Multi-agent planning and coordination for automated aircraft ground handling

Chen, Szu Tung; Ermiş, Gülçin; Sharpanskykh, Alexei

DOI
10.1016/j.robot.2023.104480

Publication date
2023

Document Version
Final published version

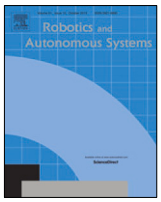
Published in
Robotics and Autonomous Systems

Robotics and Autonomous Systems 167 (2023) 104480

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot



Multi-agent planning and coordination for automated aircraft ground handling

Szu-Tung Chen, Gülçin Ermiş *, Alexei Sharpanskykh

Delft University of Technology, Faculty of Aerospace Engineering, Air Transport and Operations, Kluyverweg 1, Delft, 2629HS, The Netherlands

ARTICLE INFO

Article history:
Received 21 December 2022
Received in revised form 23 April 2023
Accepted 8 June 2023
Available online 16 June 2023

Keywords:
Automated aircraft ground handling
Task allocation
Multi-agent path planning
Vehicle routing with pick up and deliveries

ABSTRACT

Inspired by the vision of fully autonomous airside operations at Schiphol airport, this study aims to contribute to the short-term goal of automated aircraft ground handling. In this research, we design and evaluate a multi-agent system for planning of automated ground handling. There are two main components in the system: task allocation optimization and multi-agent path planning. To allocate tasks to ground support equipment (GSE) vehicles, an auction mechanism inspired by temporal sequential single item (TeSSI) auction is proposed. Ground handling tasks scheduling for GSE vehicles is modeled as several single-vehicle pickup and delivery optimization problems (SPDP), and the values of the objective functions are used to generate bids for GSE vehicle agents in the auction. Prioritized safe interval path planning for large agents (LA-SIPP) is used to plan collision-free paths for GSE vehicle agents in the model to execute tasks. The aim is to increase the success rates of allocating tasks and finding collision free paths without causing flight delays, given the limited resources such as a small number of available GSE vehicles, time windows constraints and conflicting interests of different agents. Due to the results, even for the instances with frequent flights and the most limited resources, the success rates of allocation and path planning were higher than 81% and 98%, respectively. Furthermore, periodic task allocation and path planning of the ground handling tasks for flights in three aircraft stands during a planning time window of the day, as well as replanning in case of disruptions were performed in a short CPU time. There is a lack of research dealing with the complete process of ground handling, since existing studies concerning the automation of ground handling operations involve fleet assignment or task scheduling models without an integration of detailed path planning. Our main contribution is to present a framework that combines task allocation and path planning for automation of ground handling operations and provides solutions using a multi-agent perspective.

1. Introduction

Similarly to many other airports, at Amsterdam Schiphol Airport, Royal Schiphol Group aims at achieving fully-autonomous airside operations by 2050 [1]. All vehicles will be replaced by interconnected, autonomous, and emission-free vehicles [2]. One of the anticipated implementations is the automation of ground handling processes. While the long-term goal is the fully-autonomous airside operations by 2050 at Schiphol, the short-term goal is automated docking of GSE vehicles. The automation will help using the resources more effectively, shortening the traveling distances and time, and reducing the emission produced by the aviation sector [3].

We deal with the problem where a limited number of automated ground handling vehicles are shared and reused to complete the common ground handling services at different aircraft stands and they travel between different aircraft stands. The vehicles travel on the service road connected to the terminal. The ground handling equipment should arrive at the ground handling service locations and complete the service within specific time windows matching the flight schedules. Each type of ground handling vehicle can serve only a specific type of ground handling task, so it is not possible to use or share any vehicle for all types of tasks. So far, only a few types of automated ground handling vehicles have been tested at some airports. An example is the baggage handling vehicle that is used to load and unload the baggage at terminal and aircraft stands. In our model, we consider the problem where the ground handling services, *catering, refueling, baggage handling, water* and *lavatory service* are automated, which is also one of the strategic goals for the ports. Ground handling service tasks are allocated to autonomous GSE vehicles which should travel on conflict-free paths to complete the assigned services over a period of time. We propose a new framework to solve the task allocation and path planning problem for the GSE vehicle agents with conflicting interests. The problem

* Corresponding author.
E-mail address: g.ermis@tudelft.nl (G. Ermiş).

https://doi.org/10.1016/j.robot.2023.104480

Acronyms

GPU	Ground Power Unit
GSE	Ground Support Equipment
LA − SIPP	Large Agents Safe Interval Path Planning
MAMP	Multi-Agent Motion Planning
MAPF	Multi Agent Path Finding
PCA	Preconditioned Air Unit
SIPP	Safe Interval Path Planning
SIPPwRT	Safe Interval Path Planning with Reservation Table
SPDP	Single Vehicle Pick-up and Delivery Problem
SSI	Sequential Single Item
TeSSI	Temporal Sequential Single Item
TSP	Traveling Salesperson Problem
VRP	Vehicle Routing Problem

of task allocation and path planning is NP-complete. The best allocation decision might not lead to the best collision-free paths or collision-free paths might increase the makespan of allocated tasks.

We define and solve a complex task allocation and path planning problem that is specific to airside operations. We focus on the ground handling operations at one pier at the airport. A pier includes several aircraft stands where ground handling tasks should take place. The GSE vehicles should move between these stands and terminal over a period of time without colliding. We define grid-based environments with obstacles, that are specific to a pier with aircraft stands, service road, and exits to terminal. The static obstacles for operation of ground handling in the proposed environments are the occupied areas by the aircraft or passenger. The proposed environment allows to simulate a realistic model for automated of ground handling.

To automate the ground handling process, we propose a framework that includes multi-agent path planning and multi-agent optimization based task allocation. For agent-based task allocation, we convert the single agent task scheduling problem of each GSE vehicle into single vehicle pick-up and delivery problem with time windows as a subproblem of the framework, after which we run iteratively the mathematical model of the vehicle routing model for the GSE vehicles within an auction process to improve decentralized decisions. The system allocates ground handling tasks to GSE vehicles, complying with temporal and operational constraints. After the tasks of GSE vehicles are allocated, trajectories of GSE vehicles are planned by obtaining collision-free paths considering time windows constraints. The task allocation and path planning for ground handling tasks can be performed in real-time within the predefined turnaround times of flights in offpeak, normal, peak time periods of the day.

An overview of the components of the framework is given in Fig. 1: The inputs in [1] are the ground handling tasks, depot and service locations, task processing times, time windows constraints derived from flight schedules, available GSE vehicle agents, grid-maps of the environment. At the first stage, we use an adapted version of TeSSI [4] auction model to allocate ground handling tasks among GSE vehicles, which is described in [1, 3, 4, 5] in Fig. 1. The bid of each agent for a candidate ground handling task in the auction is the optimized cost of performing the tasks by that agent. Alternatively, we apply a procedure called task bundling in [2] before allocating tasks, with the aim of accelerating task allocation. We repeat the experiments with bundling [1, 2, 3], and without bundling [1, 3]. We optimize the

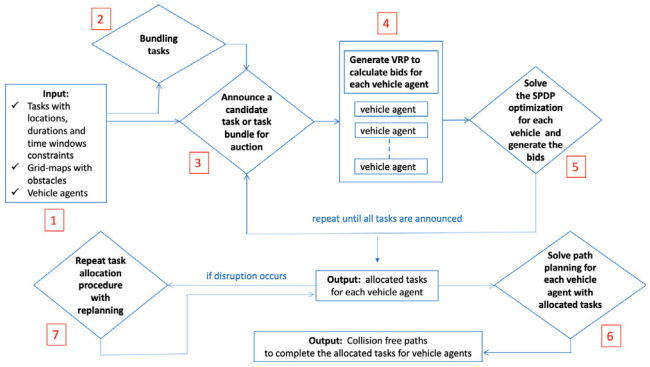


Fig. 1. An overview of the proposed task allocation and path planning approach.

schedule of each GSE vehicle by solving the single-vehicle pickup and delivery optimization problem to generate bids [4, 5]. We repeat the rescheduling iteratively and at each iteration one task is allocated to the winning agent (GSE vehicle) based on the bids of the agents. The procedure is repeated until each task is either allocated to a GSE vehicle or proven to be infeasible [3, 4, 5]. To find collision-free paths for the GSE vehicles to execute the allocated tasks, we use adapted versions of MAPF algorithms [6] and we use replanning in case of disruptions [7].

To solve the MAPF for GSE vehicles, we use priority based approaches [5–8], since precedence relations exist for different types of GSE vehicles or due to different time window restrictions for completing the tasks. We also deal with unit-size and square shaped large agents. We use *prioritized* SIPP to solve path planning problem for unit-size agents. SIPP [8] is a single agent path finding algorithm that considers the other agents as dynamic obstacles. For solving the problem with large square-shaped agents, we use an adapted version of LA-SIPP, SIPP for large agents, to consider the shapes and sizes of ground handling vehicles.

Experimental analyses were conducted to evaluate the performance of the model. We performed the tests using GUROBI and Python. We limited the solution time for the GUROBI optimization solver by 5 s for each agent, thus finding a feasible solution, might not always be possible. If none of the GSE vehicles finds a feasible position for a task in their schedules, the task remains unallocated. Thus, one of the performance indicators is the allocation rate. We analyzed the *makespan*, *task allocation rate*, and *CPU time* for task allocation in different scenarios, and *path length*, *path duration*, *CPU time*, *success rate*, *average delays per task and per agent* and *ontime rates per task and per agent* for path planning algorithm. Furthermore, sensitivity analyses were performed by changing the number of ground handling vehicles. Task allocation experiments were also repeated with bundling and replanning. Path planning solutions were compared to shortest paths with collisions.

Our main contributions are:

- We developed an agent-based combined task allocation and path planning approach specifically designed to solve the automated ground handling problem. Due to the additional constraints required for ground handling, our problem is more complex than the classical task allocation and path planning problem which is already known to be NP-complete.
- Our framework was able to generate solutions with high success rate in reasonable time which allows dynamic planning of operations in real time.
- We developed a realistic model taking into account flight schedules at peak hours at the hub airport and airport environment with static and dynamic obstacles.

- The collision free paths that we generate for GSE vehicles on aircraft stands and service road also contribute to the safety of autonomous GSE vehicles while traveling on and between the aircraft stands and service road, which has not been explored before.
- Although the original problem is not a vehicle routing problem with pick-up and deliveries, we converted the original problem into several single machine pick-up and delivery problems with time windows constraints by splitting the main ground handling tasks into pick-up and delivery subtasks. This allows us to solve the problem to optimality using the classical integer programming models of single vehicle pick-up and delivery problem and exploit the advantage of using time windows constraints in which case sub-tour elimination constraints can be eliminated and the vehicle routing problem can be solved more efficiently.

The paper is structured as follows. The literature is given in Section 2. The assumptions regarding ground handling operations and model specifications are introduced in Section 3 and Section 4. The task allocation mechanism using auctions and single vehicle pick-up and delivery optimization model is discussed in Section 5. In Section 6, the path planning algorithm is discussed. Experimental results are presented in Section 7. Statistical analysis, discussions and conclusion are given in Section 8, Section 9 and Section 10, respectively. Appendices are presented in Appendix A, Appendix B, Appendix C, Appendix D.

2. Related studies

The automation of aircraft ground handling operations is a novel research direction. In terms of hardware, robotics and autonomous vehicles technologies have been developed. However, the required software system supporting the automated operations and the transformation from current to automated systems are still under development [9].

The advancements in technology and standardization of the procedures create opportunities for automated aircraft ground handling. The ground handling procedures on aircraft stands are similar among commercial aircraft. The configuration of most modern civil aircraft is similar [10]. The aircraft service interfaces follow standards, and aircraft service points are usually placed at common areas. The connectors of electricity are usually at the front and the refueling interfaces are under the wings. [11] analyzed the service interfaces used in ground handling of different types of aircraft and cluster the interfaces based on normalized aircraft length and half-wingspan. The interfaces include passenger and cargo door position, service panels of portable and waste water, preconditioned air unit (PCA) interfaces, ground power unit (GPU) interfaces, and fuel connectors. The cluster analysis reveals a trend of shifting of the interfaces towards common locations. According to [10], the aircraft doors and their usages for passenger boarding, catering, cleaning, cargo are standardized. Considering the similarities among the procedures, aircraft configurations, and clustered interfaces, automation of ground handling operations is an achievable goal and a promising research field. The ramp layout for Boeing 737-800 and standard locations for ground handling operations are given in Fig. 2.

Agent-based modeling uses an approach that defines the system from the perspective of interacting heterogeneous autonomous entities. It can capture various goals and interests of the involved agents, as well as the interaction and the interdependence between them. In the operation of aircraft ground handling, multiple stakeholders are involved, including the airline, airport operator, ground handling service companies, and air traffic control. The stakeholders have different interests and goals and their behaviors are dependent on other agents. Thus, intensive communication and cooperation are required. Some aviation-related applications include agent-based delay management in autonomous taxiing [13] and resilient airport surface movement [14]. However, in the field of aircraft ground handling, little research on agent-based modeling approaches exists. Multiple stakeholders are involved in the automation of aircraft ground handling. To assign the ground handling tasks to GSE vehicles such that a global objective function is optimized, centralized approaches are usually inefficient. On the other hand, auction mechanisms [15–19] are considered suitable frameworks for task allocation of automated aircraft ground handling. Throughout the auction processes, bids of agents can capture the individual goals and utilities of agents under various circumstances. Auction-based multi-agent task allocation solvers can also efficiently perform with different team objectives. Dynamic environments of ground handling operations create uncertainties. To quickly adapt to operational changes and deviations, the real time replanning and computational efficiency are important. To improve the computational efficiency, task assignment and path planning are conducted separately in most works [20]. Using these approaches, although the solution quality is compromised, solution time becomes reasonable for real time applications. Solving the task allocation and path planning simultaneously is more complex due to the temporal and operational constraints in automated ground handling. We present the literature related to task allocation, path planning and ground handling in Section 2.1, Section 2.2, and Section 2.3, respectively.

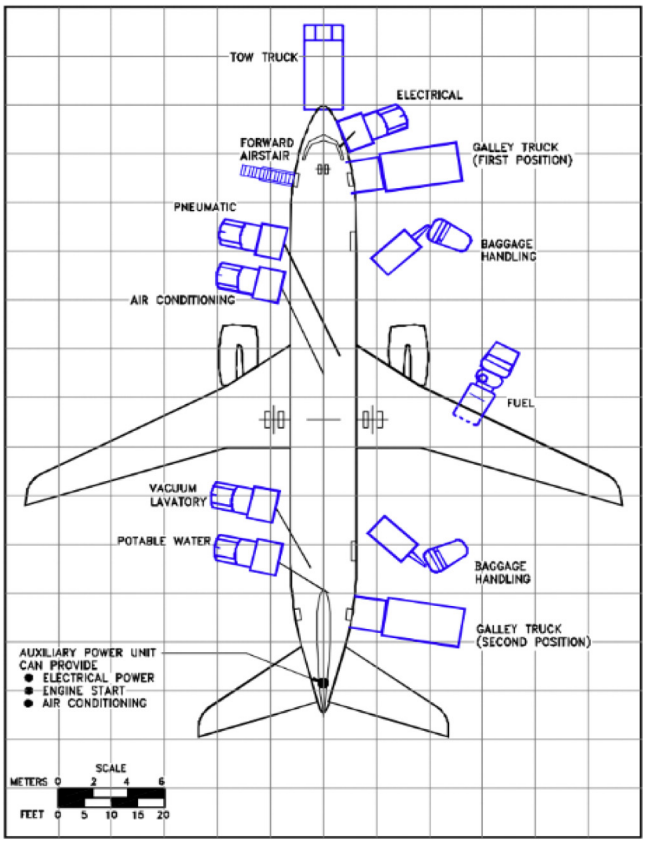


Fig. 2. Ramp layout for Boeing 737-800 [12].

2.1. Related work on task allocation

Allocating ground handling tasks to ground service equipment is one of main steps in automation of ground handling

operations. From centralized solvers to decentralized approaches, some common classes of task allocation solvers are operation research algorithms, evolutionary and swarm intelligence algorithms, auction-based multi-agent systems, and game-theoretic approaches. Task allocation of aircraft ground handling operations can be modeled as the assignment problem, which is one of the fundamental problems in operation research. Kuhn–Munkres Hungarian algorithm was developed by [21] to solve the assignment problems in $O(rt^2)$ time, in which r is the number of agents and t is the number of tasks. Other solution approaches are vehicle routing solvers, dynamic programming, branch and bound methods. Evolutionary and swarm algorithms include genetic [22,23], ant colony [24,25], and bee colony [26] algorithms. Genetic algorithm has a wide range of applications in airport operations, including optimization of airport ground operations [27], scheduling for baggage transport vehicles [28], ground traffic optimization [29] and finding optimal locations of airport fire stations [30]. Ant colony and bee colony algorithms solve difficult assignment problems with fast convergence to high-quality solutions. [31] use ant colony algorithm to solve the gate assignment problem. [32] propose an improved ant colony optimization algorithm, and verify its performance on the traveling salesperson and gate assignment problems.

Centralized approaches are often inefficient for distributed systems due to the computational time and communication limits. Therefore, auctions were studied by researchers to solve the task allocation problems [16,33]. Auctions have been studied extensively in the field of artificial intelligence and multi-agent task allocation. An early work applying auctions into distributed problems is the contract net protocol [19]. Other examples of using auction in task allocation include allocating roles to robots in RoboSoccer [15], tasks to ambulance teams, fire bridges and police force in RoboCup Rescue [17], and task allocation for online pickup and delivery problems [18]. Multi-agent task allocation problems can be modeled as auction-based agent coordination systems. Auction-based multi-agent task allocation solvers include sequential single item auctions [16] where tasks are allocated in multiple auction rounds, combinatorial auctions where tasks are allocated in a single round auction, sequential single cluster auction [16,34] where agents bid on some subsets of tasks, consensus-based auction algorithm and consensus-based bundle algorithm [35] which solve static allocation problems using a consensus routine based on local communication. Another approach is the game-theoretic approach where agents are modeled as independent decision-makers that attempt to maximize their own utility. In game theory, agents are often competitive and unwilling to share their private information. Thus, the agents decide based on local information and expectations.

