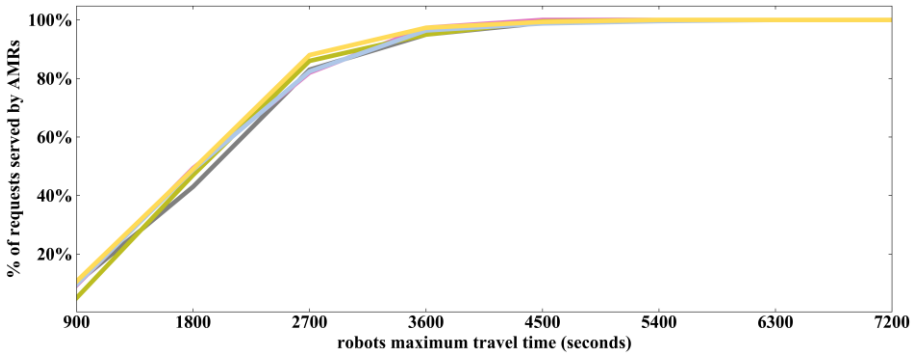
Instance	Public Transit	Total cost	% served by AMRs	% used public transit	Average AMR battery consumption	Solving time (seconds)
tlv_100	No	1,400,630	90.00%	0%	2,224	300.54
	Yes	1,148,054	95.00%	48%	1,996	648.71
tlv_150	No	2,110,788	90.00%	0%	2,238	547.79
	Yes	1,707,525	97.33%	47%	2,120	1,307.71
tlv_200	No	2,928,079	89.00%	0%	2,301	456.64
	Yes	2,449,669	95.00%	40%	2,158	1,758.08
tlv_250	No	3,551,012	88.80%	0%	2,190	443.80
	Yes	2,926,509	96.40%	35%	2,130	2,880.45
tlv_300	No	4,147,365	89.67%	0%	2,162	827.98
	Yes	3,457,048	97.33%	29%	2,149	4,146.44

In instance tlv_200, the column generation approach provides a suboptimal solution with a relative gap of 0.001%, which is negligible in terms of the quality of the service and the overall cost, for the other four instances, the model provides an optimal solution.

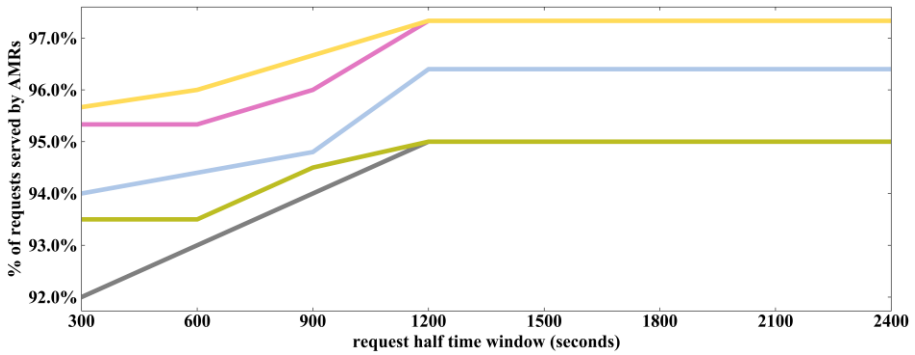
The percentage of service requests that were served using public transit for parts of the journey provides an insight into the possibility of incorporating public transit into AMR's delivery services. On average, 39.8% of requests served by AMRs involved the utilization of public transit, indicating the practicality and viability of integrating public transit within the AMR service. When the use of public transit is enabled, the average number of outsourced requests decrease by 6.84% and the average robot battery consumption reduces.

In all examined cases, some requests that cannot be fulfilled by the AMR service and therefore need to be outsourced. One possible reason is the inability of the considered public transit system to handle a large volume of requests simultaneously. Currently, the public transit system has a limit of 8 requests, which is insufficient to accommodate the high number of requests ranging from 100 to 300. Another factor that could cause the infeasibility of the robot service is the maximum travel time allowed for the robot, which is currently set to one hour. As can be observed in Figure 4, some request nodes are located in remote areas of the studied region, requiring long routes that exceed the robot's battery capacity. Lastly, the narrow time windows assigned to the requests, currently set to 1200 seconds, may also contribute to the infeasibility as they might be too restrictive for some robots.

