



HAL
open science

Transitioning from AGVs to AIVs in Integrated Job Shop Scheduling with Transportation Tasks: a Multi-agent Simulator for Comparative Analysis

Kader Sanogo, Abdelkader Mekhalef Benhafssa, M'hammed Sahnoun

► **To cite this version:**

Kader Sanogo, Abdelkader Mekhalef Benhafssa, M'hammed Sahnoun. Transitioning from AGVs to AIVs in Integrated Job Shop Scheduling with Transportation Tasks: a Multi-agent Simulator for Comparative Analysis. SIMULATION: Transactions of The Society for Modeling and Simulation International, In press, 10.1177/ToBeAssigned . hal-04741046

HAL Id: hal-04741046

<https://hal.science/hal-04741046v1>

Submitted on 17 Oct 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Transitioning from AGVs to AIVs in Integrated Job Shop Scheduling with Transportation Tasks: a Multi-agent Simulator for Comparative Analysis

Journal Title
XX(X):1–26
©The Author(s) 2024
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/

SAGE

Kader SANOGO^{1,2}, Abdelkader MEKHALEF BENHAFSSA¹ and M'hammed SAHNOUN³

Abstract

Optimizing job shop scheduling in modern factories demands flexibility and adaptability to handle unexpected events and Unmanned Ground Vehicles (UGVs) limitations. This paper addresses these challenges by introducing a novel multi-agent simulator for the Job Shop Scheduling Problem (JSSP) with UGVs handling transportation tasks. The simulator, designed with Netlogo, incorporates real-world constraints, such as collision avoidance, UGV fleet size, and battery limitations, often overlooked in prior studies. By comparing the two categories of UGVs, namely Autonomous Guided Vehicles (AGVs) and Autonomous Intelligent Vehicles (AIVs), under different scheduling methods (static vs. dynamic), we evaluate their performance in constrained manufacturing environments. Our findings highlight the superior performance of AIVs in terms of overall makespan and resilience to additional constraints. Among other results, we found that schedules with a fleet of 2 AIVs produced similar makespans than schedules with a fleet of 4 AGVs. Consequently, the simulator and the conducted study provide valuable insights for optimizing JSSP within constrained environments and making informed decisions regarding AIVs adoption.

Keywords

Simulation, AIV, AGV, Job-shop scheduling, FMS, Multi-agent System, Collision avoidance, Battery limitation, Fleet size

1 Introduction

In today's competitive manufacturing landscape, companies constantly seek to streamline operations and eliminate waste. This relentless pursuit of efficiency compels them to discover innovative methods and tools to enhance production and maintain high customer satisfaction. However, managing manufacturing processes can be complex, with numerous interconnected elements and unforeseen events. Therefore, companies need to rigorously test their ideas before implementing significant changes. In this context, simulation emerges as a powerful tool for examining and analyzing situations that are either too intricate for mathematical modeling or too costly to test in the real world^{1,2}.

A persistent challenge for factories is efficiently scheduling job shop tasks. In academic circles, this is referred to as the Job Shop Scheduling Problem (JSSP)³. JSSP focuses on determining the optimal sequence for processing products (jobs) through machines, including the transportation between each processing stage⁴. Historically, these transportation tasks were performed by human operators. But today, with the evolution of technology, they are now carried

out by robots, namely UGVs. In this paper, we use the term "UGV" to refer to the mobile robots that carry out transportation tasks. UGVs have been used in industry since the 1950s and have steadily improved over time⁵. They are divided into two main types: AGVs and AIVs (also known as Autonomous Mobile Robots - AMRs). AGVs have proven their effectiveness in repetitive material handling tasks for several decades now⁶. However, they require dedicated movement areas⁷, which could be expensive to set up, and can toughly handle surprises like people or unexpected objects in their way⁸, potentially causing operational downtime. AIVs, on the other hand, can navigate around obstacles

¹CESI LINEACT, EA 7527, Angouleme Campus, La Couronne, 16400, France

²ENSAM, Paris, 75013, France

³CESI LINEACT, EA 7527, Rouen Campus, S. E du Rouvray, Rouen, 76800, France

Corresponding author:

Kader SANOGO, Campus CESI Angoulême - 4 Rue Camille Claudel, 16400 La Couronne, France

Email: ksanogo@cesi.fr

and people without needing special tracks, making them better suited for modern factories in Industry 5.0 context, where people and robots work seamlessly together^{5,9}. They are also more sophisticated, adaptable, and safer than older methods^{5,7}.