2.2. Related work on path planning

Multi-agent path finding (MAPF) is an important field of multi-agent planning. Paths are planned for multiple agents, so that agents can travel from their origins to destinations simultaneously without collision with other agents or obstacles in the environment. MAPF has a wide range of real-world applications, including automated warehouse [36], traffic control [37], robotics [38], and airport surface operations. [39,40] provide a survey of MAPF algorithms and their applications. Existing algorithms that solve classical MAPF problems include A^* approaches such as operation decomposition [41], independence detection [41], enhanced partial expansion $A*$ [42], increasing cost tree search [43], conflict based search [44], meta-agent conflict based search [44], cooperative $A*$ [5], hierarchical cooperative $A*$ [5], windowed hierarchical cooperative $A*$ [5], conflict oriented windowed hierarchical cooperative $A*$ and conflict oriented windowed hierarchical cooperative $A*$ with prioritization [6], conflict-based search with priorities [7], priority based search [7], boolean satisfiability [45], constraint satisfaction problem [46,47], answer set programming [48], integer linear programming [38]. Most of these algorithms are based on the assumptions that time steps are discrete, the shapes and sizes of the agents are uniform, and movements are in orthogonal directions. These methods are improved by considering continuous time steps or non-uniform shapes and sizes. The MAPF solvers using continuous time steps are SIPP [8] and continuous-time conflict-based search [49]. A MAPF solver that considers the shapes of agents is LA-MAPF [50], multi-agent path finding for large agents. Multi-agent motion planning, MAMP, is the generalization of the MAPF. [51] use the term MAMP to refer to the task of finding kinodynamically feasible plans for agents. MAMPs are used to model the movement of ground handling vehicles. SIPP [8] considers the paths reserved by the other agents as obstacles and forms the basis for MAMP approaches. Safe interval path planning with reservation table [52] handles agents with volumes. Soft conflict interval path planning [53] is an extension of SIPP that adapts (enhanced) conflict-based search to continuous time.

The main difference between SIPP and other priority based approaches [5–7] is that SIPP searches in a state space consisting of pairs of agent configurations and intervals rather than configurations and time steps. A configuration is a set of variables, describing an agent's physical position, heading, velocity and/or other related information. Safe intervals and conflict intervals are used to limit the size of the search space. One timestep before and after a safe interval, the configuration is in collision with dynamic obstacles, and the opposite is true for collision intervals.

2.3. Related work on ground handling

A recent work focusing on cooperative scheduling models for ground handling operations under flight uncertainty is the study of [54]. The focus of this study was on refuellers and coordination with bus shuttles. They developed mixed integer programming models with chance constraints to reflect the uncertainty of flights and provide a coordinated optimization schedule for ground-handling vehicles' dispatching, to reduce the waiting times and delays. [55] focuses on scheduling airport baggage transport vehicles using genetic algorithm. [56] presents an agent-based distributed control system for resource scheduling and provide an application on managing ground handling operations based on radio frequency identification feedback. The work of [57] is one of the few studies addressing the whole ground handling operations at the airport ramp. They developed a bi-objective optimization model for scheduling operations of ground handling vehicles including catering, baggage handling, deboarding, cleaning, fueling, water and lavatory services, to minimize waiting times of vehicles and turnaround time. Another study that consider all ground handling operations as part of a mathematical optimization model for schedule recovery optimization is the work of [58]. The mathematical model is applied on peak hours at an airport with 20 turnarounds.[59] model and simulate the turnaround process using Petri Nets. A recent work towards automation of ground handling operations is provided by [60] where they propose an on-line fleet assignment problem for assigning GSE vehicles to ground handling tasks. A stochastic approach that combines simulation based on actual data with optimization, to find robust scheduling solutions for GSE equipment is proposed by [61], and the results show that number of aircraft handled within the planned time windows and robustness of the schedules increased. The limitation of these studies is that they do not address the collision-free movements of GSE equipment on the airside in an automated system, thus,

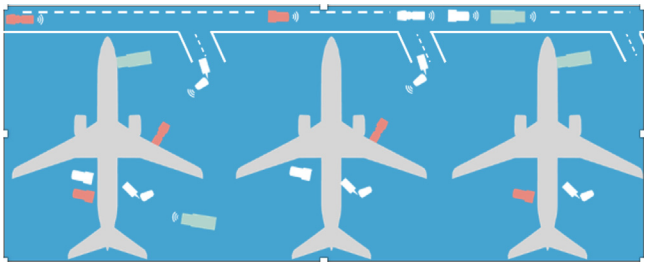


Fig. 3. Automated ground handling at aircraft stands of a pier.

the path planning and safety of automated GSE vehicles, which are in the scope of Robotics and Automation, are not included in research. The solutions are usually provided for a single aircraft stand and traveling between different stands are not considered. Also, many of the existing studies suggest centralized solutions for improving the ground handling process, rather than modeling a distributed agent-based approach.

3. Problem definition

We present a brief description of the problem in Sections 3.1, 3.2, 3.3, 3.4, 3.5, 3.6. Further details related to problem and model description are presented in Section 4.

3.1. Scope and layout

Amsterdam Schiphol Airport has 91 connected gates, located across 7 piers. We focus on ground handling operations of a single pier. We take the pier B at Amsterdam Schiphol Airport as reference, to set the configuration, the number of aircraft stands, infrastructure and operating flights. We consider the ground handling tasks, refueling, catering, baggage handling, water service and lavatory service. We present the general overview of the problem in Fig. 3.

The aircraft stands are located next to each other in the pier. Each stand is occupied by an aircraft. GSE vehicles serve to the aircraft at several stands. At the top in Fig. 3, GSE vehicles are traveling on the service road. Within the stands, they are heading to the service points around the aircraft or to the exits of the stands. Some of them are parked at the service locations to complete the tasks. At each stand, white vehicles serving on the right are the baggage handling trucks, while the green vehicles are the galley trucks. Each type of GSE vehicle must serve at specific positions around the aircraft.

3.2. Aim and constraints

The aim is to find multi-agent task allocation and path planning solution which minimizes delays, makespans, collisions for all agents that are serving the aircraft located at several stands between flights during a time period of the day. GSE vehicle fleet size is limited, and they are shared and reused at different times and platforms. Each GSE vehicle agent tries to minimize the makespan of its own schedule while a short turnaround time for each flight is also aimed to be maintained. The tasks have to be completed within specific time windows. GSE vehicles have capacity and precedence constraints, and mobility restrictions. There are traveling direction constraints on the service road, and blocked parts at the layout which are occupied by static obstacles such as the aircraft or passenger boarding bridges.

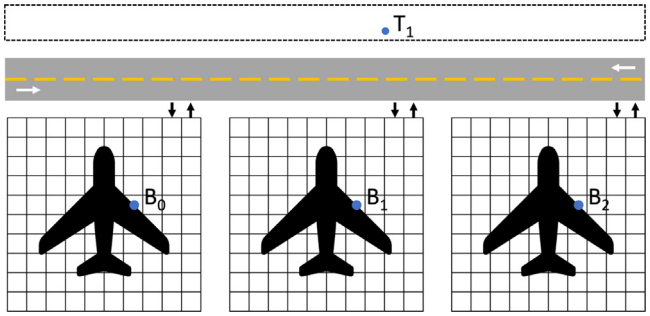


Fig. 4. The depot (terminal), service road, and aircraft stands, T_1: depot, B_1, B_2, B_3: aircraft stands.

3.3. GSE vehicles

Five types of GSE vehicles, refueling truck (RE), catering truck (CA), baggage handling truck (BA), water service truck (WA) and lavatory service truck (WC) are defined to execute refueling, catering, baggage handling, water service and lavatory service tasks at the bays. The vehicles are positioned at the terminal where their depots are located before the service starts.

3.4. Flight schedules

We consider the flight schedules within the four hours planning windows. The time intervals between flights define the time windows constraints to complete ground handling tasks. Let F_1, F_2, F_3, F_4 be the time intervals during which an aircraft stands at BAY_1 within the planning time window of the day. The ground handling tasks have to be completed for a different aircraft during each one of the time intervals F_1, F_2, F_3, F_4 at BAY_1. Similarly, if F_5, F_6, F_7, F_8 and F_9, F_{10}, F_{11}, F_{12} are respectively the time intervals the aircraft stand at BAY_2 and BAY_3, ground handling services have to be completed during each of these intervals at BAY_2 and BAY_3 by the available GSE fleet.

3.5. Bays

Available GSE vehicles serve three bays, BAY_1, BAY_2, BAY_3, with specific entrance and exit points for GSE vehicles. A refueling truck follows a path that passes through the service road and entrance point of a bay, reaches to the refueling service point at that bay for service, and leaves the bay using the exit point to go back to terminal or to another bay. The same is valid for other vehicles. Fig. 4 illustrates the depot, service road and the bays.

3.6. Grid-maps, distances, paths

We define a grid based environment and find the paths on grid-maps using Manhattan distance. The terminal, service road, and the bays are included as grid-maps. Although there are shortest paths to reach destinations in Manhattan form, these paths can change to avoid obstacles or collisions with other vehicles. We define two types of grid-based environment: the basic and the extended model. The basic model is defined by 16×16 grid-map with 4 m cell size, and unit-sized agents, while the extended model is defined by 64×64 grid-map with 1 m cell size and agents have non-unit sizes occupying several cells. Fig. 5 presents the basic (a) and extended (b) grid-based environments for aircraft stands with the service locations for refueling (RE), catering (CA), baggage handling (BA), water service (WA), and lavatory service (WC), and the complete modeling environment (c) including the service road, terminal, aircraft stands, and pick-up and delivery locations on this environment.

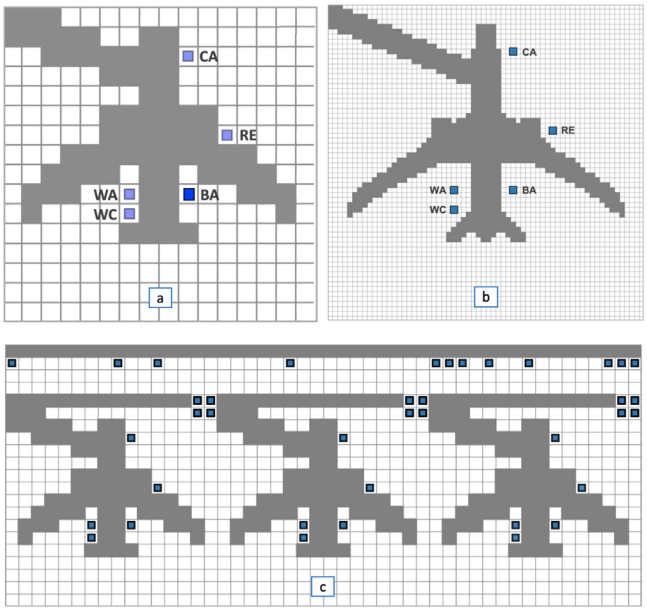


Fig. 5. Modeling environment: (a) 16 × 16 grid-map environment of aircraft stands including the locations of service points. (b) 64 × 64 grid-map environment for aircraft stands including the locations of service points. (c) The complete environment including the service road, terminal, bays, bay entrance and exists, and pick-up and delivery locations of GSE vehicles on 16 × 16 grid-maps.

4. Design considerations

We design a multi-agent system model for automation of aircraft ground handling operations. We specify the features and interactions of ground handling equipment, operations, infrastructure, environment as grid maps, and agents in Sections 4.1–4.3.

We focus on the ground handling activities at the aircraft stands. Aircraft stands are areas of an airport where aircraft park between flights for passenger boarding and deplaning, cargo loading and unloading, refueling, and other cabin services. Typical ground handling activities carried inside or outside the cabin are, passenger deplaning and boarding, cabin cleaning and preparation, catering preparations, safety and security checks, cargo unloading and loading, catering galleys unloading and loading, connecting and disconnecting Ground Power Unit (GPU), connecting and disconnecting pre-conditioned air unit (PCA), refueling, water and lavatory service, connecting and disconnecting passengers boarding stairs or Passenger Boarding Bridge (PBB). In this study we consider the activities outside the aircraft cabin. The activities we consider are, refueling, catering, baggage handling, water service and lavatory service. The ramp layout and ground service equipment handling the activities outside the cabin for Boeing 737-800 are shown in Fig. 2. We refer to this layout to set-up the environment for the model. In Fig. 2, the baggage handling, refueling, and catering (galley truck) vehicles are located on the right side of the Boeing 737-800 aircraft, and water and lavatory service trucks are located on the left side of the Boeing 737-800 aircraft. Thus, we set the service locations of catering, refueling, baggage handling, water service and lavatory service trucks in our environments based on the layout of Boeing 737-800. We also model the area covered by the passenger boarding bridge as an obstacle that the GSE vehicles cannot move on.

4.1. Assumptions

4.1.1. Ground handling activities

We focus on the ground handling operations of the aircraft used for short-haul flights. We consider the ground handling activities of a single type of aircraft, Boeing 737-800, outside the aircraft cabin. These activities include refueling the aircraft, unloading and loading the catering galleys, unloading and loading the baggage, water and lavatory service. Refueling and charging of the ground handling vehicles, seasonal operations such as de-icing are not included.

4.1.2. Infrastructure

Aircraft stands (also referred to as aircraft bays) are the parking spots for aircraft to park between flights. Regarding the infrastructure of the aircraft stands, it is assumed that all aircraft stands are equipped with Passenger Boarding Bridges (PBB), and all passengers are able to board the aircraft via PBBs. Also, all aircraft stands have an underground fueling system so that only small fuel filling vehicles are needed.

4.1.3. Operation

We consider the case where Ground Service Equipment are shared by the aircraft stands in the same pier and the same type of ground handling vehicles are homogeneous. GSE vehicles required to serve one short haul flight are listed and sorted as Fuel filling truck (RE), Catering truck (CA), Baggage vehicle (BA), Water service truck (WA), Lavatory service truck (WC) from highest to lowest priority (adapted from [10] and internal documents). In case of conflicts between movements of GSE vehicles, the listed order is used to set the priority. There is no bound on the number of personnel available to support the operation of GSE. Movements of aircraft are not considered in the path planning stage of the model. Thus, we perform path planning only for the GSE vehicles. Taxiing and towing of aircraft are not included in the scope of this research. The path planning of PBB is also not considered. For safety reasons, ground handling vehicles have to enter or exit the ramp through specified traffic lanes.

4.1.4. Task duration and time windows

We define the duration and time windows of different types of ground handling tasks referring to the turnaround time chart of aircraft B737-800 provided in technical documentation of Boeing [12]. The time chart is only a guideline for various turnaround operations. The real execution time of the ground handling tasks is affected by different factors. The execution duration of ground handling tasks should lie in the specified time windows.

4.2. Environment specification

An airport stand can be modeled as an environment with grids, as shown in Fig. 2. In the multi-agent system, we model an airside environment, including vehicle depots, a service road, and several aircraft stands. All modeled elements are represented by grid maps with obstacles. Consulting the typical GSE vehicle layouts on an aircraft stand, we modeled grid-map environments with 15%–20% obstacles. The aircraft, the passenger boarding bridge, and the towing vehicle are fixed objects and they are modeled as obstacles. An instance of modeling environment is shown in Fig. 6.

To model the aircraft stands, we use 16 × 16-grid maps where each GSE vehicle agent occupies one cell of the map, *the basic model*, and 64 × 64-grid maps where GSE vehicle agents occupy several cells around reference points, *the extended model*. The environment of aircraft stand modeled by 64 × 64-grid

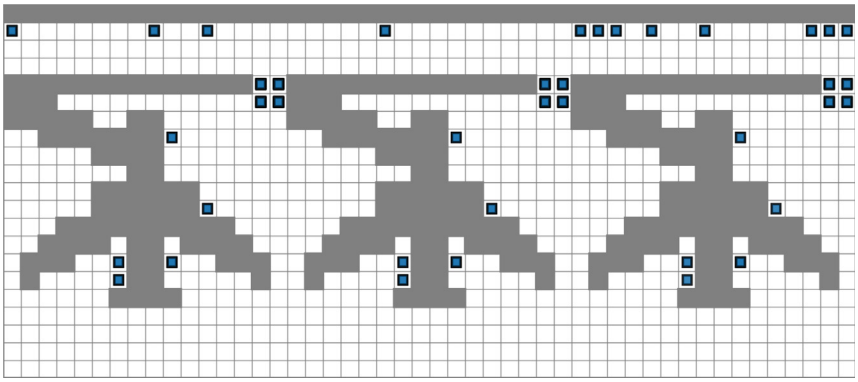


Fig. 6. An instance of modeling environment: Service road, aircraft stands, entrance, exit, pickup and delivery locations of ground handling vehicles.

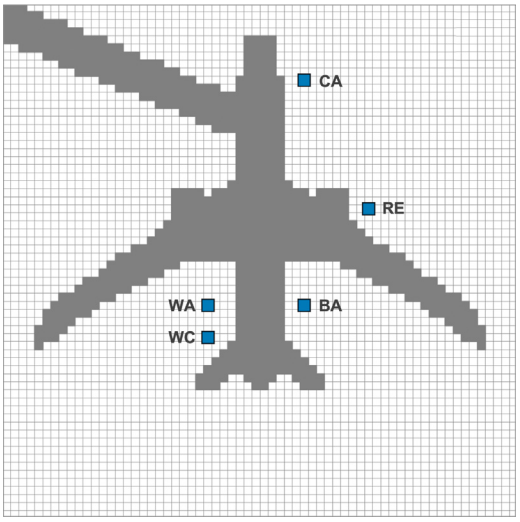


Fig. 7. Environment of an aircraft stand modeled by 64 × 64-grid map and the positions of non-unit size GSE vehicles for refueling (RE), baggage handling (BA), catering (CA), water service (WA), and lavatory service (WC).

map, together with the locations for performing different ground handling tasks, is illustrated in Fig. 7.

Depending on the grid size used for the aircraft stands, we modeled the service road environment as open grids with the same grid size as the aircraft stands. We applied directional constraints on the grids of the service road. The constraints make sure that the GSE vehicles always drive on the right side of the road. The vehicle depots are represented by the cells connected to the service road. GSE vehicles have to enter or exit the aircraft stands via corresponding bay entrances and exits.

4.3. Agent specifications

The multi-agent system consists of the following types of agents: task allocation agents, GSE vehicle agents, path planning coordination agents, and environment agents.

4.3.1. Task allocation agents

The task allocation agents are modeled as the auctioneers of the auctions. Every type of ground handling task is allocated separately, so there is one task allocation agent for each type of ground handling task. The agents have the following properties:

- *Task announcement property*: Given the flight schedule within the model planning window, agents first formulate different types of ground handling tasks that have to be performed for the flights. Then, as auctioneers, the task allocation agents announce the tasks to the bidders, the GSE vehicle agents. The auction has multiple rounds and one task is announced in each round. Information about tasks including their locations, duration, and time windows is declared at this stage.
- *Winner determination property*: After announcing the task in every round of the auction, GSE vehicle agents bid on the task and send their bids to the task allocation agents (the auctioneer). The task allocation agent determines the winner of the task depending on the agents' bids.
- *Task allocation property*: After determining the winning agent of a task, the task allocation agent announces the winner and allocates the task to the GSE vehicle agent.