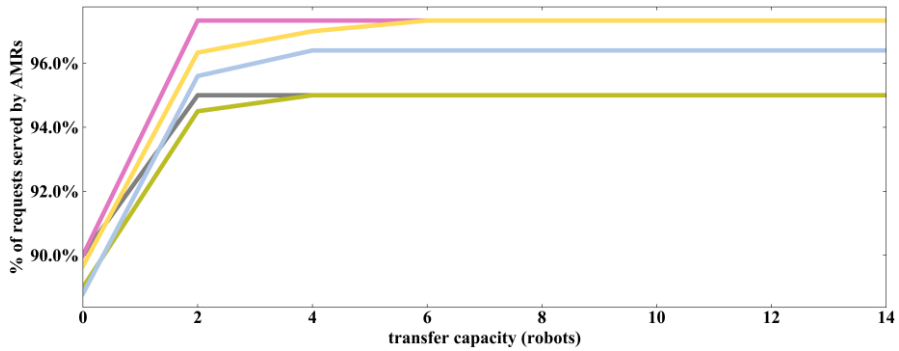
To investigate the underlying causes of the infeasibility, we conducted a sensitivity analysis on three input parameters: transfer capacity, AMR's battery capacity, and request half time window. We tested various values for each parameter, while all other parameters remain unchanged. Ultimately, we measured the percentage of requests that the AMR service was able to fulfill, as shown in Figure 7.



(a) AMRs usage by robot maximum travel time



(b) AMRs usage by request half time window



(c) AMRs usage by transfer capacity

Figure 7: AMRs usage by different parameters settings

In each of the sub graphs presented, it is evident that there exists a point where the utilization of the AMR's service reaches its maximum. Figure 7(a) clearly illustrates this trend, where the AMR's maximum travel time is set at 5,400 seconds. However, it can be observed that a significant portion of the utilization is already achieved with a more practical limit of 3,600 seconds. This aligns with the specifications of AMRs used in various industries. Furthermore, the parameter related to the request half time window presented in Figure 7(b) proves to be restrictive up to a time limit of 2,400 seconds, which represents a time window of 40 minutes. This duration remains valid and suitable for numerous delivery scenarios, including fast-food delivery and e-commerce. Finally, as can be observed in Figure 7(c), the transfer capacity constraint becomes non-binding with a maximum value of 6 robots. This indicates the maximal number of robots that is simultaneously onboard a public transit vehicle in the delivery service. In conclusion, all three parameters demonstrate their effectiveness through the utilization of realistic and reasonable values. These values are not only feasible but also applicable in real-world usage scenarios.

6. Conclusions and future directions

This study examines the potential of using public transit to enhance AMRs delivery services. It addresses this problem through the development of two mixed integer programming formulations, an arc-based formulation, and a path-based formulation. A column generation framework developed using the path-based model is capable of handling large problem instances within reasonable processing times. The results of the study indicate that using public transit in conjunction with AMRs can significantly increase the service range and reduce energy consumption. The study highlights the potential of AMRs in urban logistics and the need to further explore the integration of AMRs with other forms of transportation.

We see several directions for further research. First, the models developed in this study can be extended to consider recharging activities during the planning horizon, enabling the AMR to serve multiple requests sequentially. Second, the dynamic version of the problem, in which service requests appear on-line should be examined. The static models and column generation approach developed in this study may be utilized in a rolling-horizon framework designed for this purpose. Lastly, the integration of AMRs and public transit can be evaluated within a truck-and-robot concept. Adding another mobility layer to this approach may further increase the flexibility and efficiency of this last-mile delivery concept.

Acknowledgements

We are grateful for the generous Excellence Scholarships awarded to Yishay Shapira during his studies by The Israeli Smart Transportation Research Center (ISTRC) and the Shlomo Shmeltzer Institute for Smart Transportation.

References

Alfandari, L., Ljubić, I., & da Silva, M. D. M. (2022). A tailored Benders decomposition approach for last-mile delivery with autonomous robots. European Journal of Operational Research, 299(2), 510-525.

Allen J, Piecyk M, Cherrett T, Juhari MN, McLeod F, Piotrowska M, Bates O, Bektas T, Cheliotis K, Friday A, et al., 2021. Understanding the transport and co2 impacts of on-demand meal deliveries: A london case study. *Cities* 108:102973.

Amazon Zoox, 2022. *What's next for Amazon Scout?*. URL https://www.aboutamazon.com/news/transportation/whats-next-for-amazon-scout, accessed April 30, 2023.

Baldacci, R., Bartolini, E., & Mingozzi, A. (2011). An exact algorithm for the pickup and delivery problem with time windows. *Operations research*, *59*(2), 414-426.