JSSP with UGVs performing transportation tasks has been extensively studied by researchers¹⁰. The proposed scheduling techniques can be split into two categories: static scheduling approaches and dynamic scheduling approaches. While static scheduling approaches in job shop scheduling optimization excel at upfront, comprehensive planning (from start to finish), their inflexibility can lead to disruptions when unexpected events, such as machine breakdowns, occur within the workshop. In this regard, the authors in¹¹ show that optimal advanced schedules, which are schedules where task assignments and their sequence of execution are known in advance, could become infeasible when additional constraints, such as collision avoidance between transporters, are taken into account, leading to deadlocks. Dynamic scheduling approaches, on the other side, can deal with real-world job shop disruptions and react by adjusting schedules in real-time, minimizing their impact on overall production efficiency^{12,13}. Furthermore, the limited battery capacity of UGVs adds another constraint. Their reliance on batteries necessitates recharging during operation to maintain activity. This issue is gaining attention^{14,15} but despite research acknowledging the influence of UGVs' battery management on the overall performance of the manufacturing system¹⁶, most prior studies overlooked this important factor¹⁷ in JSSP resolution. Hence, considering battery limitations is essential for building a more accurate representation of the production system. Simulation can be a useful tool to achieve this by incorporating battery constraints into scheduling models.

Therefore, this paper introduces a simulator, based on a multi-agent system, for addressing the JSSP with UGVs handling transportation tasks. The simulator can handle both advanced and dynamic schedules while ensuring UGVs avoid collisions and operate within battery limitations. JSSP traditionally relies on AGVs for transportation tasks. However, introducing AIVs as an alternative transporter within JSSP holds promise for enhancing overall system performance compared to traditional AGVs. This paper particularly focuses on the influence of UGV fleet size, collision avoidance constraints, and battery limitations under both static and dynamic scheduling techniques. This comparative analysis aims to evaluate the performance of AGVs and AIVs within the context of JSSP. To summarize, the paper provides the following contributions:

- Development of a Simulator.

- Utilization of AIVs as transporters in JSSP
- Evaluation of the additional constraints' impact on the job shop scheduling
- Comparison of Scheduling Methods
- Investigation on the effect of UGV Fleet Size with additional constraints

The rest of the paper is organized as follows. In Section 2, a literature review related to simulation techniques and their utilization to address FMS issues is presented. Section 3 is dedicated to the problem description, where the integration of UGVs' battery limitation into the JSSP is explained. Section 4 provides a comprehensive analysis of the simulator's functionalities, delving into its underlying framework and core components. Section 5 presents the experiments conducted during this work, and the results obtained are discussed in Section 6. Finally, Section 7 contains the paper's conclusion and the perspectives.

2 Related work

Simulation is a powerful tool that served as a valuable asset for researchers for decades, allowing them to observe and analyze the behavior of complex phenomena. It is a technique that involves building a model of a real-world system that can then be run to observe and analyze the behavior of the system, all without impacting the actual system itself^{1,2}. The key to simulating a system with a computer is being able to describe it in a way the computer understands. This is achieved by defining a set of variables that capture the system's state. Each unique combination of these variable values represents a specific condition, or "snapshot," of the system. By changing these variables according to pre-defined rules, the simulated system is essentially moved from one state to another. This process of manipulating variables to represent the system's dynamic behavior is the essence of simulation^{2,18}. The usefulness and efficiency of simulation have led to its adoption in a wide range of fields¹⁹, and it's particularly valuable for representing systems that are challenging to model with traditional methods^{20,21}. For instance, J. de Moij *et al.*²² use simulation to develop a framework to study how complex social behaviors and interventions can influence disease outbreaks. This framework is used to model the spread of COVID-19 in Virginia, demonstrating its ability to handle large-scale simulations with complex agents. Likewise, C. Adam *et al.*²³ developed a simulator for studying disease outbreaks. It was designed for educational purposes to lead users to understand the complex core mechanisms of epidemics over predicting real-world scenarios. Simulation

is also used to study how media influence crowd behavior²⁴ for understanding the conditions that could lead to or prohibit the formation of revolutionary crowds.