4.3.2. GSE vehicle agents

Each GSE vehicle is modeled as an agent. The GSE vehicle agents are involved in both task allocation and path planning mechanisms of the model. The agents are classified by their type of ground handling tasks. There are two available GSE vehicle agents for each type of ground handling task except for baggage handling, for which four available GSE vehicle agents exist. The GSE vehicle agents have the following properties:

- *Bidding property*: The GSE vehicle agents are bidders in the task allocation auctions of their type of ground handling task. After the auctioneer announces the task to bid in each round of the auction, every bidder determines its bid for the task. The bidder then sends its bid to the auctioneer. Task allocation is discussed in more detail in Section 5.
- *Scheduling property*: If the GSE vehicle agent is chosen as the winner of a ground handling task, the agent adds the task into its schedule. Based on its schedule, a GSE vehicle agent can plan trip itineraries including the locations it needs to visit.
- *Path planning property*: Given the trip itineraries of all agents, map of the environment, and continuous periods without collisions for every grid on the map (safe intervals), the GSE vehicle agents are able to plan their paths without colliding with other agents or obstacles in the environment. Paths of individual trips in the agents' itineraries are planned separately. Path planning is discussed in more detail in Section 6.

4.3.3. Path planning coordination agent

The path planning agent manages the priority order of the agents. Agents are first sorted based on their types of ground handling tasks, then they are sorted by their departure time from the depot. The priorities of agents from highest to lowest are as follows: Fuel filling truck (RE), Catering truck (CA), Baggage vehicle (BA), Water service truck (WA), Lavatory service truck (WC).

4.3.4. Environment agents

The environment contains two kinds of maps: service road and bay maps. There are several bay maps and one service road map in the model. For each map the corresponding map agent is defined.

- *Safe interval management property*: The map agents keep track of the safe intervals on their map. Every time a path is planned on a map by a GSE vehicle agent, the path is sent to the corresponding map agent. The safe intervals of the cells on the map are then updated.

4.4. Solution approach

First, task allocation problem is solved to assign tasks to GSE vehicle agents. The shortest paths to complete the assigned tasks are known after the task allocation is finished, since the time to traverse the paths between various cells of grid-maps are also included the schedules of GSE vehicle agents. However, these paths are not collision-free. At this stage, path planning is used to find the collision-free paths for GSE vehicles. The vehicles either stop and wait or prefer to use alternative paths in case of conflicts. Task allocation is solved using an optimization based auction procedure. GSE vehicles optimize their schedules at each round of the auction and bid accordingly. We convert the single agent task scheduling problem into single vehicle pick-up and delivery problem with time windows. We solve the single vehicle pick-up and delivery optimization model using GUROBI solver with a solution time limit of 5 s, for each GSE vehicle agent, to generate the bids of GSE vehicles at each round of the auction. Collision-free paths are found using priority based path planning algorithms.

5. Task allocation

Task allocation is one of the main components of automated aircraft ground handling. To complete various kinds of ground handling tasks of all flights on time, ground handling tasks must be allocated to GSE vehicles using a effective and efficient procedure that minimizes the completion times, while respecting the capacity, connectivity, and temporal constraints. In this research, an auction mechanism inspired by temporal sequential single-item (TeSSI) auction [4] is proposed to allocate tasks to GSE vehicles. The distinctive part of the proposed procedure from TeSSI is that the executions of ground handling tasks in schedules of GSE vehicles are modeled as single vehicle pickup and delivery optimization problems (SPDP) [62] to generate the bids for the auction. Bundling of tasks is also included in the model to speed up the allocation process. As the required GSE vehicles for different types of ground handling tasks are different, each type of ground handling task is allocated separately.

5.1. Adapted TeSSI auction for ground handling task allocation

Within the planning window and planning scope of the model, different types of ground handling tasks for flights are allocated separately in several adapted temporal sequential single item (TeSSI) auctions. TeSSI auctions are extensions of single sequential item auctions introduced by [4]. The strength of TeSSI is its ability to handle temporal constraints. Time windows are used to indicate the time intervals within which tasks have to be processed.

Consider a set of agents $A = \{a_1, a_2, \ldots, a_n\}$, representing the available ground handling vehicles, and a set of ground handling tasks $T = \{t_1, t_2, \ldots, t_m\}$. Each task t_i has its earliest possible start time Est_i and its latest start time Lst_i, where $Est_i \leq Lst_i$. The duration of the task is denoted as Dur_i and the latest finish time is Lft_i, with $Lst_i + Dur_i = Lft_i$. The interval $[Est_i, Lft_i]$ defines the time interval in which the task has to be performed.

In the auction, tasks are allocated independently in multiple rounds. In each round of the auction, all participating agents bid on an unallocated task. The agents evaluate the cost of committing to the task-to-bid, taking into account their current schedule. After generating bids on tasks, the agents send their bids to the auctioneer. Once the auctioneer receives all the bids, the auctioneer performs winner determination and allocates the task to the agent who bids the best price. If no agent bids on a task, resulting in the task cannot being assigned, the task is removed from the set T. The auction continues until all the tasks in T have been allocated or discarded. If a task is discarded because there is no feasible solution and infeasibility is proven before the time limit of the solver is reached, the number of GSE vehicles could be increased to avoid discarded tasks in real world. In the event that neither the infeasibility is proven nor a feasible solution is found within the time limit, repair heuristics could be used to find feasible allocations for discarded tasks if such solutions exist.

In the original TeSSI auction, agents maintain the temporal consistency of their allocated tasks using simple temporal networks (STN). In this research, we model the execution of ground handling tasks as pickup and delivery operations. Therefore, STN for agents is substituted with single agent pickup and delivery optimization models, which is discussed in Section 5.2. We refer to the new optimization based TeSSI auction approach as Adapted TeSSI auction.

5.1.1. Bidding rules

Unlike common auctions, auction mechanisms used for task allocations are usually cooperative auctions. It is crucial in cooperative auctions to establish bidding rules that optimize the problem objectives. In the model, we consider two different team objectives. The first objective is to minimize the maximum completion time over all tasks allocated to a team of agents (also known as *makespan*). Minimizing *makespan* is often referred to as MINIMAX and it is commonly used in auction-based routing literature [63]. Another objective we aim to minimize is the total traveling time of all GSE vehicles to complete all tasks in the task set. It is a variant of the objective MINISUM [63]. We use the term *sum-T* to refer to the objective in this paper. Let $F = \{F_1, F_2, \ldots, F_n\}$ be the final allocation of tasks, where F_i contains all allocated tasks of agent a_i. $C_{max}(F_i)$ denotes the *makespan* for agent a_i to complete all allocated tasks in F_i. Also, $TT(F_i)$ represents the minimum traveling time that agent a_i must spend to perform its allocated tasks in F_i. The objective functions that are to be minimized are the maximum *makespan* over all agents, $z_1 = max_i[C_{max}(F_i)]$, and the sum of travel times of all agents, (*sum-T*), $z_2 = \sum_i TT(F_i)$.

Let t be the task-to-bid in the current round of auction, and $S = \{S_1, \ldots, S_n\}$ be the current partial allocation of the tasks to agents a_i, $i = 1, \ldots, n$. Depending on the objective, the following bidding rules are applied:

- bid for objective *makespan – makespan of agent* a_i: Agent a_i bids the new minimum makespan of its tasks after including task t in its plan, $min[C_{max}(S_i \cup t)]$.
- bid for objective *sum-T – marginal-sum-T*: Agent a_i bids the marginal increase of traveling time incurring in adding t to its plan, $min[TT(S_i \cup t)] - min[TT(S_i)]$

5.1.2. Pseudo-code

The procedure of TeSSI auction for the auctioneer is shown in Algorithm 1. The inputs of the algorithm contain a set of agents, A, and a set of tasks, T, that are to be allocated to agents. Task set, $T_{unallocated}$, contains the tasks that cannot be allocated. In the

beginning, the task set $T_{unallocated}$ is an empty set. For each task t in T, every agent $a \in A$ evaluates the task using the function *evaluateTask*. According to the applied bidding rule, each agent a generates a bid on the task and a corresponding schedule if the task can be included in its schedule without violating any constraint. The generated bids for task t from agents are compared to the current lowest bid *lowestBid*. If the minimum bid is lower than the existing lowest bid, the current lowest bid replaces the existing lowest bid. The bidding agent a and its generated schedule *tempSchedule* are saved. Also, the task t is marked as *assigned*. After all the agents have tried to bid on the task t, it is allocated to the winning agent *winner*. The schedule of the agent *winner* is updated. On the other hand, if task t is not assigned to any of the agents, it is moved into the unallocated task set $T_{unallocated}$. The auction procedure continues until all the tasks in T are allocated or moved to $T_{unallocated}$. The outputs of the algorithm are sets of assigned tasks for all agents $a \in A$, and the final schedules $Schedule_a$, $a \in A$.

Input:
  A: Set of agents
  T: Set of tasks to be allocated
Output:
  S_a: Sets of assigned tasks for agents a, $a \in A$
  $Schedule_a$: Final schedules $Schedule_a$ for agents, $a \in A$
$T_{unallocated} = \emptyset$;
for $t \in T$ **do**
  $lowestBid = \infty$
  $assigned = False$
  for $a \in A$ **do**
    $Bid, tempSchedule = a.evaluateTask(t)$
    if $Bid < lowestBid$ **then**
      $winner = a$
      $lowestBid = Bid$
      $winnerSchedule = tempSchedule$
      $assigned = True$
    end if
  end for
  if $assigned = True$ **then**
    $S_{winner} = S_{winner} \cup \{t\}$ {update the set of assigned tasks for the winning agent}
    $Schedule_{winner} = winnerSchedule$ {update the schedule of the winning agent}
  else
    $T_{unallocated} = T_{unallocated} \cup \{t\}$
  end if
end for

Algorithm 1: TeSSI auction for the auctioneer

Algorithm 2 presents the *evaluateTask* procedure for the vehicle agents. The inputs are the set of assigned tasks for the agent, S_a, and the task to be evaluated, t. The evaluation set, $T_{evalSet}$, includes the set of tasks in the existing partial schedule and the candidate task. To calculate the agent's bid value, the tasks in the evaluation set are optimized in a single vehicle pickup and delivery optimization model, which is discussed in detail in Section 5.2. For every pickup and delivery optimization task, a solution time limit of 5 s is applied. If the optimal solution is found within the time limit, the optimizer returns the objective value and the optimized schedule of the set of tasks $T_{evalSet}$. The objective value is the *Bid* that the agent bids on task t. The procedure *evaluateTask* generates the value of the bid and the corresponding schedule as output for any agent.

If the time limit is reached before an optimal solution is found, the agent bids the feasible solution obtained at the end of the time limit. In this case, the chance that the task will be assigned to the agent might decrease, as there are other competing agents. If no feasible solution exists or no feasible solution is found within the time limit, the bid of the agent is set to infinity and the agent is withdrawn from the current round of the auction.

Input:
  S_a: Set of assigned tasks for agent a, $a \in A$
  t: Task to be evaluated
Output
  Bid: Bid of task t for agent a, $a \in A$
  $Schedule_a$: Schedule of agent a considering task t, $a \in A$

$T_{evalSet} = S_a \cup \{t\}$
$objective, Schedule_a = optimizerSPDP(T_{evalSet}, a)$
if objective exists **then**
  $Bid = objective$
else
  $Bid = -$
end if

Algorithm 2: Procedure *evaluateTask* for agent a

5.1.3. Handling uncertainties

In real-world aircraft ground handling operations, the duration of executing a certain ground handling task is not fixed. For GSE vehicles, the travel time between their depots and locations on aircraft stands varies, depending on the airside traffic congestion. To deal with the temporal uncertainty, uncertain vehicle traveling time is incorporated in the auction mechanism of the model. To successfully plan the paths for agents, possible traveling delays and waiting times due to conflicts should already be taken into account in the task allocation phase. Considering a constant speed of agents, the traveling time between two airside locations is defined in Eq. (1). The distance buffer coefficient is normally distributed random variable. We use a mean value of 1.4 and a standard deviation of 0.2.

$$distance\ buffer\ coefficient \times shortest\ distance \div speed \tag{1}$$

5.2. Pickup and delivery optimization

To generate a bid on a certain ground handling task in the task allocation process of automated ground handling, an agent needs to evaluate the task-to-bid considering its existing schedule. We use an optimization solver that optimizes the agent's schedule including the existing tasks and the task-to-bid. We model the single agent task scheduling problem as capacitated vehicle routing problem with time windows, pickups and deliveries.

5.2.1. Solving the single agent scheduling problem as a vehicle routing problem with pickups and deliveries

To complete its allocated tasks, a GSE vehicle has to visit a set of pick up and delivery points at different locations of aircraft stands and the vehicle depot, starting and finishing its service within specific time windows at the pickup and delivery locations. The route of a GSE vehicle needs to be optimized by determining the pick-up and delivery times at visited task locations. GSE vehicle has a loading capacity. While traveling between locations outside and inside aircraft stands, it has to pass through specific entrance and exit locations of different aircraft stands. The goal is to minimize the makespan or total traveling time of the GSE vehicle by scheduling a set of tasks, to bid on the candidate task. Although the original problem is a single agent scheduling problem with earliest start time and latest completion time constraints, the required time to complete a task is not only dependent on the processing time, but also affected by the positions of GSE vehicles, their service points and routes between loading and unloading locations.

We present a novel approach by re-designing the single agent task scheduling problem as a vehicle routing problem with time windows, pickups and deliveries. To convert the GSE vehicle scheduling problem to the vehicle routing problem with pickups

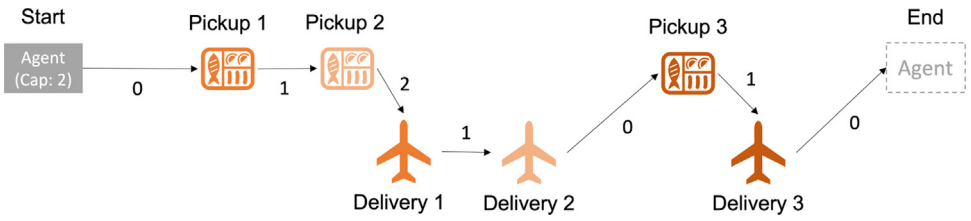


Fig. 8. Representation of a capacitated catering truck (GSE) tour with pick-up and delivery service points, and vehicle loads between visited points.

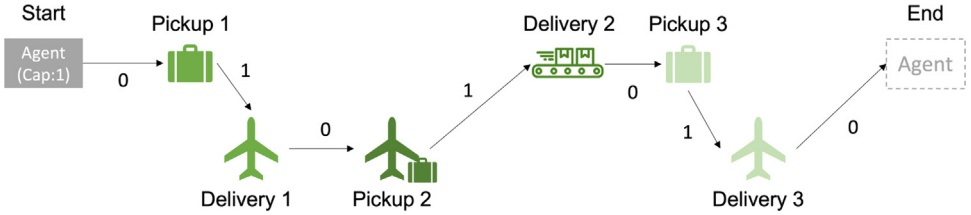


Fig. 9. Representation of a capacitated baggage handling vehicle (GSE) tour with pick-up and delivery service points, and vehicle loads between visited points.

Table 1
The ground handling sub-tasks for GSE vehicles at pick-up locations.

Vehicle	*Pick-up location*	*Task*
RE	terminal	N.A.
CA	terminal	pick up food
BA unload	bay	unload baggage from aircraft
BA load	terminal	pick up baggage from terminal
WA	terminal	fill up water tank of the truck
WC	bay	drain the waste from aircraft lavatory

RE: refueling; CA: catering service; BA: baggage handling; WA: water service; WC: lavatory service.

Table 2
The ground handling sub-tasks for GSE vehicles at delivery locations.

Vehicle	*Delivery location*	*Task*
RE	bay	refuel the aircraft
CA	bay	load new catering galleys to aircraft
BA unload	terminal	unload baggage from vehicle
BA load	bay	load baggage to aircraft
WA	bay	fill the water tank of aircraft
WC	terminal	drain the waste from truck

RE: refueling; CA: catering service; BA: baggage handling; WA: water service; WC: lavatory service.

and deliveries, ground handling tasks are redefined by being split into several pickup and delivery sub-tasks that occur at various loading/unloading locations and service points. The paths between the locations are generated by considering the exit and entrance positions at the bays in addition to the pick up and delivery locations. These paths serve as single links or edges between the task nodes.

For outbound flights, the loading of resources to vehicles is set as a pickup task, while loading or supplying the resources to the aircraft represent the delivery task. Two examples for pickup and delivery representation of ground handling tasks are shown in Figs. 8 and 9. For the catering service task in Fig. 8, loading the food from the terminal to the GSE vehicle and loading the food from GSE vehicle to the aircraft represent the pick up and delivery sub-tasks, respectively. For the baggage handling task in Fig. 9, loading the baggage from terminal and from aircraft to the GSE vehicle are defined as pickup sub-tasks, whereas loading the baggage from GSE vehicle to the aircraft and unloading the baggage from GSE vehicle to be placed on the conveyor belt are modeled as delivery sub-tasks. In Figs. 8 and 9, the values on the links connecting pickup and delivery nodes denote the load of the vehicle while traveling between two nodes.

The pick-up and delivery sub-tasks for different ground handling tasks are given in Tables 1 and 2.

Apart from the makespan, various objective functions can be used to generate the bidding value in the auction model. We use two different objectives, *makespan* and *sum-T*, for bidding in the auction models. The proposed pickup and delivery optimization model is used to generate these objective values for each agent, thus being a sub-problem of the higher level auction model. When the bidding strategy is based on makespan, each agent makes an offer with the minimum makespan of the GSE vehicle that is obtained by solving the pickup and delivery model, using the *makespan* objective, over the set of previously assigned tasks and including the candidate task. In the *sum-T* case, agent offers the minimum *marginal-sum-T* by solving the model using the *sum-T* objective which includes only the traveling time of the GSE vehicle.

Data related to the set of tasks to be evaluated and optimized contains the following information: task identity numbers i, resource (pickup) locations P_i, task (delivery) locations D_i, earliest pickup time Est_{P_i}, latest pickup time Lst_{P_i}, earliest delivery time Est_{D_i}, latest delivery time Lst_{D_i}, pickup duration Dur_{P_i}, and delivery duration Dur_{D_i} of the tasks. The pickup and delivery locations are either the depot of the vehicle or the locations of certain ground handling tasks on aircraft stands. We define pickup and delivery locations and distances between locations based on realistic airport airside operations.