Benarbia, T., & Kyamakya, K. (2021). A literature review of drone-based package delivery logistics systems and their implementation feasibility. *Sustainability*, *14*(1), 360.

Berbeglia G, Cordeau JF, Gribkovskaia I, Laporte G, 2007. Static pickup and delivery problems: a classification scheme and survey. *Top* 15(1):1–31.

Berbeglia, G., Cordeau, J. F., & Laporte, G. (2010). Dynamic pickup and delivery problems. *European journal of operational research*, *202*(1), 8-15.

Boysen N, Fedtke S, Schwerdfeger S, 2021. Last-mile delivery concepts: a survey from an operational research perspective. *Or Spectrum* 43(1):1–58.

Boysen, N., Schwerdfeger, S., & Weidinger, F. (2018). Scheduling last-mile deliveries with truck-based autonomous robots. *European Journal of Operational Research*, *271*(3), 1085-1099.

Brouer, B. D., Alvarez, J. F., Plum, C. E., Pisinger, D., & Sigurd, M. M. (2014). A base integer programming model and benchmark suite for liner-shipping network design. *Transportation Science*, *48*(2), 281-312.

Caris, A., & Janssens, G. K. (2009). A local search heuristic for the pre-and end-haulage of intermodal container terminals. *Computers & Operations Research*, *36*(10), 2763-2772.

Chen C, Demir E, Huang Y, Qiu R, 2021. The adoption of self-driving delivery robots in last mile logistics. *Transportation research part E: logistics and transportation review* 146:102214.

Choi, E., & Tcha, D. W. (2007). A column generation approach to the heterogeneous fleet vehicle routing problem. Computers & Operations Research, 34(7), 2080-2095.

Choudhury S, Knickerbocker JP, Kochenderfer MJ, 2019. Dynamic real-time multimodal routing with hierarchical hybrid planning. *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2397–2404 (IEEE).

Christiansen, M., & Nygreen, B. (1998). A method for solving ship routing problems with inventory constraints. *Annals of operations research*, *81*(0), 357-378.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. Operations Research, 8(1), 101-111.

De Maio, A., Ghiani, G., Laganà, D., & Manni, E. (2023). Sustainable last-mile distribution with ground drones and public transportation. Working Paper.

Desaulniers, G., Desrosiers, J., & Solomon, M. M. (Eds.). (2006). Column generation (Vol. 5). Springer Science & Business Media.

FedEx Roxo, 2022. *Meet roxo, the fedex sameday bot*. URL https://www.fedex.com/en-us/innovation/ roxo-delivery-robot.html, accessed October 26, 2022.

Fink, M., Desaulniers, G., Frey, M., Kiermaier, F., Kolisch, R., & Soumis, F. (2019). Column generation for vehicle routing problems with multiple synchronization constraints. European Journal of Operational Research, 272(2), 699-711.

Fortune business insights official website, 2023. *Delivery Robots Market, Share & Impact Analysis*. URL https://www.fortunebusinessinsights.com/delivery-robots-market-106955

Gendreau M, Nossack J, Pesch E, 2015. Mathematical formulations for a 1-full-truckload pickup-and-delivery problem. *European Journal of Operational Research* 242(3):1008–1016.

Ghiani, G., Guerriero, F., Laporte, G., & Musmanno, R. (2003). Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies. *European journal of operational research*, *151*(1), 1-11.

Gronalt, M., Hartl, R. F., & Reimann, M. (2003). New savings based algorithms for time constrained pickup and delivery of full truckloads. *European Journal of Operational Research*, *151*(3), 520-535.

Grippa, P., Behrens, D. A., Wall, F., & Bettstetter, C. (2019). Drone delivery systems: Job assignment and dimensioning. Autonomous Robots, 43, 261-274.

Ham, A. M. (2018). 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.

Huang, H., & Savkin, A. V. (2020). A method of optimized deployment of charging stations for drone delivery. *IEEE Transactions on Transportation Electrification*, *6*(2), 510-518.

Irnich, S., & Desaulniers, G. (2005). Shortest path problems with resource constraints. In Column generation (pp. 33-65). Boston, MA: Springer US.

Janssens, G. K., & Braekers, K. (2015). An exact algorithm for the full truckload pick–up and delivery problem with time windows: concept and implementation details. *International Journal of Computer Aided Engineering and Technology*, *7*(2), 260-272.