In our model, the focus of the ground handling task executions is on the operations that are performed on the aircraft stands. In this procedure, in some cases delivery sub-tasks, in other cases pick-up sub-tasks are critical due to the strict time windows constraints and being the main task rather than a supporting task such as collecting or delivering an empty vehicle at the depot. To give an example, for a refueling task, the delivery sub-task is more critical than pickup sub-task. Pickup sub-task of a refueling truck has significantly shorter duration and has no strict time window constraint.

5.2.2. Optimization model

We present the mathematical model for optimization of ground handling operations of a single agent at terminal and aircraft bays. This model, which we design by adapting the mathematical model of the single vehicle pickup and delivery

problem (SPDP) presented in a survey of pickup and delivery problems [62], is given as follows:

Parameters. The parameters of the model are as follows:

n: number of pickup vertices
$\tilde{n}$: number of delivery vertices
P: set of pickup vertices, $P = \{1, \ldots, n\}$
D: set of delivery vertices, $D = \{n+1, \ldots, n+\tilde{n}\}$

A virtual start node 0 and a virtual end node $n+\tilde{n}+1$ are also added to the original graph. The problem is modeled on a complete graph $G = (V, A)$, where V is the set of all vertices, and A is the set of all arcs, defined as follows:

V: the set of all vertices, $V = \{0, n+\tilde{n}+1\} \cup P \cup D$
A: the set of all arcs, $A = \{(i,j) : i,j \in V, i \neq n+\tilde{n}+1, j \neq 0, i \neq j\}$
q_i: demand/supply at vertex i; pickup vertices have positive q_i values, while delivery vertices have negative values. The start vertex 0 and the end vertex $n+\tilde{n}+1$ have zero supply/demand values, $q_0 = q_{n+\tilde{n}+1} = 0$
e_i: earliest time to begin service at vertex i
l_i: latest time to complete service at vertex i
d_i: service duration at vertex i
t_{ij}: travel time from vertex i to vertex j
C: capacity of the ground handling vehicle (depends on the type of ground handling task)
S: the set of groups of vertices, $S = \{s_1, \ldots, s_m\}$, where pickup or delivery tasks executed on the same aircraft stand belong to the same group, $s_k \in S$, $k = 1, \ldots, m$

In airport environment, GSE vehicles enter or exit aircraft stands via specified bay entrances and exits. The travel distance is found by computing the length of the path from vertex i to vertex j, which also passes through entrance or exit points in between. For simplicity, we assume that GSE vehicle agents move with unit speed. Therefore, the travel distance and travel time between nodes are equivalent. To align with the design of the path planning algorithm, distances used here are not Euclidean but Manhattan distances.

Decision variables. The decision variables of the model are as follows:

x_{ij}: binary decision variable that is equal to 1 if arc (i,j) is selected for the vehicle's route, or 0 otherwise
Q_i: load of the vehicle after leaving vertex i
B_i: beginning time of service at vertex i

Objectives. The objective of the optimization problem depends on the bidding rule of the auction that is used to allocate the ground handling tasks. The bidding rule of the auction can be *makespan* or *sum-T*. Bidding *makespan* means that the agent bids the total time span of its schedule including its current allocated tasks and the task-to-bid; while bidding *sum-T* means that the vehicle bids the extra travel time if the task-to-bid is included in its current schedule. Below we formulate the objective function for the SPDP based on two bidding rules.

Bid *makespan*:

$$\textit{minimize} \quad B_{n+\tilde{n}+1} - B_0 \tag{2}$$

Bid *sum-T*:

$$\textit{minimize} \quad \sum_{(i,j)\in A} t_{ij} x_{ij} \tag{3}$$

Note that the objective (3) is the total travel time of the vehicle. The marginal travel time is the required extra travel time if the task-to-bid is included in its current schedule. The value of the bid used in the TeSSI auction should be the value of the objective (3) minus the total travel time of the agent's current schedule.

Constraints. The problem is subject to the following constraints:

$$\sum_{i:(i,j)\in A} x_{ij} = 1, \quad \forall j \in V \setminus \{0\}. \tag{4}$$

$$\sum_{j:(i,j)\in A} x_{ij} = 1, \quad \forall i \in V \setminus \{n+\tilde{n}+1\}. \tag{5}$$

$$M(1 - x_{ij}) + Q_j \geq Q_i + q_j, \quad \forall (i,j) \in A. \tag{6}$$

$$max\{0, q_i\} \leq Q_i \leq min\{C, C + q_i\}, \quad \forall i \in V. \tag{7}$$

$$B_i \leq B_{n+i}, \quad \forall i \in P. \tag{8}$$

$$M(1 - x_{ij}) + B_j \geq B_i + d_i + t_{ij}, \quad \forall (i,j) \in A. \tag{9}$$

$$e_i \leq B_i, \quad \forall i \in V. \tag{10}$$

$$B_i + d_i \leq l_i, \quad \forall i \in V. \tag{11}$$

$$x_{ij} = 0, \quad \forall i \in s_k \ \ \forall j \in s_k \ \ \forall s_k \in S. \tag{12}$$

Constraints (4) and (5) ensure that every vertex is visited exactly once. Constraint (6) is a big-M constraint which ensures that the load of the vehicle is updated after visiting a pick-up or delivery node. Constraint (7) makes sure the vehicle's load is not greater than its capacity. In SPDP, pickup and delivery vertices are paired. Therefore, the number of pickup vertices n equals the number of delivery vertices $\tilde{n}$. The pickup vertices are indexed by $i = 1, \ldots, n$ and the corresponding delivery points are indexed as $n+i$. Using start time variables, constraint (8) ensures that for every pickup–delivery pair, the pickup vertex is visited before the delivery vertex. The big-M constraint in (9) ensures that the start time of the task at vertex j is greater than or equal to the completion time of task at vertex i plus the travel time from i to j, if vertex j is visited right after vertex i. Finally, (10) and (11) are added to ensure that the start time and end time of the service are within certain time windows. It is worth noting that by introducing constraints (9), (10), and (11), subtours of vertices are eliminated. Typical subtour elimination constraints can be omitted. Constraint (12) is a special constraint implemented to fit the need of the path planning part in this research. If the pick up and delivery tasks of an aircraft are to be consecutively processed by the same GSE vehicle in the same stand, there exists a waiting time after the completion of the preceding task due to the earliest start time constraint of the successive task. In path planning, having an agent occupying spaces on the aircraft stand for a long period can cause some problems. For example, the agent can block the necessary path of other agents with lower priority, or it can stop the other agent from entering the pickup or delivery location it occupies. Therefore, (12) forbids the consecutive processing of tasks if they belong to the same aircraft. There might be different types of path planning methods where collisions are dealt with using different strategies. In these cases, constraint (12) can be omitted.

By optimization of single vehicle pickup and delivery problem, the schedule for the agent is constructed. A route of a GSE vehicle is a tour of multiple pick up and delivery locations, starting and ending at its depot at the terminal without visiting the depot in between. The values of the decision variables x_{ij} are used in

generating the route or task sequence for the agent. The values of start time variables B_i are the start times of tasks or services at the nodes from which start times of ground handling tasks on an agent can be extracted. The final schedules of agents are formed based on the iterative calls of optimization solver during the TessI auction, till the last candidate is allocated to an agent. These schedules and routes of agents are used as inputs for path planning part of the model.

5.3. Bundling

In TeSSI auctions, tasks are allocated one by one in multiple rounds. The allocation process take long if the number of tasks to be allocated is high. To accelerate the auctioning process, tasks can be bundled before allocation. Agents evaluate the cost of performing all tasks in a bundle and generate a single bid on the bundle. All tasks in the bundle are then allocated to the same agent with the best bid in the same round of auction. Tasks of the same type of ground handling operation, which are also the tasks to be allocated in the same auction, can be combined into bundles. Checking all possible bundles is not applicable in real time due to the high computational time. Rather than exploring all possible combinations, [64] defined the bundle generation problem (BuGP) to generate attractive bundles. The attractive bundles are non-overlapping bundles covering all tasks to be assigned. In this research, we adopt the idea of BuGP which generates non-overlapping bundles. However, instead of applying genetic algorithm, we use a heuristic method to generate the bundles and a reallocation procedure. The aim is to keep bundling procedure less complex since optimization based auction and path planning procedures are already complex in terms of computational requirements.

5.4. Replanning

Aircraft ground handling is operated in a dynamic environment with many uncertainties, and disruptions occur frequently. Removing the baggage of a no-show passenger can delay the baggage loading operation, or malfunctions of GSE vehicles cause delays in ground handling. The disruptions often happen during the handling of the operation, and they cannot be taken into account beforehand in the task allocation phase. The re-allocation of ground handling tasks is needed to deal with operational disruptions in a multi-agent system for automated ground handling. Therefore, we introduced a replanning function to respond to disruptions and performed the experiments simulating a set of disruptions of ground handling tasks. We apply replanning to deal with disruptions as follows: When an allocated task cannot be completed as planned, a disruption occurs. We allow the delayed completion of the disrupted task and reschedule the remaining tasks solving the optimization model with respect to new conditions and repeat the TeSSI auction.

For replanning, the additional variable fixing constraints, (13) and (14), are introduced to the existing optimization model. The new constraints and related parameters are presented as follows:

Parameters

E: Set of directed links that exist on path segment up to and including the disrupted task node, in the original solution.
F: Set of task nodes that belong to the path segment up to and including the disrupted task node.
T_i^I: Start time of task i in the original solution.

Additional constraints for variable fixing

$$x_{i,j} = 1, \quad \forall (i,j) \in E \tag{13}$$

$$B_i = T_i^I, \quad \forall i \in F \tag{14}$$

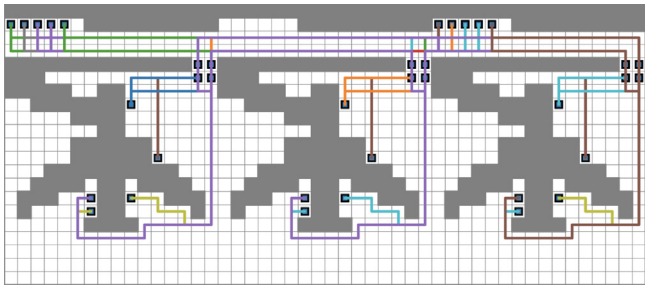


Fig. 10. Trajectories of GSE vehicles between the terminal and bays.

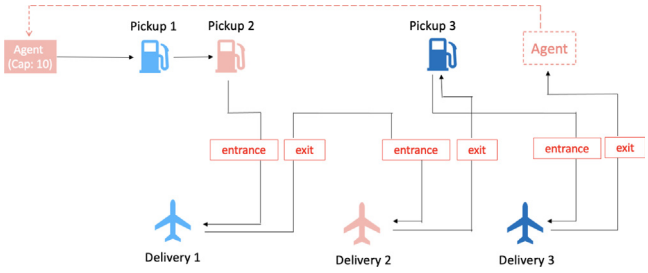


Fig. 11. Pick up and delivery tour with planned paths for refueling vehicle.

The constraint (13) ensures that the value of the binary decision variable, $x_{i,j}$, regarding the directed link from node i to node j is fixed to 1 in the optimization model, if the directed link (i, j) belongs to the path segment that ends with the disrupted task node in the original solution. Thus, the binary decision variable, $x_{i,j}$, cannot take a different value for this node pair and directed link in the new replanning solution.

The constraint (14) fixes the value of the integer decision variable B_i to the start time value, T_i^I, if T_i^I is the start time value of a task belonging to the path segment ending with disrupted solution in the original schedule. Thus, the start time decision variables cannot take different values for the task nodes belonging to the sequence up to and including the disrupted task in the new replanning solution.

6. Path planning

After allocating the ground handling tasks, collision-free paths are found for GSE vehicle agents. Trajectories of GSE vehicle agents between their depot locations at terminal and service locations at aircraft stands are given in Fig. 10.

We modeled the path planning problem for GSE vehicle agents as multi-agent path finding (MAPF) problem. For solving the model, we employed priority based path finding approaches which include SIPP and its adaptations, due to the priority orders of GSE vehicles. We modeled a grid-based path planning environment as depicted in Fig. 10. We solved MAPF for unit-sized vehicle agents on 16 × 16 grid-maps and for square shaped non-unit (large) agents on 64 × 64 grid-maps. To solve the large agent path planning model on 64 × 64 grid-maps, we used an adapted LA-SIPP. Fig. 11 shows the pick up and delivery tour with planned paths for one GSE vehicle.

6.1. Prioritized SIPP

Inspired by SIPP [8] and SIPPwRT [52], we use a *prioritized* SIPP which takes pre-defined priority order as input and transforms the paths of agents with higher priority into (non-)safe intervals for the cells on the map and stores them in a reservation table.

Table 3
Example of precedence setting for GSE vehicles in path planning.

Prec	*GSE*	*t*	*Route*
1	RE_1	t_1	*terminal - bay*$_1$ *- bay*$_3$ *- terminal*
2	RE_1	t_2	*terminal - bay*$_4$ *- terminal*
3	RE_2	t_3	*terminal - bay*$_2$ *- bay*$_3$ *- terminal*
4	CA_1	t_4	*terminal - bay*$_1$ *- bay*$_2$ *- terminal*
5	BA_1	t_5	*terminal - bay*$_1$ *- bay*$_2$ *- terminal*
6	BA_2	t_6	*terminal - bay*$_3$ *- terminal*

RE: refueling; CA: catering service; BA: baggage handling.

6.2. Defining priority orders for GSE vehicles

We set the precedence orders for GSE vehicles as we explain using the example in Table 3. Priorities of GSE vehicles are mainly defined considering the types of tasks they operate. Accordingly, refueling vehicle precedes catering vehicle, catering vehicle precedes baggage handling vehicle, baggage handling vehicle precedes water service vehicle and water service vehicle precedes lavatory service vehicle. The precedence order is s RE $\prec$ CA $\prec$ BA $\prec$ WA $\prec$ WC with respect to vehicle task types. For the vehicles with the same task type, the priority is determined by the departure time, t, of the vehicle, and priorities are adjustable within the same type of vehicle.

In Table 3, the GSE vehicle RE_1 with departure time t_1 precedes the GSE vehicle RE_1 with departure time t_2, since $t_1 < t_2$. Thus, the precedence orders for RE_1 and RE_2 are respectively $prec = 1$ and $prec = 2$, where $1 \prec 2$. Similarly, the priorities of the vehicles RE_1, RE_1, RE_2 with $prec = 1, 2, 3$ are higher than the priorities of the vehicles CA_1, BA_1, BA_2 with $prec = 4, 5, 6$, due to the vehicle types and departure times.

6.3. Prioritized SIPP for path planning of GSE vehicle agents

We form the trip itineraries for GSE vehicle agents, based on the allocated ground handling tasks. The trip itinerary shows the start location, the intermediate stops, and the final destination for an agent. For example, a trip itinerary for a refueling truck agent might be: *depot* $\rightarrow$ *bay*$_1$ *entrance* $\rightarrow$ *refueling location at bay*$_1$ $\rightarrow$ *bay*$_1$ *exit* $\rightarrow$ *depot*. Every trip segment in the itinerary belongs to a separate part in the modeling environment, either the service road environment or one of the bay environments. Every trip segment is a path finding problem for the associated agent. To plan paths for all involved GSE vehicle agents, the trip itineraries are ordered by the type of GSE vehicle used. We sort the trips with respect to their start times if they belong to the same type of GSE vehicle. Algorithm 3 shows the prioritized SIPP approach we used to solve automated ground handling problem.

The inputs of Algorithm 3 are the set of agents $A = \{a_1, a_2, \ldots, a_n\}$, the trip itineraries G_a for agents, the map of the service road and the aircraft stand, and the safe intervals of the cells on the service road and the aircraft stand. According to the priority order for different types of GSE vehicles, paths for agents are planned. Paths in the itinerary of the same agent are planned in sequential order. The paths either belong to the service road environment or one of the aircraft stands. Corresponding maps and safe intervals are used as the input for SIPP. Once a path is planned, it is stored in P_a, and the safe intervals of the grids in the corresponding environment are updated. The outputs of the model are sets of paths P_a for agent a.

In the solution in Fig. 11, GSE vehicle first visits the points *pickup* 1 and *pickup* 2 at the terminal to pick up resources and goes to *delivery* 1 and *delivery* 2 locations at BAY_1 and BAY_2. From BAY_2, it goes to the location *pickup* 3 at terminal to pick up new resource and continues to delivery location *delivery* 3 at BAY_2, and returns back to terminal. We show the path planning solutions

Input:
  A: Set of agents
  G_a: Sets of trip segments (itinerary) for agents a, $a \in A$
  M_{road}: Map of the service road
  M_{bay}: Map of the aircraft stand (bay). It is identical for all bays
  S_{road}: Safe intervals of grids on the service road
  S_{bay_b}: Safe intervals of grids on the b^{th} aircraft stand (bay)
Output
  P_a: Sets of paths for agent a, $a \in A$

Initialize S_{road}, S_{bay}
for $a \in A$ **do**
  for *segment* $\in G_a$ **do**
    if *segment* belongs to the service road **then**
      $path = SIPP(segment\ start, segment\ goal, M_{road}, S_{road})$
      add *path* to P_a
      $S_{road} = updateSafeInterval(path, M_{road}, S_{road})$
    else if *segment* belongs to an aircraft stand **then**
      b = id of the aircraft stand
      $path = SIPP(segment\ start, segment\ goal, M_{bay}, S_{bay_b})$
      add *path* to P_a
      $S_{bay} = updateSafeInterval(path, M_{bay}, S_{bay})$
    end if
  end for
end for

Algorithm 3: Prioritized SIPP

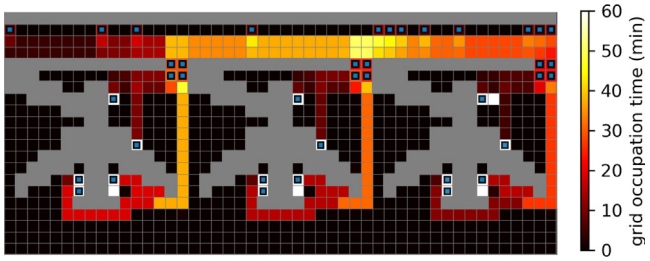


Fig. 12. Heat map of planned paths in the *basic* model.

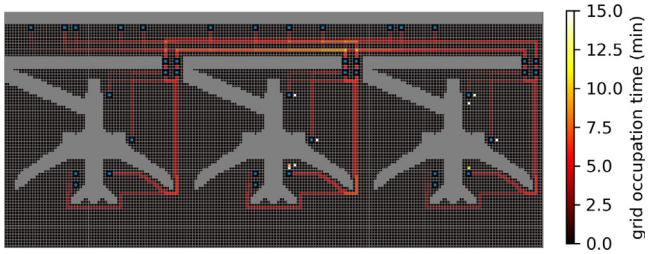


Fig. 13. Heat map of planned paths in the *extended* model.

for all GSE vehicles over a set of simulations using heatmaps, which are presented in Figs. 12 and 13 on 16 $\times$ 16 and 64 $\times$ 64 grid-maps in Section 7.4.1.

6.4. LA-SIPP for path planning of large GSE vehicle agents

In real-world aircraft ground handling, GSE vehicles have different geometry shapes. In the environments for aircraft stands, GSE vehicle agents can occupy multiple cells. In the *extended* model on 64 $\times$ 64 grid-map, we included the squared shape agents with non-unit sizes. To deal with such agents, some adjustments to SIPP are needed. [52] proposed the concept of time offsets in the SIPPwRT algorithm that is able to deal with circular agents of different sizes. We apply similar ideas and generalize them to agents with square shapes. The algorithm is called

Table 4
Randomly generated turnaround times for three aircraft stands in normal scenario.

Flight no	Aircraft stand 1 $[t_a, t_d]$	Aircraft stand 2 $[t_a, t_d]$	Aircraft st. 3 $[t_a, t_d]$
1	[7, 52]	[35, 77]	[33, 83]
2	[52, 92]	[77, 119]	[83, 129]
3	[92, 134]	[119, 168]	[129, 172]
4	[134, 180]	[168, 213]	[172, 222]
5	[180, 229]	[213, –]	[222, –]
6	[229, –]	[–, –]	[–, –]

Table 5
GSE vehicle types and codes.

GSE vehicle type	GSE code
refueling truck	RE
catering truck	CA
baggage handling vehicle	BA
water service vehicle	WA
lavatory service vehicle	WC

Table 6
Minimum sizes of GSE resources for each type of ground handling task to complete the operations for one aircraft, in the original ground handling model.

GSE code	Task type	Required number of GSE
RE	refueling	1
CA	catering	1
BA	baggage handling	1
WA	water service	1
WC	lavatory service	1

Table 7
The tasks GSE agents operate and bid on in the original model.

GSE code	Tasks to bid on
RE	refueling tasks
CA	catering tasks
BA	baggage handling tasks
WA	water service tasks
WC	lavatory service tasks

large agents safe interval path planning (LA-SIPP). In LA-SIPP, the search for state space is similar to that of the original SIPP. Elements of LA-SIPP that differ from SIPP are the generation of successors and the way that the safe intervals are updated.

7. Results

Experimental design is based on the assumptions discussed in Section 4.1 and the model environment specifications presented in Section 4.2. For the task allocation experiments, the size of aircraft stand map was not considered as a critical parameter, thus the environment was represented by a 16 × 16-grid. Path planning experiments were performed using both the basic 16 × 16-grid and the extended 64 × 64-grid. Moreover, a sensitivity analysis is included.

7.1. Instance generation and instance features

We explain the generation of datasets and instance characteristics in Sections 7.1.1, 7.1.2, 7.1.3, 7.1.4, 7.1.5, 7.1.6, and 7.1.7. The parameters to generate instance sets were selected in consultation with a major European airline.

7.1.1. Flight schedules

We generate three types of flight schedules based on frequent, normal, less-frequent flights during 4-hour time window, and the operations of these flights are handled in three aircraft stands. Once the previous aircraft leaves, an aircraft can enter the aircraft stand and the turnaround starts. We generate the turnaround times of the flights using three uniform distributions, U(50,60), U(40,50), U(30,40), of time intervals in minutes. This random generation leads to 13 flights in off-peak scenario, 16 flights in normal scenario, and 20 flights in busy scenario. An instance of randomly generated turnaround times for three aircraft stands within 4-hours in normal scenario is given in Table 4.

7.1.2. Vehicle types, tasks, and sub-tasks

In problem instances, we include five types of GSE vehicles which are listed in Table 5.

The number and type of vehicles required to perform the ground handling tasks for one short-haul flight are given in Table 6. This shows the most limited (and necessary) resource case for the original ground handling problem. In general, a fleet can contain several GSE vehicles with the same type that can be used at various times or locations.

In the original ground handling problem, each type of GSE vehicle can operate and bid on only one type of task. These are shown for the original ground handling model in Table 7.

There are several tasks with the same task type since there are several aircraft occupying the stands or several flights during a day. Thus, the GSE vehicle can bid for any of the suitable task types but not allowed to bid for a different task type. For example, when there are several aircraft, catering truck (CA) can operate several catering tasks at various bays or at different times at the same bay, as long as the task types are catering and unless the load exceeds its capacity.

When we convert the problem into vehicle routing problem with pick-ups and deliveries, we redefine the task types each GSE vehicle can handle, since we split the original tasks into pick-up and delivery sub-tasks. In this case, a catering truck (CA) can operate catering pick-up sub-tasks at specific pick-up locations at the terminal and catering delivery sub-tasks at catering positions around the aircraft at the bays. Similarly, a refueling truck (RE) and water truck (WA) pick up fuel and water at pick-up locations at the terminal and deliver the fuel and water at delivery locations at aircraft bays. We redefine the baggage handling truck (BA). One type of baggage handling vehicle picks-up the baggage from the aircraft at its pick-up location at the bay and delivers it at its delivery location at the terminal, while another type picks-up the baggage from its pick-up location at the terminal and delivers it to aircraft at its delivery location at the bay. In Table 8, we present the list of the GSE vehicle types and the sub-task types they operate, in the *transformed model*, which is the pick-up and delivery version of the ground handling model.

7.1.3. GSE fleet size

In the instances, except for the sensitivity analysis, we set the minimum fleet size required to complete the services for one aircraft as available resource. The numbers of vehicles for each GSE type in this fleet are given in Table 9. The fleet is reused to serve several aircraft at the same bay for different flights or one type of vehicle can serve several aircraft at different bays. We aimed to test the most limited resource availability case by setting the available fleet size to minimum required value. This instance type is the hardest one to find feasible solutions. Increasing the fleet size, leads to solutions that are easier to find and have better quality, in sensitivity analysis in Section 7.5.

7.1.4. Number of tasks to bid for each GSE vehicle

The number of tasks that vehicles can bid on depends on the number of flights, which is a variable in the simulation. For

Table 8
GSE tasks in pick-up and delivery model, *transformed model*.

GSE	Task	Sub-task	Type	Location
RE	*refueling*	*load fuel to GSE*	pick-up	terminal
RE	*refueling*	*load fuel to aircraft*	delivery	bay
CA	*catering*	*load food to GSE*	pick up	terminal
CA	*catering*	*load food to aircraft*	delivery	bay
BA	*baggage handling*	*load baggage to GSE*	pick-up	terminal
BA	*baggage handling*	*load baggage to aircraft*	delivery	bay
BA	*baggage handling*	*unload from aircraft*	pick-up	bay
BA	*baggage handling*	*unload from GSE*	delivery	terminal
WA	*water service*	*load water to GSE*	pick-up	terminal
WA	*water service*	*load water to aircraft*	delivery	bay
WC	*lavatory service*	*drain waste from aircraft*	pick-up	bay
WC	*lavatory service*	*drain waste from GSE*	delivery	terminal

Table 9
Number of available GSE vehicles for each type in simulations.

GSE vehicle type	Number of available vehicles
refueling truck (RE)	2
catering truck (CA)	2
baggage handling vehicle (BA)	4
water service vehicle (WA)	2
lavatory service vehicle (WC)	2

Table 10
Number of tasks the available GSE vehicles can bid on: An example for the normal scenario with 16 flights within 4-hours time window.

GSE resource	Tasks to bid on for 16 flights in normal scenario
2 RE vehicles	16 refueling tasks to bid on
2 CA vehicles	16 catering tasks to bid on
4 BA vehicles	16 baggage loading and 16 baggage unloading tasks to bid on
2 WA vehicles	16 water service tasks to bid on
2 WC vehicles	16 lavatory service tasks to bid on

example, in the normal scenario, there are 16 flights in the 4-hour-planning window, and the numbers and types of tasks that available GSE vehicles can bid on in this planning window and traffic scenario are given in Table 10.

7.1.5. Locations of GSE vehicles

The source locations (depots) of GSE vehicles at the terminal are randomly selected cells connected to the service road on the grid-maps. The service locations of GSE vehicles at aircraft stands are specific standard positions around the aircraft. Each GSE vehicle type has to serve the aircraft at a certain location, which is dissimilar to the service locations of other types of GSE vehicles. We have defined these locations considering the service locations for the aircraft type, Boeing 737.

7.1.6. Distance buffer coefficient

The distance buffer coefficient handling uncertainties on traveling times of GSE vehicles is Y, where $Y \sim N(1.4, 0.2^2)$. Although the assumption for the GSE vehicles was to travel one unit step at a time, we used this coefficient to consider the variable travel times due to uncertain conditions.

7.1.7. Other features

The turnaround of the next flight starts immediately after the turnaround time of the previous flight ends. The duration of ground handling tasks were identical for all flights. The time windows were determined based on the arrival times of flights.

7.2. Experimental setup

For the experiments we define eight simulation groups as given in Table 17. These simulation groups include the tests with normal, busy, offpeak scenarios with no bundling or replanning, normal scenarios with bundling, normal scenarios with disruption and replanning, normal scenarios with disruption and no replanning, scenarios with the objectives of minimizing the *makespan* and $sum - T$, and the scenarios with the *basic* and *extended* environments. In Experiment A, we ran the experiments on simulation sets I, II, III, to see the performance of the algorithm for normal, offpeak, busy scenarios on the *basic* environment with the objective *makespan*. In Experiment B, we compared the results for simulation sets I and IV, to see the effect of bundling compared to no bundling, for the normal scenario on the *basic* environment with *makespan* objective. In Experiment C, we ran the tests on simulation sets V and VI, to compare the cases (i) when a disruption occurred and no replanning was used, and (ii) when a disruption occurred and replanning was used, in the normal scenario with the *basic* environment and *makespan* objective. In addition to experiments A, B, C, we ran tests on simulation sets VI and VII, to compare the results when the *makespan* and $sum - T$ objectives were aimed to be minimized on the *basic environment* in the busy scenario. In path planning experiments, we repeated the tests on simulation set VIII on the *extended environment*, in the normal scenario with *makespan* objective. Furthermore, for sensitivity analysis, we compared the results by increasing the number of available GSE vehicles, and changing the processing times of ground handling tasks.

We reported the obtained objective function values *makespan* and $sum - T$, CPU time, *allocation rate*, *path length*, *path duration*, *success rate*, *agent on-time rate*, *task on-time rate* when applicable. We compared the collision-free path planning results with the shortest paths with collisions, which we used as *baseline* for solution quality.

For each simulation set, we repeated the simulations over a set of 100 instances. We set the number of simulations based on preliminary stability tests provided in Section 8.1.

Gurobi 9.5.0 was used to solve the pickup and delivery optimization problem for each agent during the bidding phase. Simulations were conducted on an 8-core Apple M1 chip with 8 GB RAM. For each pickup and delivery optimization, the solution time limit was 5 s.

7.3. Task allocation experiments

Experiments were performed to investigate the model performance for different traffic demand scenarios and task allocation objectives. The performances of the task bundling and replanning were also evaluated by the experiments. These experiments include the Experiments A, B, C, and experiments regarding the objective functions. For each experiment, 100 simulation runs were performed on the simulation sets I, II, III, IV, V, VI, VII. We determined the number of simulations after performing a large number of simulations until the *coefficient of variation* of simulation outputs reach to a stable level. The stability was reached after

Table 11
The median values of simulation outputs over 100 simulation runs on different instances for each type of ground handling task, and the overall results based on mean and total values over the outputs of all GSE vehicles.

Task (s)	*Makespan* (minutes) *median over 100 inst..*			*Allocation rate* *median over 100 inst.*			*CPU time* (seconds) *median over 100 inst.*		
RE	142.53	202.62	166.13	0.9231	0.8750	0.7500	18.38	20.29	43.09
CA	138.65	220.47	209.07	1.0000	1.0000	0.9500	2.7800	15.08	43.16
BA	203.48	221.05	218.71	1.0000	1.0000	1.0000	0.90	1.32	2.08
WA	140.12	199.97	218.15	1.000	0.875	0.7500	7.73	32.20	63.50
WC	144.25	222.88	191.88	1.0000	0.9375	0.8000	9.10	16.70	40.53
	offpeak	normal	busy	offpeak	normal	busy	offpeak	normal	busy
	mean makespan *mean over all tasks*			*mean allocation rate* *mean over all tasks*			*total CPU time* (seconds) *total over all tasks*		
overall	153.81	213.40	200.79	0.9846	0.9375	0.8500	38.89	85.59	192.38
	offpeak	normal	busy	offpeak	normal	busy	offpeak	normal	busy

100 simulation runs in most of the tested scenarios. We provided these stability tests in Section 8.1. In Table 11, Table 12, Table 13, and Table 14, the results related to different types of ground handling tasks are the median values retrieved from the outputs of 100 simulations runs each of which were performed on different instance sets. The results regarding the overall model performances are the mean of medians of makespans and the mean of medians of allocation rates, over all types of ground handling tasks, and the total CPU time that is the sum of computational times of all different types of tasks.

7.3.1. Experiment A: Traffic demand scenarios

To test scenarios of different traffic demand levels, three different flight schedules were used. We use turnaround times of 50 to 60 min, 40 to 50 min, and 30 to 40 min to simulate the traffic demand of offpeak hours, normal hours, and busy (peak) hours in an airport terminal. A longer turnaround time does not necessarily mean that the ground handling tasks take longer to perform. It was used to represent longer waiting time between the arrival of two successive flights because the task duration and the time windows of ground handling tasks were consistent among all flights. For three aircraft stands and four-hour planning window, there are 13, 16, and 20 flights in the offpeak, normal, and busy hours schedule, respectively.

Table 11 shows the median values of the simulation outputs, *makespan*, *allocation rate*, *CPU time*, over 100 problem instances for each type of ground handling task and the overall performance of the model. The overall performance outputs are the mean values of medians of makespans over all task types, the mean values of medians of task allocation rates over all task types, and the total computational times over all task types, for different traffic scenarios.

The *mean makespan* of the normal scenario was higher than that of the offpeak scenario in Table 11. However, the *mean makespan* of the busy scenario was lower than that of the normal scenario. The reason is that more tasks were able to be allocated to some GSE vehicles in the normal scenario compared to the busy scenario.

The *mean task allocation rates* in the scenarios with busier schedules were lower (Table 11). Because, the number of flights was higher in busier scenarios, while the number of available GSE vehicles remained the same.

The *CPU times* in Table 11 are affected by increasing the number of flights and ground handling tasks. For the offpeak scenario, allocation of tasks for all flights in the 4-hour planning window took around 38 s. The *total CPU* time (total over all tasks' median CPU times over 100 instances) increased to 85 s, when the number of flights increased from 13 to 16, and to 192 s, when the number of flights increased from 16 to 20.

7.3.2. Experiment B: Bundling

Allocating bundles of tasks reduces the number of auction rounds. However, the allocated task sets with bundling would be different than the sets allocated without bundling. This can also affect the solution quality and CPU time. We tested whether bundling tasks accelerates or improves the task allocation procedure or not. Bundles of tasks were assigned using TeSSI auctions to evaluate the makespan, allocation rate and CPU time. In bundling, tasks in the same bundle cannot be allocated to different agents. This may affect the optimization of the task allocation and lead to changes in obtained makespan. Bundles are formed using the bundle generation heuristics, taking temporal and spatial proximity among tasks into account. Tasks bundles that cannot be assigned to any agents are broken down into individual tasks and these tasks are allocated in the last few rounds of the auction. Agents with allocated bundled task sets have less flexibility in their schedules.

Table 12 shows the simulation results for bundling. The median values of makespans, allocation rates, and CPU times are presented for bundling and not bundling. The *median makespan* was higher for each GSE vehicle in the allocation with bundling. Bundling has not improved the makespan. Bundling of tasks increases the solution time of pickup and delivery optimization and for some task bundles the agents lose the opportunity to bid. The *median task allocation rate* was lower with bundling for the refueling truck and the lavatory service vehicle. *median task allocation rate* was higher with bundling for the water service truck. *median task allocation rate* was equal to 1 for the baggage handling truck and for the catering truck. For these vehicles, the CPU times are also less than the other vehicles. Task allocation rates depend on the performance of SPDP optimization to generate bids within the solution time limit. The CPU times with bundling are the CPU times of task allocation plus bundle generation. However, the higher CPU times with bundling were not mainly caused by generation of bundles. Finding feasible positions takes more time for the pickup and delivery optimizer for bundled tasks. The time windows that were originally generating cuts and leading to faster convergence become tighter, thus less effective with bundling. This makes it more difficult for the optimizer to find feasible solutions within limited time or it takes longer to find feasible or optimal solutions. Therefore, although bundling reduces the number of auction rounds, since we use an optimization based hybrid auction method, we do not observe an improvement in CPU time.

7.3.3. Experiment C: Replanning

To deal with the disruptions of the ground handling operations, a replanning function was implemented. We evaluated the performance of the replanning function by simulating two scenarios with the same disruptions. In one of the simulation sets,

Table 12
Median values of simulation outputs of 100 instances, for each GSE vehicle, with and without bundling.

Task	*Makespan* (minutes)		*Allocation rate*		*CPU time* (seconds)	
	No bundle	Bundle	No bundle	Bundle	No bundle	Bundle
RE	202.62	237.87	0.8750	0.8125	20.29	52.78
CA	220.47	224.14	1.0000	1.0000	15.08	22.02
BA	221.05	241.88	1.0000	1.0000	1.32	3.55
WA	199.97	218.82	0.8750	1.0000	32.20	36.25
WC	222.88	229.38	0.9375	0.8125	16.70	40.10

Table 13
Median values of simulation outputs of 100 instances, for each GSE vehicle, with and without replanning.

Task	*Makespan* (minutes)		*Allocation rate*	
	Replan	No replan	Replan	No replan
RE	221.68	199.72	0.9375	0.8750
CA	220.73	223.63	0.9375	1.0000
BA	233.75	225.02	1.0000	1.0000
WA	200.88	200.12	1.0000	0.8750
WC	222.95	219.38	0.9375	0.9375

the allocation of tasks remained unchanged. The completion time constraints of all tasks that had not been performed or had partly been performed at the time of the disruption were prolonged by the length of delay. In another simulation set, the replanning function was activated and task allocation was repeated for a group of tasks if disruptions occurred. The function works as described in Section 5.4. In the simulation scenario, for every ground handling task, the probability of being disrupted was 20% and in the event of a disruption, the processing duration of the task was increased by 20%.