Jennings D, Figliozzi M, 2019. Study of sidewalk autonomous delivery robots and their potential impacts on freight efficiency and travel. *Transportation Research Record* 2673(6):317–326.

Jeon, A., Kang, J., Choi, B., Kim, N., Eun, J., & Cheong, T. (2021). Unmanned aerial vehicle last-mile delivery considering backhauls. *IEEE Access*, *9*, 85017-85033.

Liu, S., Hua, G., Cheng, T. C. E., & Dong, J. (2021). Unmanned vehicle distribution capacity sharing with demand surge under option contracts. Transportation Research Part E: Logistics and Transportation Review, 149, 102320.

Lübbecke, M. E., & Desrosiers, J. (2005). Selected topics in column generation. *Operations research*, *53*(6), 1007-1023.

Mitchell, J. E. (2002). Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization*, *1*(1), 65-77.

Mourad A, Puchinger J, Van Woensel T, 2021. Integrating autonomous delivery service into a passenger transportation system. *International Journal of Production Research* 59(7):2116–2139.

OpenRouteService of Heidelberg University's. URL github.com/GIScience/openrouteservice, accessed March 01, 2023.

Ostermeier, M., Heimfarth, A., & Hübner, A. (2022). Cost-optimal truck-and-robot routing for last-mile delivery. Networks, 79(3), 364-389.

Paradiso, R., Roberti, R., Laganá, D., & Dullaert, W. (2020). An exact solution framework for multitrip vehicle-routing problems with time windows. Operations Research, 68(1), 180-198.

Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2007). A survey on pickup and delivery problems. *Part II: Transportation between pickup and delivery locations, to appear: Journal für Betriebswirtschaft*.

Pugliese, L. D. P., & Guerriero, F. (2013). A survey of resource constrained shortest path problems: Exact solution approaches. Networks, 62(3), 183-200.

Rist, Y., & Forbes, M. (2022). A column generation and combinatorial benders decomposition algorithm for the selective dial-a-ride-problem. Computers & Operations Research, 140, 105649.

Savelsbergh, M. W., & Sol, M. (1995). The general pickup and delivery problem. *Transportation science*, *29*(1), 17-29.

Starship, 2022. *Starship technologies: Autonomous robot delivery*. URL https://www.starship.xyz/ business/, accessed October 26, 2022.

Tel Aviv GIS portal. URL https://gisn.tel-aviv.gov.il/iview2js4/index.aspx, accessed March 01, 2023.

Wang, Z., & Sheu, J. B. (2019). Vehicle routing problem with drones. Transportation research part B: methodological, 122, 350-364.

Xia, Y., Zeng, W., Zhang, C., & Yang, H. (2023). A branch-and-price-and-cut algorithm for the vehicle routing problem with load-dependent drones. Transportation Research Part B: Methodological, 171, 80-110.

Yu, M., Nagarajan, V., & Shen, S. (2022). Improving column generation for vehicle routing problems via random coloring and parallelization. *INFORMS Journal on Computing*, *34*(2), 953-973.

Zang, Y., Wang, M., & Qi, M. (2022). A column generation tailored to electric vehicle routing problem with nonlinear battery depreciation. Computers & Operations Research, 137, 105527.

Zhang J, Campbell JF, Sweeney II DC, Hupman AC, 2021. Energy consumption models for delivery drones: A comparison and assessment. *Transportation Research Part D: Transport and Environment* 90:102668.

תקציר

טכנולוגיית הנהיגה האוטונומית התפתחה באופן משמעותי בשנים האחרונות. יישום בולט של טכנולוגיה זאת מבוסס על רובוטים ניידים אוטונומיים. במערכות אלה, רובוטים קטנים הנעים על המדרכות במהירות הולכי רגל מספקים שירות משלוחים מנקודה לנקודה. הרובוטים מונעים באמצעות סוללה שקיבולתה מגבילה את טווח השירות של הרובוטים לרדיוס שירות של כשלושה קילומטר.

עבודה זו עוסקת במערכת בה צי של רובוטים מפוזרים במספר תחנות עגינה באזור השירות, רובוטים אלה מספקים שירותי משלוחים עבור בקשות כאשר כל בקשה מאופיינת בנקודת איסוף, בנקודת פריקה של המשלוח ובחלונות זמן לשירות בנקודות אלה. אנו בוחנים את הפוטנציאל של שיפור השירות באמצעות תחבורה ציבורית. כלומר, לאפשר לרובוטים לבצע חלקים מהנסיעה שלהם על גבי תחבורה ציבורית. בזמן נסיעה על גבי תחבורה ציבורית הרובוטים אינם צורכים אנרגיית הסוללה. הרחבה זו טומנת בחובה מספר הזדמנויות: ראשית, הרחבת טווח השירות לאזורים בהם הרובוט אינו יכול להגיע על ידי שימוש בסוללה בלבד, ובמקרים מסוימים, קיצור משך השירות. שנית, הפחתת צריכת האנרגיה הכוללת במערכת.