The simulation results of test scenarios with and without the replanning are shown in Table 13. The median values of makespans were lower in the replanning scenario for the catering task. For all other ground handling tasks, the median values of the makespans were higher. This might be due to the higher allocation rates in the replanning scenario. Except for the catering tasks, the task allocation rates after replanning were larger than or equal to the allocation rates of the scenario with no replan.

7.3.4. Exploring the task allocation objectives

Two objectives to be minimized, *makespan* and *sum-T*, were tested as bidding rules in the adapted TeSSI auction for task allocation, in this experiment. The comparison of the simulation results using the objective *makespan* and *sum-T* is shown in Table 14. The *median allocation rate* was smaller with the objective *makespan*, except for the catering and baggage handling. Using different objectives, the agents bid different values on the same tasks, resulting in different task allocation. The *median CPU time* was lower for allocation of the catering and baggage handling tasks, compared to the others, using the objective *makespan*. This is consistent with the higher *median allocation rates* of catering and baggage handling tasks, using the *makespan* objective. For both objectives, the *median CPU time* of allocating baggage handling tasks was the lowest.

7.4. Path planning experiments

We kept all settings in Section 7.3 unchanged to evaluate the performance of the path planning algorithm. We used the result of task allocation as input, to plan paths of the GSE vehicles. We ran the simulations in the *basic* and the *extended* modeling environments, as defined in Section 4.2. In the *basic* model, each vehicle agent has unit size. In the *extended* model, each agent occupies a space of 3 × 3 grid. Agents move one cell per time step, which is equivalent to a speed of 4 m/s in our environment setting.

Table 14
Median values of simulation outputs of 100 instances, for each GSE vehicle, for different objectives.

Task	*Allocation rate*		*CPU time* (seconds)	
	Makespan	Sum-T	Makespan	sum-T
RE	0.8750	1.0000	20.29	4.06
CA	1.0000	1.0000	15.08	23.78
BA	1.0000	1.0000	1.32	1.48
WA	0.8750	1.0000	32.20	21.40
WC	0.9375	1.0000	16.70	13.37

Table 15
Mean values of performance indicators over 100 instances for path planning.

Performance indicators	*Basic model*	*Extended model*
path length (m)	400.63	372.76
path duration (s)	128.45	129.09
average delay per agent (s)	22.28	37.96
average delay per task (s)	15.49	27.66
agent ontime rate	0.94	0.93
task ontime rate	0.96	0.95
agent success rate	1.00	0.98
computational time (s)	5.11	86.83

Table 16
Solution quality of collision-free path planning compared to *baseline*, shortest paths with collisions.

	Suggested method	*Baseline*
	collision free path planning	*shortest path with collisions*
PI →	path length (m) *mean over* 100 *instances*	shortest path length(m) *mean over* 100 *instances*
basic m.	400.63	400.15
extended m.	372.76	372.23
PI →	path duration (s) *mean over* 100 *instances*	shortest path duration (s) *mean over* 100 *instances*
basic m.	128.45	100.04
extended m.	129.09	93.06

We investigated the length and duration of paths for the GSE vehicle agents in both models. For both models, the mean values of the path length and duration in the performed simulations are listed in Table 15. The shortest possible path length and duration without considering conflicts with other agents are used as baseline in Table 16 to prove the solution quality of the outputs obtained by suggested collision free path planning. The closer the collision free path lengths to the shortest path lengths with collision is, the higher the quality of the path length obtained by path planning algorithm, since it means that collision free solutions were able to be found without deviating significantly from the shortest paths. Similarly, the closer the collision free path duration to shortest path duration is, the higher the quality of solution of collision free path planning algorithm, since it means that collisions were avoided without having to employ too many waiting moves to avoid collisions.

Although the speed was constant for all agents, the *path duration* in the results was usually longer than the minimum time the agents needed to complete traveling. This means that the agents stopped and waited on their route to avoid conflicts with other agents. For the *basic* and the *extended* models, the lengths of paths were respectively 1.0012 and 1.0014 times the shortest path length. That is, the obtained traveling distances of agents considering conflicts were barely longer than the shortest possible

distances without considering conflicts. Thus, the path planning algorithm is able to generate good solutions in our problem setting. The path durations are 1.28 and 1.39 times the shortest durations. Compared to the ratio of (actual) path length to shortest path length, the ratio of (actual) path duration to shortest duration is higher. This result discloses that agents tend to stop and wait on the path while encountering conflicts instead of taking detours. This behavior can be interpreted as follows: Substitution paths might not exist in the environment. Also, it is possible that the obstacles (agents with higher priorities) only occupy the paths for a short period, so agents are willing to wait rather than detour.

In Table 15, we also showed the average delay per agent and the average delay per task. For both models, the average delays per agent were higher than the average delays per task. Similarly, the ratio of tasks with no delays, the task on-time rate, was higher than the ratio of agents with no delays, the agent on-time rate. These results indicated that agents were assigned multiple tasks, and delays occurred on multiple tasks of agents. The average task delays of 15 and 28 s for the *basic* and the *extended* models were within acceptable ranges for online use. The high on-time rates were also sufficient for real-world cases.

The algorithm used for path planning is a decoupled approach. As paths for agents are planned one after another based on the priority order, complete solution is not guaranteed. The success rates that agents can successfully find paths in the modeling environment are also listed in Table 15. Although the success rate is not 1 in the *extended* model, it is still quite high. The last row in Table 15 presents the *mean CPU time* for both models.

The maps, distribution of static obstacles, service locations, paths lengths from GSE depots to service points, locations of GSE depots, ratio of agent sizes to available moving spaces, and the ratio of agents' time step to their sizes are slightly different in the proposed *basic* and *extended* environments, thus performance indicators are not comparable for these models. Similarly, the standard grid-maps used in previous works [7,65] are not comparable to our environment setting. We present a brief discussion on the benchmark models and algorithm performances in Appendix A.

7.4.1. Analysis of results for path planning using heat maps

We analyzed the results of path planning algorithms for GSE vehicles using heat maps. Figs. 12 and 13 show the heat maps of the planned paths for the *basic* and *extended* models with 16×16 and 64×64 grid-maps. The blue squares in Figs. 12 and 13 show the depots of GSE vehicles, the entrances and exits of bays, and the ground handling service locations. The color bar besides each figure show the average times that the grids were occupied over 100 simulation runs. The number of simulations were determined by measuring the *coefficient of variation* of results over a large number of simulations for each GSE vehicle individually, and for the overall multi-agent system including all vehicles. In most of the instances, the *coefficient of variation* became stable after 100 simulation runs. Therefore, for each simulation set, we repeated simulations for 100 runs with a variable set of instances and reported all outputs as mean values over 100 simulation runs.

In the generated heat maps, the locations where the ground handling tasks took place were where the GSE vehicle agents stayed the longest. The intersections between the service road and bay areas were also the places that were occupied most. Also, the middle section of the service road connecting the two bay entrances on the left was an area with busy traffic. The heatmaps show the frequency of occupation of cells by agents, but they do not show the waiting times of agents. The cells with brighter colors on the maps indicated that the most of the agents took the shortest possible routes instead of taking detours.

Some cells next to the task locations, especially the baggage handling locations, also had high occupation rates in the heat map. For a baggage handling vehicle, the reason was usually the early arrival of a baggage loading vehicle at task location, thus its waiting in front of the task location, till the baggage unloading task is complete, before it can enter the cell. For other types of vehicles, the vehicles which were already assigned new tasks for the aircraft of next flight in the same aircraft stand could prefer to occupy the cells while waiting for the next aircraft, instead of leaving the cell and the bay.

7.5. Sensitivity analysis

In the previous experiments, the numbers of available ground handling vehicles were fixed. Also, consulting experts in ground handling operations, we had used the smallest possible numbers of ground handling vehicles that are able to perform the tasks for all flights within the planning window. The numbers of different types of GSE vehicles represent the limited resources in task allocation. The more the resources are available, the more flexible the allocation is. Therefore, the numbers of available GSE vehicles are important factors in the allocation of ground handling tasks. Another important factor that affects allocation is the execution duration of tasks. We provide sensitivity analyses on these factors.

7.5.1. Analysis on number of available GSE vehicles

Changing the number of GSE vehicles, we investigated three main performance indicators: makespan, task allocation rate, and computational time in different traffic demand scenarios.

Figs. B.15, B.16, B.17 show the simulation results when we change the number of available GSE vehicles. The number of available vehicles for the refueling, catering, water service, and lavatory service is set to be 2, 3, or 4 for each task. The number of available baggage handling vehicles is set to be 4, 6, or 8 for sensitivity analysis, since both unloading and loading operations are needed for a flight. The analyses show that the model is sensitive to the number of available GSE vehicles. By increasing the number of available vehicles, makespan and CPU time were decreased and the allocation rate was increased, in all scenarios. The analyses are presented in Appendix B.

7.5.2. Analysis on task processing duration

The refueling tasks have the most variable task duration. Refueling time depends on the destination of the flight and whether the aircraft carries fuels for the return flight or not. For European flights departing from Amsterdam, refueling duration ranges from 10 to 25 min. In the simulations, we changed the refueling duration from 10 to 25 min in offpeak, normal, busy scenarios, for 2, 3, and 4 available refueling vehicles. The *makespan* was used as the bidding rule, and for each duration-scenario-fleet combination 100 simulation runs were performed.

The result is shown in Fig. 14. The colorbar in Fig. 14 shows the maximum vehicle makespan. For each scenario, the maximum makespan values tend to be higher with smaller number of vehicles, since more tasks are allocated to each vehicle. Higher processing time increases the makespan for any vehicle, however, the incremental change is smaller with higher number of vehicles. The change in makespan also depends on optimization of allocation in flexible scenarios where there is room to rearrange the order of tasks.

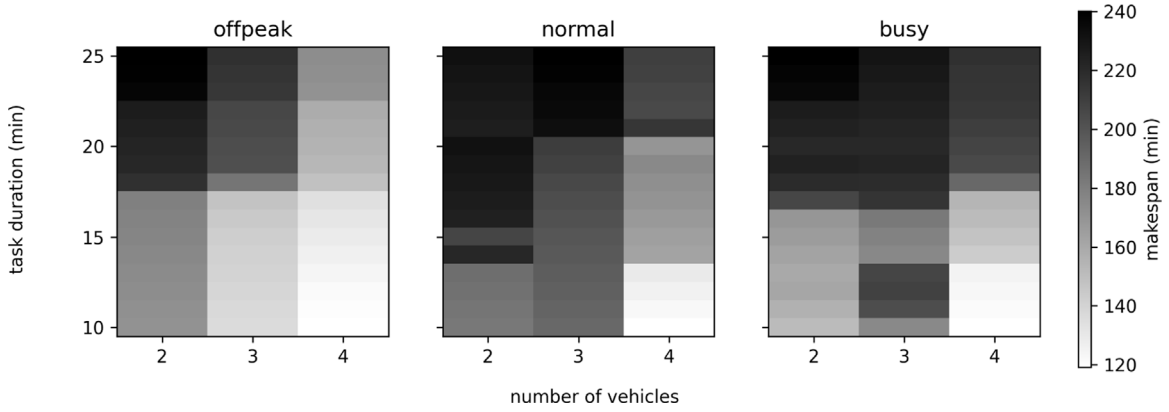


Fig. 14. Contour plot of maximum vehicle makespan with the *makespan* bidding rule for three traffic demand scenarios, with varying numbers of available vehicles and task duration.

7.6. Scalability

With higher number of ground handling tasks and limited number of available vehicles, the problem is more complex. The number of ground handling tasks increases when the number of flights within a specified time window is high. Considering the operations at several bays and the operations for several flights rather than one aircraft also increases the complexity.

The scalability of our framework is measured by its performance when the number of flights and ground handling tasks at several bays are high within a short time window and the GSE fleet size is small.

The problem of task allocation and path planning is more complex than the NP-hard VRP, since the traveling is also affected by the collisions with other vehicles. In ground handling, there are also the static obstacles such as the aircraft and PBB.

We have not aimed to solve very large sizes of problem instances, however we proposed a framework that can handle the operations in realistic conditions, efficiently.

Given the minimum GSE fleet size, the proposed framework generates solutions for frequent, normal, and less-frequent flights in 4-hour planning window, with high success rates. In frequent scenario, there were 20 flights in 4 h and 20 main tasks to bid on for each vehicle. The number of sub-tasks are higher than the number of main tasks, and the SPDP optimization is solved for a number of pick-up and delivery sub-tasks at each round of the auction, for each GSE vehicle. In the frequent scenario, the overall allocation rate and CPU time were 0.8500 and 192 s, respectively. Considering he allocation rates in high frequency scenario and short CPU times with high success rates in path planning, the scalability is good enough to automate the planning and operations to handle the ground handling tasks of various flights in one pier of the major hub airport.

8. Statistical analysis

8.1. Coefficient of variation

The model outputs obtained by the experimental analysis are not normally distributed. To determine the minimum required number of simulations we used the measure of *coefficient of variation*, $c_v = \sigma/\mu$. The *coefficient of variation* is often applied in agent-based simulations with a non-normal distribution of outputs [66]. For each simulation set, we performed a high number of simulation runs until the *coefficient of variation* stabilize. In most of the experiments the *coefficient of variation* stabilized when a number of 100 simulation runs was reached. The plots of the coefficients of variation over a number of simulation runs are presented for simulation set I in Figs. C.18, C.19, and C.20 in Appendix C. The reported results are for the outputs, *makespan*, *allocation rate*, and *CPU time*, for each one of the GSE vehicle agents, RE, CA, BA, WA, WC, and for the overall outputs regarding all GSE vehicles. The simulation sets, I to VIII, for which we ran these tests are described in Table 17.

8.2. Statistical significance

To test the statistical significance, we used the non-parameteric Wilcoxonsigned-rank test as the values retrieved from experiments are paired but not normally distributed. The statistical significance results were reported as p-values and the effect sizes of all pairs of experiments were reported as Vargha–Delaney A-values. The A-values reveal the probability that the experimental values (makespans, allocation rates, or computational times) in one data set is larger than the values in another set of data. The p values with $p < 0.05$ are associated with statistically significant test outcomes and the A values with $A > 0.71$ or $A < 0.29$ are related to large effect sizes. The results regarding the Experiment A (simulation sets I, II, III) and the hypotheses $H_{A1,1}$, $H_{A1,2}$ are given in Table 18, the results regarding the Experiment B (simulation sets I, IV) and $H_{B1,1}$, $H_{B1,2}$, $H_{B1,3}$ are given in Table 19, and the results regarding the Experiment C (simulation sets V, VI) and $H_{C1,1}$, $H_{C1,2}$ are given in Table 20. Results are evaluated in Appendix D.

9. Discussions and further research

The model is sensitive to the number of available GSE vehicles. In ground handling, the size GSE fleet is limited. Efficient allocation of ground handling tasks to GSE vehicles is important to deal with limited resources. We combine a decentralized task allocation mechanism, TeSSI auction, with centralized local optimization, the SPDP optimization.

We focused on the ground handling operations of flights, however, the impact of ground handling task delays on flight delays were not explored. We investigated the task allocation rates after replanning, but we did not analyze the flight delays caused by the interrupted tasks. For path planning, we modeled the agents as standard and large agents. However, we only used square shapes and kinematics of agents are not included in research.

We achieved high task allocation rates within the 4 hour-planning window in reasonable time. However, we limited the solution time of GUROBI to 5 s to avoid high CPU time. Thus, in some cases, the optimizer might reach to the solution time limit without finding a feasible allocation and rescheduling solution. In other cases, it is also possible that infeasibility is proven before

Table 17
Simulation sets I to VIII used in stability tests and experiments.

Simulation set	I	II	III	IV	V	VI	VII	VIII
scenario	normal	offpeak	busy	normal	normal	normal	normal	normal
objective	*makespan*	*makespan*	*makespan*	*makespan*	*makespan*	*makespan*	*sum − T*	*makespan*
bundling	no	no	no	yes	no	no	no	no
delayed	no	no	no	no	yes	yes	no	no
replanning	no	no	no	no	no	yes	no	no
environment	basic	basic	basic	basic	basic	basic	basic	extended

Table 18
p and *A* values, non-parameteric Wilcoxonsigned-rank tests for Experiment A.

	Offpeak-normal *p*	Offpeak-normal *A*	Normal-busy *p*	Normal-busy *A*
vehicle (s)	refueling vehicle (RE)			
makespan	3.88E−18	0.0000	3.85E−18	1.0000
allocation rate	9.05E−21	1.0000	3.95E−23	1.0000
CPU time	2.49E−09	0.2889	3.90E−18	0.0000
vehicle (s)	catering vehicle (CA)			
makespan	3.88E−18	0.0000	6.04E−01	0.6400
allocation rate	1.97E−09	0.6800	3.82E−08	0.6484
CPU time	3.90E−18	0.0000	3.90E−18	0.0000
vehicle (s)	baggage handling vehicle (BA)			
makespan	3.88E−18	0.0000	2.65E−01	0.7440
allocation rate	–	0.5000	1.31E−05	0.5950
CPU time	3.90E−18	0.0000	3.90E−18	0.0000
vehicle (s)	water service vehicle (WA)			
makespan	3.88E−18	0.0000	5.65E−07	0.1670
allocation rate	3.55E−20	0.9950	1.26E−19	0.9923
CPU time	3.90E−18	0.0000	3.90E−18	0.0000
vehicle (s)	lavatory service vehicle (WC)			
makespan	3.85E−18	0.0000	3.89E−18	1.0000
allocation rate	3.15E−22	1.0000	1.71E−19	1.0000
CPU time	3.90E−18	0.0000	3.90E−18	0.0000
vehicle (s)	overall results (all vehicles)			
makespan	3.90E−18	0.0000	5.27E−18	0.9852
allocation rate	1.41E−18	1.0000	3.25E−18	0.7143
CPU time	3.90E−18	0.0000	3.90E−18	0.0000

the time limit is reached. In both conditions, the GSE vehicle agent cannot bid on the candidate task and loses the chance to insert a new task into its existing schedule. If this happens for all other vehicle agents, none of the agents bids on the task and task is assigned into the set of *unallocated tasks*. The meaning of this in real life is that some of the ground handling tasks, *unallocated tasks*, cannot be finished within specified time windows. However, if they are critical tasks such as refueling, they have to be completed before aircraft takes off. This might cause flight delays at the airport. One improvement approach would be to re-run a centralized optimization or a heuristic method to check whether a feasible position exists for the *unallocated tasks* in alternative solutions. Such a feasible solution might have been lost due to the decentralized approach. On the other hand, if it is proven that no alternative feasible solution exists, it means that there is no way to allocate and complete that task within the desired time interval and operational delays are unavoidable. This can only be resolved by increasing the number of GSE vehicles dedicated to operate the task. The exploration of the impact of using a centralized optimization solver as an additional repair or feasibility check procedure remains as an item of future research.

10. Conclusion

We propose a multi-agent system for automation of aircraft ground handling operations. The system is able to generate task allocation optimization and path planning solutions for multiple agents. For task allocation we use an optimization based TESSI auction method since there are multiple stakeholders in the system. Unlike the existing auction methods, we generate the bids by modeling and solving the task scheduling for agents as single-vehicle pickup and delivery optimization problems. We also propose and test bundling procedures for saving time and replanning functions to deal with disruptions in real time. For path planning, we use Prioritized SIPP since vehicle priorities are commonly used in airside environments. We also use LA-SIPP to represent non-unit sizes of GSE vehicles.

The experiments for task allocation include different traffic demand scenarios, task bundling, replanning, and different objectives. We reported the allocation rates within 0.00–1.00 scale. The task allocation rate was close to 1.00 for many of the tested instances, higher than 0.875 in most of the cases, and 0.81 in the worst case.

For path planning, the mean values of the performance indicators were reported both for the 16×16 and 64×64 grid-maps. The outputs were *mean path length* and *mean shortest path length* in meters, *mean path duration* and *mean shortest path duration*, *mean of average delays for tasks*, *mean of average delays for agents* in seconds, *mean agent on-time rate*, *mean task on-time rate*, *agent success rate* within a scale of 0.00–1.00, and *mean computational time* in seconds.

The *path length* and *path duration* show the length and duration of the obtained paths when collisions were handled with path planning. In the solutions, the agent either waits for the obstacle to move or travels around the obstacle using a longer path. The baseline solutions, *shortest path length* and *shortest path duration* are the shortest paths without considering any obstacles. In the results, the path lengths were 1.0012 and 1.0014 times

Table 19
p and *A* values, non-parameteric Wilcoxonsigned-rank test for Experiment B.

	No bundle - bundle *p*	No bundle - bundle *A*
vehicle (s)	refueling vehicle (RE)	
makespan	3.89E−18	0.0072
allocation rate	2.86E−1	0.9750
CPU time	3.90E−18	0.0000
vehicle (s)	catering vehicle (CA)	
makespan	3.88E−18	0.0000
allocation rate	1.97E−09	0.3200
CPU time	3.90E−18	0.0017
vehicle (s)	baggage handling vehicle (BA)	
makespan	7.30E−18	0.0060
allocation rate	2.44E−02	0.5300
CPU time	3.90E−18	0.0000
vehicle (s)	water service vehicle (WA)	
makespan	9.04E−16	0.1779
allocation rate	9.91E−1	0.0318
CPU time	4.02E−18	0.0049
vehicle (s)	lavatory service vehicle (WC)	
makespan	3.88E−18	0.0000
allocation rate	1.73E−19	0.9906
CPU time	3.90E−18	0.0000
vehicle (s)	overall results (all vehicles)	
makespan	3.90E−18	0.0019
allocation rate	1.47E−10	0.7773
CPU time	3.90E−18	0.0000

Table 20
p and *A* values, non-parameteric Wilcoxonsigned-rank test for Experiment C.

	Replan - no replan *p*	Replan - no replan *A*
vehicle (s)	refueling vehicle (RE)	
makespan	2.01E−15	0.8186
allocation rate	2.78E−13	0.8650
vehicle (s)	catering vehicle (CA)	
makespan	1.08E−01	0.4030
allocation rate	2.20E−08	0.2720
vehicle (s)	baggage handling vehicle (BA)	
makespan	8.63E−18	0.9902
allocation rate	6.33E−05	0.4200
vehicle (s)	water service vehicle (WA)	
makespan	1.12E−07	0.7836
allocation rate	2.73E−19	0.9888
vehicle (s)	lavatory service vehicle (WC)	
makespan	4.12E−18	0.9887
allocation rate	1.32E−01	0.5247
vehicle (s)	overall results (all vehicles)	
makespan	1.37E−17	0.9605
allocation rate	8.53E−17	0.9344

the shortest path lengths on 16 × 16 and 64 × 64 grid-maps, respectively, which indicated that the path planning algorithms could plan paths without many collisions and produced high quality solutions.

The *path duration* was usually longer than the required traveling time to traverse the path. This means that the agents tend to stop and wait to avoid collisions instead of detours. The *path durations* were 1.28 and 1.39 times the *shortest durations* in *basic* and *extended* models, due to the tendency to wait. The means of average delays per agent were around 20 and 40 s, while the means of average delays per task were around 15 and 30 s, for 16 × 16 and 64 × 64 grid-maps. Similarly, the means of ratios of tasks with no delays (0.96 and 0.95) were higher than the means of ratios of agents with no delays (0.94 and 0.93). However, all on-time rates were significantly high and average delays were reasonable for tasks and agents. The means of the success rates that agents can successfully find paths on 16 × 16 and 64 × 64 grid-maps were also high (1.0 and 0.98). The mean of CPU times for all experiments using 16 × 16 grid-map was 5.11 s, and the mean of CPU times for all experiments using 64 × 64 grid-map was 86.83 s. We generated the heatmaps of the paths obtained in experiments, to see the frequencies of cell occupations on grid-maps over different simulations. The frequently used paths or cells were visible in these maps.

In general, the results show that the CPU times are sufficient to dynamically generate solutions in real time for the ground handling operations at one pier during a specified time window of the day. Allocation rates are high given the solution time limit for SPDP optimizer, and the collision-free paths do not highly deviate from the shortest paths. Thus, using a decoupled approach, our model can produce high quality solutions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Appendix A. Benchmarks

The commonly used environments in solving MAPF problems are: a. Dragon Age Origins (DAO) for which the maps are publicly available. b. Open N ×N grid-maps where the values of N are usually 8, 16, 32. These maps are suitable for experiments in which the ratios of agents to space are high. c. N ×N grid-maps with random obstacles. d. Warehouse grid-maps that resemble real-world autonomous warehouses. The common Dragon Age Origins (DAO) maps are *den*520*d* and *brc*202*d* maps. The *den*520*d* map has large open spaces. In contrast, the *brc*202*d* map has fewer open spaces and has more bottlenecks. The configuration of *brc*202*d* map is similar to airside environments with connected taxiways and runways. The configuration of an aircraft stand is dissimilar to *den*520*d* and *brc*202*d* maps. However, performances of MAPF solvers on those maps are still valuable guidelines for applications in ground handling. The success rates of existing path planning algorithms on standard grid-maps were presented by [7,65]. [65] provided the success rates for several optimal path planning solvers tested on DAO *brc*202*d* map environment. The success rates for optimal solvers were ranging from 0.80 to 1.00 for all solvers, up to 80 agents. [7] presented the success rates of sub-optimal path planning solvers on 20 × 20 grid-map with 10% obstacles, and on DAO *brc*202*d* map. The success rates of sub-optimal solvers on 20 × 20 grid-map with 10% obstacles were within the range of 0.8 to 1 approximately up to 40 agents. The success rates of sub-optimal solvers on DAO *brc*202*d* map were within the range of 0.8 to 1 approximately up to 100 agents. The success rates of existing optimal [65] and near-optimal [7] path planning solvers have significantly high success rates, approximately 1.00, when performed on standard environments for less than 20 agents.

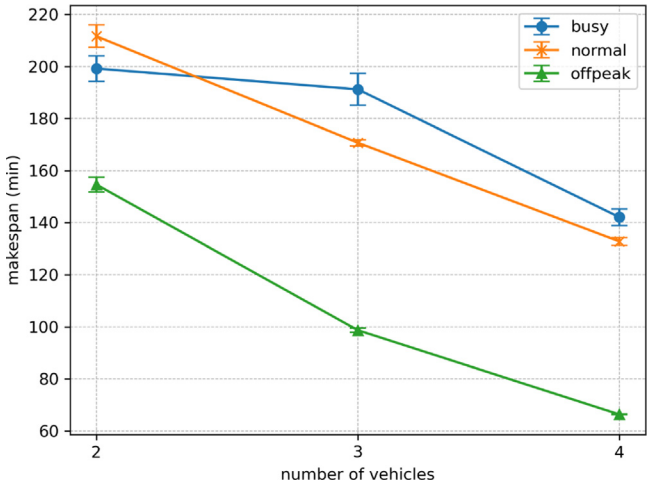


Fig. B.15. *makespan.*

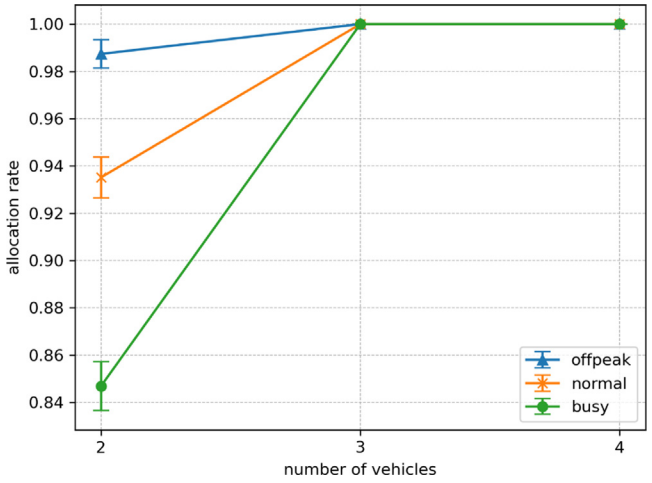


Fig. B.16. *allocation rate.*

Appendix B. Sensitivity analysis on number of available vehicles

Fig. B.15 shows the change observed in makespan for different traffic demand scenarios, by changing the number of available vehicles. When the number of vehicles increases, the makespan decreases, since a higher number of vehicles can share the same set of ground handling tasks. Also, the busier the traffic demand scenario is, the higher the makespan. The only exception occurs when the number of vehicles is 2. This is due to the low allocation rates when the number of vehicles is limited.

Fig. B.16 shows the change observed in allocation rates for different traffic demand scenarios, by changing the number of available vehicles. The largest number of ground handling tasks exist in the busy scenario due to the largest number of flights. The restricted resources do not allow all ground handling tasks to be successfully allocated, resulting in a lower allocation rate. In Fig. B.16, when the available number of vehicles is 2, allocation rates are less than 1 for all scenarios, which means that not all of the ground handling tasks could be allocated. The scenarios with fewer flights have higher allocations rates. When the number of available vehicles were increased to 3, all of the ground handling tasks were allocated in all scenarios.

Fig. B.17 shows the change in CPU times by increasing the number of available vehicles in all scenarios. CPU times were

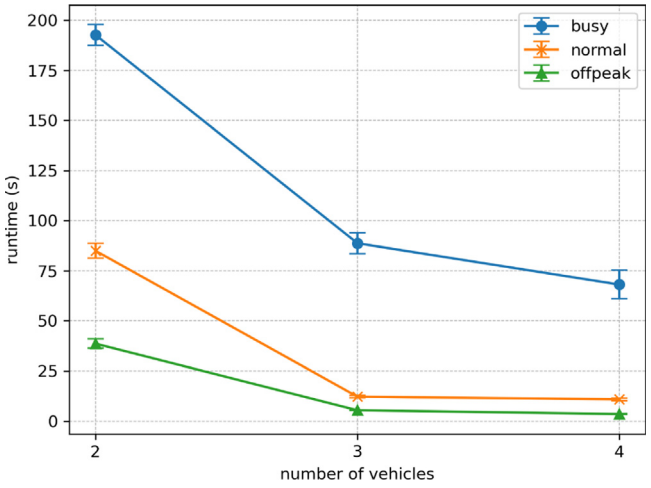


Fig. B.17. *computational time.*

the highest in the busy scenario. The CPU time decreases when the number of available vehicles increases in all scenarios. More available vehicles result in fewer tasks in one vehicle's schedule, and fewer tasks reduces the CPU time to solve SPDP optimization model for an agent to generate bids. This result is especially evident in the busy scenario.

Appendix C. Simulation set I, *coefficient of variation*

Fig. C.18, Fig. C.19, Fig. C.20 show the *coefficient of variation* for *makespan*, *allocation rate*, *CPU time* for simulation set I.

Appendix D. Evaluation of statistical significance

Table 18 shows the statistical significance and effect size values for the results related to Experiment A (Simulation sets I, II, III). Table 18 presents the non-parameteric Wilcoxonsigned-rank test results for each GSE vehicle separately, and for the complete solution. We present an evaluation of the results for the overall solution. Our hypotheses were, $H_{A1,1}$: "The *makespan* value is higher in normal scenario compared to offpeak scenario", and $H_{A1,2}$: "The *makespan* value is higher in busy scenario compared to normal scenario". The p-value for offpeak-normal scenario comparison supported the $H_{A1,1}$. Compared to offpeak scenario, the normal scenario includes a higher number of flights and operations and the makespan was higher in the normal scenario as expected. However, despite more frequent flights of busy scenario, $H_{A1,2}$ was rejected. Some of the tasks were not able to be allocated in busy scenario, due to time windows constraints. The test results regarding Experiment B (simulation sets I, IV) and $H_{B1,1}$, $H_{B1,2}$, $H_{B1,3}$ are given in Table 19. Our hypotheses were, $H_{B1,1}$: "The maximum vehicle makespan for the model with task bundles is higher than that of the model without task bundles"., $H_{B1,2}$: " The task allocation rates for the model with task bundles are lower than that of the model without task bundles"., and $H_{B1,3}$: "The CPU time for the model with task bundles is lower than that of the model without task bundles". The p-values for no bundle - bundle comparisons showed significant differences for all ground handling tasks. The Vargha–Delaney A-values showed that the incremental increases in makespans caused by bundling were significant. Therefore, $H_{B1,1}$ was supported. $H_{B1,2}$ was tested by comparing the allocation rates. The p-values showed significant differences between allocation rates. For RE and WC, the task allocation rates with bundles were smaller. Thus, $H_{B1,2}$ was supported for RE and WC. For CA and BA, the medians of task

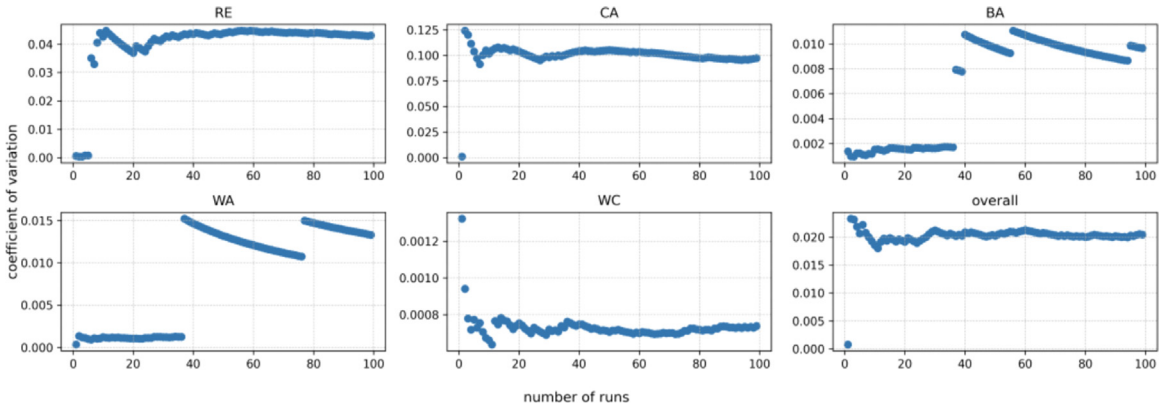


Fig. C.18. Simulation set I, *coefficient of variation* for *makespan*.

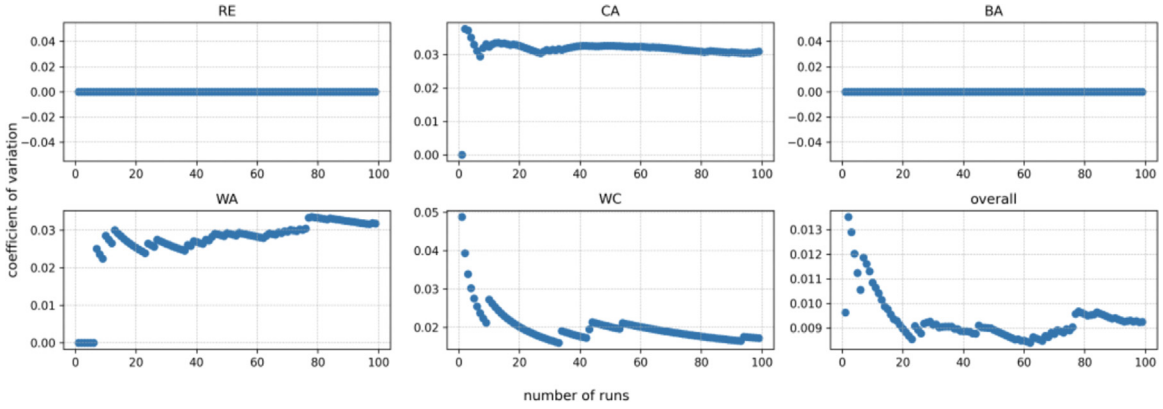


Fig. C.19. Simulation set I, *coefficient of variation* for *allocation rate*.