בעיית התכנון הנלמדת במחקר זה הינה מקרה מיוחד של בעיית האיסוף והמסירה הידועה (pickup and delivery problem) הכוללת הזמנות בגודל קיבולת הרכב (כתוצאה מכך שהרובוט יכול לשרת עד בקשה אחת בלבד באופק התכנון) וכן שילוב של מספר אמצעי תחבורה. פיתחנו שני ניסוחי מודל תכנות מעורב בשלמים, ניסוח מבוסס קשתות וניסוח מבוסס מסלולים. הראשון מייצג בצורה מפורשת כל תנועה אפשרית של רובוט בין שני קודקודים, השני מייצג מסלולים מלאים של רובוט במטרה לבחור את המסלול הטוב ביותר עבור כל בקשה (ורובוט). בעוד שלניסוח מבוסס מסלולים יש מבנה קומפקטי יותר, מספר המסלולים האפשריים גדל בצורה מעריכית יחד עם מאפייני המופע כגון מספר הבקשות ומספר הרובוטים.

על מנת להתגבר על בעיה זו פיתחנו שיטת חילול עמודות. אנו מגדירים קבוצה ראשונית של מסלולים אפשריים עבור כל זוג של רובוט ובקשת שירות ולאחר מכן אנו מנסחים את הבעיה המשנית כבעיית המסלול הקצר ביותר תחת אילוצי משאבים. פיתחנו אלגוריתם דינאמי בעל ארבעה שלבים כדי לפתור את בעיית המשנה. באמצעות ניצול סימטריה בבעיה הראשית אנו מפחיתים את מספר תתי הבעיות שיש לפתור בכל חזרה. לבסוף, אנו מיישמים מחשוב ממוקבל כדי לפתור מספר תתי בעיות בו זמנית. פעולות אלו מאפשרות לנו לצמצם את משמעותית את זמן החישוב הנדרש עבור כל חזרה של תהליך חילול העמודות.

תוצאות הניסוי מראות שבעוד שמודל מבוסס הקשתות יכול לפתור מופעים בגודל של עד 15 בקשות באופן מדויק, מודל מבוסס המסלולים בשיטת חילול עמודות יכול לפתור בעיות של עד 150 בקשות תוך שניות בודדות. יתר על כן, ערכנו מקרה בוחן תוך שימוש בנתונים אמיתיים מהעיר תל אביב, התוצאות של המחקר מוכיחות כיצד שימוש בתחבורה ציבורית מרחיב את טווח השירות של הרובוטים ומאפשר להם לטפל במספר גדול יותר של בקשות תוך חיסכון באנרגיה שלהם. מחקר זה מדגיש את היתרונות הפוטנציאליים של שילוב הרחבת שירותים משלוח מבוססי רובוטים אוטונומיים באמצעות תחבורה ציבורית ומספק גישה מעשית לפתרון בעיית התכנון התפעולי.

אוניברסיטת תל אביב

הפקולטה להנדסה ע״ש איבי ואלדר פליישמן בית הספר לתארים מתקדמים

ע״ש זנדמן-סליינר

שיטת חילול עמודות עבור שירות משלוחים רובוטי המורחב באמצעות תחבורה ציבורית

חיבור זה הוגש כעבודת גמר לקראת התואר ״מוסמך אוניברסיטה״ בהנדסת תעשייה

על-ידי

ישי שפירא

המחקר נערך במחלקה להנדסת תעשייה בהנחיית ד״ר מור כספי

כסלו תשפ״ד

אוניברסיטת תל אביב

הפקולטה להנדסה ע״ש איבי ואלדר פליישמן בית הספר לתארים מתקדמים

ע״ש זנדמן-סליינר

שיטת חילול עמודות עבור שירות משלוחים רובוטי המורחב באמצעות תחבורה ציבורית

חיבור זה הוגש כעבודת גמר לקראת התואר ״מוסמך אוניברסיטה״ בהנדסת תעשייה

על-ידי

ישי שפירא

כסלו תשפ״ד