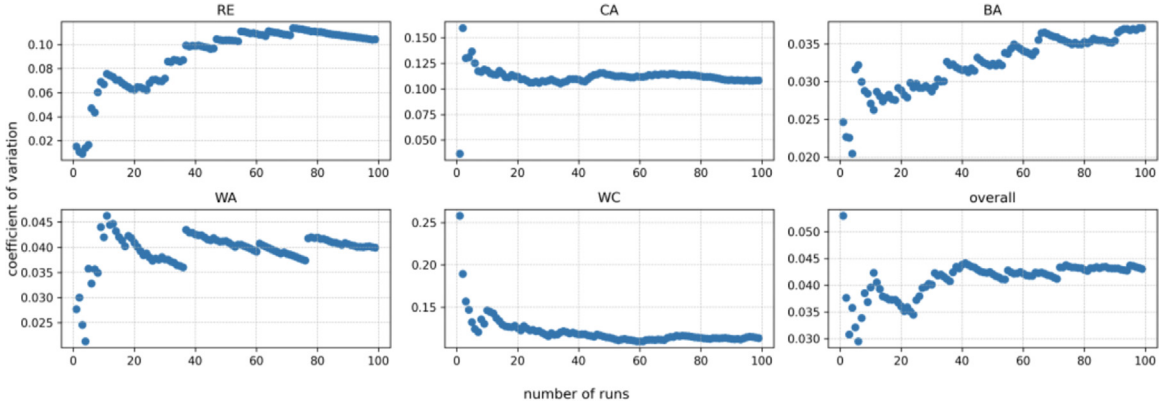


Fig. C.20. Simulation set I, *coefficient of variation* for *CPU time*.

allocation rates were equal to 1 and the Vargha–Delaney *A*-values showed that the effect sizes were small. There is no significant difference and this may be due to the short CPU time of allocation for CA and BA. For WA, allocation rate was better with bundling and $H_{B1,2}$ was rejected. We tested the $H_{B1,3}$ by comparing the CPU times. Based on the *p*-values, there were significant differences in the CPU times and the Vargha–Delaney A-values revealed large effect sizes. $H_{B1,3}$ was rejected. The optimization solver could be failing to find feasible allocation options for bundled tasks. Thus, more complex bundling procedures should be developed to identify high quality bundles. For Experiment C, we tested the significance and effect size for replan–no replan scenario pair (simulation sets VI, V). Table 20 shows the related *p* and *A* values for each GSE vehicle agent and for the overall results. Hypotheses were, H_{C1}: "The makespan of the simulation set with an activated replanning function is lower than the one without an activated replanning function", and H_{C2}: "The task allocation rate of the simulation set with an activated replanning function is higher than the one without an activated replanning function". There was no significant difference replan-no replan scenario pairs. Only for the CA, the mean of makespans were lower in the replanning scenario and except for the CA, the task allocation

rates were higher in the scenario with replanning for each GSE vehicle. Thus, H_{C1} and H_{C2} were not verified for all vehicle types.

References

[1] Royal schiphol group building a roadmap towards a fully autonomous airside operation in 2050, 2021, https://www.futuretravelexperience.com/2021/03/royal-schiphol-group-building-a-roadmap-towards-a-fully-autonomous-airside-operation-in-2050/ (Last Accessed 20 December 2021).

[2] Schiphol, An autonomous airport in 2050, 2021, https://www.schiphol.nl/en/innovation/blog/an-autonomous-airport-in-2050/ (Last Accessed 20 December 2021).

[3] Schiphol, Sustaining your world vision and strategy towards the most sustainable airports, Technical Report, Schiphol, 2020.

[4] E. Nunes, M. Gini, Multi-robot auctions for allocation of tasks with temporal constraints, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 29, no. 1, 2015, http://dx.doi.org/10.1609/aaai.v29i1.9440.

[5] D. Silver, Cooperative pathfinding, in: Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, Vol. 1, no. 1, 2021, pp. 117–122, http://dx.doi.org/10.1609/aiide.v1i1.18726.

[6] Z. Bnaya, A. Felner, Conflict-oriented windowed hierarchical cooperative a, in: 2014 IEEE International Conference on Robotics and Automation, ICRA, 2014, pp. 3743–3748.

[7] H. Ma, D. Harabor, P.J. Stuckey, J. Li, S. Koenig, Searching with consistent prioritization for multi-agent path finding, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, no. 01, 2019, pp. 7643–7650, http://dx.doi.org/10.1609/aaai.v33i01.33017643.

[8] M. Phillips, M. Likhachev, Sipp: Safe interval path planning for dynamic environments, in: 2011 IEEE International Conference on Robotics and Automation, IEEE, 2011, pp. 5628–5635.

[9] Schiphol, An autonomous airport in 2050, 2021, URL https://www.schiphol.nl/en/innovation/blog/an-autonomous-airport-in-2050/ (Last Accessed 20 December 2021).

[10] D.A. Tabares, F. Mora-Camino, Aircraft ground handling: analysis for automation, in: 17th AIAA Aviation Technology, Integration, and Operations Conference, 2017, p. 3425.

[11] M. Schmidt, P. Nguyen, M. Hornung, Novel aircraft ground operation concepts based on clustering of interfaces, Technical Report, SAE Technical Paper, 2015.

[12] Boeing Commerical Airplanes, 737: Airplane Characteristics for Airport Planning, Sept. 2013, Technical Report, D6-58325-6, 2013.

[13] J. Soomers, Agent-based delay management in autonomous engine-off taxi systems, 2022, http://resolver.tudelft.nl/uuid:52f18928-1c91-4244-aa65-2effd441c0b5.

[14] K. Fines, Agent-based distributed planning and coordination for resilient airport surface movement operations, 2019, http://resolver.tudelft.nl/uuid:2a9a2e14-f7c4-47f1-a12c-81ba6cd53397.

[15] V. Frias-Martinez, E. Sklar, S. Parsons, Exploring auction mechanisms for role assignment in teams of autonomous robots, in: D. Nardi, M. Riedmiller, C. Sammut, J. Santos-Victor (Eds.), RoboCup 2004: Robot Soccer World Cup VIII, in: Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 532–539, http://dx.doi.org/10.1007/978-3-540-32256-6_49.

[16] S. Koenig, P. Keskinocak, C. Tovey, Progress on agent coordination with cooperative auction, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 24, no. 1, 2010, pp. 1713–1717, http://dx.doi.org/10.1609/aaai.v24i1.7764.

[17] R. Nair, T. Ito, M. Tambe, S. Marsella, Task allocation in the RoboCup rescue simulation domain: A short note, in: A. Birk, S. Coradeschi, S. Tadokoro (Eds.), RoboCup 2001: Robot Soccer World Cup V, in: Lecture Notes in Computer Science, vol.2377, Springer, 2001, pp. 751–754, http://dx.doi.org/10.1007/3-540-45603-1_129.

[18] P. Rizzo, Auction-based task allocation for online pickup and delivery problems with time constraints and uncertainty, Delft University of Technology, 2021, http://resolver.tudelft.nl/uuid:3d1f41c7-da33-42e7-a76f-414d93907eac.

[19] R.G. Smith, The contract net protocol: High-level communication and control in a distributed problem solver, IEEE Trans. Comput. C-29 (12) (1980) 1104–1113, http://dx.doi.org/10.1109/TC.1980.1675516.

[20] W. Hönig, S. Kiesel, A. Tinka, J. Durham, N. Ayanian, Conflict-based search with optimal task assignment, in: Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, 2018.

[21] H.W. Kuhn, The hungarian method for the assignment problem, Nav. Res. Logist. Q. 2 (1–2) (1955) 83–97.

[22] D.E. Goldberg, J.H. Holland, Genetic algorithms and machine learning, Mach. Learn. 3 (2) (1988) 95–99, http://dx.doi.org/10.1023/A:1022602019183.

[23] J.H. Holland, Genetic algorithms, Sci. Am. 267 (1992) 66–73, URL http://www.jstor.org/stable/24939139 (Accessed 28 Oct. 2022).

[24] M. Dorigo, G. Di Caro, Ant colony optimization: A new meta-heuristic, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Vol. 2, IEEE, 1999, pp. 1470–1477.

[25] M. Dorigo, E. Bonabeau, G. Theraulaz, Ant algorithms and stigmergy, Future Gener. Comput. Syst. 16 (8) (2000) 851–871.

[26] D. Teodorovic, P. Lucic, G. Markovic, M. Dell'Orco, Bee colony optimization: Principles and applications, in: 2006 8th Seminar on Neural Network Applications in Electrical Engineering, IEEE, 2006, pp. 151–156.

[27] J. García, A. Berlanga, J.M. Molina, J.R. Casar, Optimization of airport ground operations integrating genetic and dynamic flow management algorithms, AI Commun. 18 (2) (2005) 143–164.

[28] W. Guo, P. Xu, Z. Zhao, L. Wang, L. Zhu, Q. Wu, Scheduling for airport baggage transport vehicles based on diversity enhancement genetic algorithm, Nat. Comput. 19 (4) (2020) 663–672.

[29] J.-B. Gotteland, N. Durand, Genetic algorithms applied to airport ground traffic optimization, in: The 2003 Congress on Evolutionary Computation, Vol. 1, 2003. CEC'03, IEEE, 2003, pp. 544–551.

[30] G.-H. Tzeng, Y.-W. Chen, The optimal location of airport fire stations: A fuzzy multi-objective programming and revised genetic algorithm approach, Transp. Plan. Technol. 23 (1) (1999) 37–55.

[31] H. Zhao, L. Cheng, Ant colony algorithm and simulation for robust airport gate assignment, Math. Probl. Eng. 2014 (2014).

[32] W. Deng, J. Xu, H. Zhao, An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem, IEEE Access 7 (2019) 20281–20292.

[33] S. Koenig, C.A. Tovey, X. Zheng, I. Sungur, Sequential bundle-bid single-sale auction algorithms for decentralized control, in: IJCAI, 2007, pp. 1359–1365.

[34] B. Heap, M. Pagnucco, Repeated sequential single-cluster auctions with dynamic tasks for multi-robot task allocation with pickup and delivery, in: German Conference on Multiagent System Technologies, Springer, 2013, pp. 87–100.

[35] L. Brunet, H.-L. Choi, J. How, Consensus-based auction approaches for decentralized task assignment, in: AIAA Guidance, Navigation and Control Conference and Exhibit, 2008, p. 6839, http://dx.doi.org/10.2514/6.2008-6839.

[36] P.R. Wurman, R. D'Andrea, M. Mountz, Coordinating hundreds of cooperative, autonomous vehicles in warehouses, AI Mag. 29 (1) (2008) 9, http://dx.doi.org/10.1609/aimag.v29i1.2082, URL https://ojs.aaai.org/index.php/aimagazine/article/view/2082.

[37] K. Dresner, P. Stone, A multiagent approach to autonomous intersection management, J. Artif. Int. Res. 31 (1) (2008) 591–656, http://dx.doi.org/10.5555/1622655.1622672.

[38] J. Yu, S.M. LaValle, Planning optimal paths for multiple robots on graphs, in: 2013 IEEE International Conference on Robotics and Automation, 2013, pp. 3612–3617, http://dx.doi.org/10.1109/ICRA.2013.6631084.

[39] R. Stern, N.R. Sturtevant, A. Felner, S. Koenig, H. Ma, T.T. Walker, J. Li, D. Atzmon, L. Cohen, T.S. Kumar, et al., Multi-agent pathfinding: Definitions, variants, and benchmarks, in: Twelfth Annual Symposium on Combinatorial Search, 2019.

[40] A. Felner, R. Stern, S.E. Shimony, E. Boyarski, M. Goldenberg, G. Sharon, N. Sturtevant, G. Wagner, P. Surynek, Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges, in: Tenth Annual Symposium on Combinatorial Search, 2017.

[41] T. Standley, Finding optimal solutions to cooperative pathfinding problems, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 24, no. 1, 2010, pp. 173–178, http://dx.doi.org/10.1609/aaai.v24i1.7564.

[42] A. Felner, M. Goldenberg, G. Sharon, R. Stern, T. Beja, N. Sturtevant, J. Schaeffer, R. Holte, Partial-expansion A* with selective node generation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 26, no. 1, 2021, pp. 471–477, http://dx.doi.org/10.1609/aaai.v26i1.8137.

[43] G. Sharon, R. Stern, M. Goldenberg, A. Felner, The increasing cost tree search for optimal multi-agent pathfinding, Artificial Intelligence 195 (2013) 470–495, http://dx.doi.org/10.1016/j.artint.2012.11.006.

[44] G. Sharon, R. Stern, A. Felner, N.R. Sturtevant, Conflict-based search for optimal multi-agent pathfinding, Artificial Intelligence 219 (2015) 40–66, http://dx.doi.org/10.1016/j.artint.2014.11.006.

[45] P. Surynek, Towards optimal cooperative path planning in hard setups through satisfiability solving, in: Pacific Rim International Conference on Artificial Intelligence, Springer, 2012, pp. 564–576, http://dx.doi.org/10.1007/978-3-642-32695-0_50.

[46] M. Ryan, Constraint-based multi-robot path planning, in: 2010 IEEE International Conference on Robotics and Automation, IEEE, 2010, pp. 922–928, http://dx.doi.org/10.1109/ROBOT.2010.5509582.

[47] M.R.K. Ryan, Exploiting subgraph structure in multi-robot path planning, J. Artificial Intelligence Res. 31 (2008) 497–542, http://dx.doi.org/10.5555/1622655.1622670.

[48] E. Erdem, D.G. Kisa, U. Oztok, P. Schüller, A general formal framework for pathfinding problems with multiple agents, in: Twenty-Seventh AAAI Conference on Artificial Intelligence, 2013, http://dx.doi.org/10.5555/2891460.2891501.

[49] A. Andreychuk, K. Yakovlev, P. Surynek, D. Atzmon, R. Stern, Multi-agent pathfinding with continuous time, Artificial Intelligence 305 (2022) 103662, http://dx.doi.org/10.1016/j.artint.2022.103662.
[50] J. Li, P. Surynek, A. Felner, H. Ma, T.S. Kumar, S. Koenig, Multi-agent path finding for large agents, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, no. 01, 2019, pp. 7627–7634.
[51] L. Cohen, S. Koenig, Bounded suboptimal multi-agent path finding using highways, in: IJCAI, 2016, pp. 3978–3979.
[52] H. Ma, W. Hönig, T.S. Kumar, N. Ayanian, S. Koenig, Lifelong path planning with kinematic constraints for multi-agent pickup and delivery, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, no. 01, 2019, pp. 7651–7658.
[53] L. Cohen, T. Uras, T.S. Kumar, S. Koenig, Optimal and bounded-suboptimal multi-agent motion planning, in: Twelfth Annual Symposium on Combinatorial Search, 2019.
[54] S. Zhu, H. Sun, X. Guo, Cooperative scheduling optimization for ground-handling vehicles by considering flights' uncertainty, Comput. Ind. Eng. 169 (2022) 108092, http://dx.doi.org/10.1016/j.cie.2022.108092.
[55] W. Guo, P. Xu, Z. Zhao, L. Wang, L. Zhu, Q. Wu, Scheduling for airport baggage transport vehicles based on diversity enhancement genetic algorithm, Nat. Comput. 19 (4) (2020) 663–672, http://dx.doi.org/10.1007/s11047-018-9703-0.
[56] P. García Ansola, A. García Higuera, F.J. Otamendi, J. de las Morenas, Agent-based distributed control for improving complex resource scheduling: Application to airport ground handling operations, IEEE Syst. J. 8 (4) (2014) 1145–1157, http://dx.doi.org/10.1109/JSYST.2013.2272248.
[57] S. Padrón, D. Guimarans, J.J. Ramos, S. Fitouri-Trabelsi, A bi-objective approach for scheduling ground-handling vehicles in airports, Comput. Oper. Res. 71 (2016) 34–53, http://dx.doi.org/10.1016/j.cor.2015.12.010.
[58] J. Evler, E. Asadi, H. Preis, H. Fricke, Airline ground operations: Schedule recovery optimization approach with constrained resources, Transp. Res. C 128 (2021) 103129, http://dx.doi.org/10.1016/j.trc.2021.103129.
[59] A. Vidosavljevic, V. Tosic, Modeling of turnaround process using Petri nets, in: Air Transport Research Society (ATRS) World Conference, Air Transport Research Society, Porto, Portugal, 2010, URL https://hal.science/hal-01821674.
[60] D. Alonso Tabares, F. Mora-Camino, A. Drouin, A multi-time scale management structure for airport ground handling automation, J. Air Transp. Manag. 90 (2021) 101959, http://dx.doi.org/10.1016/j.jairtraman.2020.101959.
[61] D. Guimarans, S. Padrón, A stochastic approach for planning airport ground support resources, Int. Trans. Oper. Res. 29 (6) (2022) 3316–3345, http://dx.doi.org/10.1111/itor.13104.
[62] S.N. Parragh, K.F. Doerner, R.F. Hartl, A survey on pickup and delivery problems. Part II: Transportation between pickup and delivery locations, J. Für Betriebswirtschaft 58 (1) (2008) 21–51.
[63] M.G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A.J. Kleywegt, S. Koenig, C.A. Tovey, A. Meyerson, S. Jain, Auction-based multi-robot routing, in: Robotics: Science and Systems, Vol. 5, Rome, Italy, 2005, pp. 343–350.
[64] M. Gansterer, R.F. Hartl, Centralized bundle generation in auction-based collaborative transportation, Or Spectrum 40 (3) (2018) 613–635.
[65] E. Boyarski, A. Felner, R. Stern, G. Sharon, D. Tolpin, O. Betzalel, E. Shimony, ICBS: Improved conflict-based search algorithm for multi-agent pathfinding, in: IJCAI 2015 - Proceedings of the 24th International Joint Conference on Artificial Intelligence, 2015, pp. 740–746.
[66] J.-S. Lee, T. Filatova, A. Ligmann-Zielinska, B. Hassani-Mahmooei, F. Stonedahl, I. Lorscheid, A. Voinov, G. Polhill, Z. Sun, D.C. Parker, The complexities of agent-based modeling output analysis, J. Artif. Soc. Soc. Simul. 18 (4) (2015) 4, http://dx.doi.org/10.18564/jasss.2897.

Szu-Tung Chen received her M.Sc. degree from Faculty of Aerospace Engineering at Delft University of Technology in 2022 and her B.Sc. degree from Department of Mechanical Engineering at National Taiwan University in 2019. She is currently a baggage handling specialist at NACO (Netherlands Airport Consultants). Her research interests include air transport operation, air traffic management, agent-based modeling, and optimization modeling for task allocation and path planning.

Gulcin Ermis is a post-doctoral researcher at Air Transport and Operations group at Faculty of Aerospace Engineering, Delft University of Technology. She received her Ph.D. degree in Industrial Engineering and Operations Management from Koc University in 2014 and M.Sc. degree in Industrial Engineering from Istanbul Technical University in 2009. After receiving her Ph.D. degree, she worked as a post-doctoral researcher and software developer in academic and research institutions including the Royal Netherlands Academy of Arts and Sciences, NEC Laboratories Europe GmbH, Sabanci University. Her research interests include single machine scheduling, discrete optimization, constraint programming, vehicle routing, multimodal distribution networks, algorithms and complexity, scheduling theory and applications.

Alexei Sharpanskykh is an assistant professor at Air Transport and Operations group at Faculty of Aerospace Engineering, Delft University of Technology. He received his Ph.D. degree in the area of artificial intelligence from VU University Amsterdam in 2008. After that, he worked as a postdoctoral researcher in the areas of Complex Adaptive Systems and artificial intelligence at the same university. His research interests include: mathematical and computational modeling and analysis of safety, security, and resilience of complex sociotechnical systems in aviation; application of multiagent systems and artificial intelligence techniques in aviation; development of tools and techniques for simulation and optimization of sociotechnical systems.