Agent-based modeling (ABM) is a simulation technique where independent entities (agents) follow programmed rules to interact with their surroundings and each other²⁵. This method is particularly useful for modeling and simulating intricate systems characterized by interactions between their constituent parts, like social and economic systems. In this way, Andrew J. C. and Erika F.²⁶ explore strategic group formation with an agent-based model while focusing on modeling people as group members, rather than individuals. From their side, Gayane G. *et al.* used a hybrid agent-based modeling approach, called ABMSCORE, to compare which of the computerized agents and actual humans is more efficient in finding an ideal coalition in glove games. Moreover, ABM has also been used to address issues related to mobility and smart cities. For example, to understand how changes in bus fares affect passenger choices, Baozhen Y. *et al.*²⁷ employed an agent-based simulation model to mimic passenger flow by simulating their decision-making processes. Similarly, García-Suárez A. *et al.*²⁸ adopted ABM, combined with cellular automata, to analyze the deployment of electric vehicle charging stations through microscopic traffic simulations.

In recent years, various challenges in *Flexible Manufacturing System (FMS)*^{29,30} have been addressed with simulation techniques. Running simulations can uncover surprising effects that might be missed when relying solely on theoretical models^{11,31}. Simulations offer an advantage because they can handle complex limitations that might be tricky to express in theoretical models³². Zaidi *et al.*³³ showcase this by using simulation to reveal the discrepancies between a theoretical schedule and a simulated one, even in a straightforward example. Furthermore, optimizing job shop scheduling in FMS is a major challenge, especially when mobile robots handle transportation tasks. Bilge *et al.*³⁴ addressed this by treating machines and vehicles as resources that need to be scheduled simultaneously. They introduced ten job sets and four layouts, becoming valuable benchmarks for subsequent research. Ham³⁵ tackled the job shop scheduling problem with UGV transport using a constraint programming approach. He emphasizes that both machines and UGVs have limitations that need to be considered. Similarly, Abderrahim *et al.* view workstations and vehicles as resources in their solution to job shop scheduling problems (JSSP) with automated transportation. They employ a Variable Neighborhood Search (VNS) algorithm to optimize the makespan - the total time required to

complete all jobs - by scheduling both manufacturing and transportation tasks together. Traditional job shop scheduling methods rely on precise mathematical models. However, these models struggle when dealing with complex situations with many variables and uncertainty^{11,36}. To address this limitation, simulation has been positioned as an alternative approach³⁷. Moreover, simulation allows the development of dynamic scheduling algorithms^{13,38} that are capable of optimizing scheduling while being resilient to disruptions that occur in the workshop.

Dynamic scheduling approaches offer several advantages over static scheduling approaches in job shop scheduling optimization^{39,40}. Indeed, they can deal with real-world job shop disruptions like machine breakdowns, unexpected job arrivals, and changes in due dates. Dynamic algorithms can react to these events and adjust schedules in real-time, minimizing their impact on overall production efficiency^{12,13}. Dynamic scheduling algorithms can consider real-time information about machine availability and job status to optimize resource allocation^{38,41}. By actively responding to changes, dynamic scheduling can potentially help reduce the makespan compared to static schedules that may become outdated with disruptions. Concerning this, in¹¹, The authors show that previously considered optimal schedules could become infeasible when additional constraints, such as collision avoidance between transporters, are taken into account. This can lead to deadlocks.

Furthermore, the energy consumption of manufacturing workshops is one of major interests today, especially in the current context of decarbonization⁴² and industrial sustainability^{29,43}. Particularly, the energy management of battery-powered UGVs within the workshop has been gaining attention for several years now^{14,15}. As UGV batteries deplete during operation, recharging becomes necessary, which could significantly affect the manufacturing process⁴⁴. While research acknowledges the influence of UGVs' battery management on the overall performance of the manufacturing system¹⁶, most of the prior studies have not considered this important factor in JSSP resolution^{17,45,46}. Hence, incorporating battery management constraints into models is essential for creating a more realistic representation of the production system.

Through this literature review, it appears that simulation techniques, particularly agent-based modeling, are valuable tools for studying complex systems like FMS and addressing issues like the JSSP. The significance of our simulation-based approach, in comparison with classical techniques, lies in its capability to integrate complex constraints, like space and shape constraints mentioned in^{11,30}, which are intricate

to model formally. This enables theoretical solutions to be evaluated under more constrained conditions and reduces the gap between theory and actual implementation. Besides, AIVs, which are more recent transportation robots than AGVs, have received very little attention in JSSP problems with transportation tasks, even though they have the potential to improve the overall performance of production systems and enable the transition to Industry 5.0^{47,48}. In addition, while battery management is a major issue in the current context, it has often been overlooked in previous research related to JSSP with UGVs. Likewise, the influence of UGVs fleet size within integrated JSSP with transportation tasks is not investigated enough. As a consequence, our study stands out within the literature by addressing simultaneously the following contributions:

- **Development of a Simulator:** we developed a simulator specifically designed to address the JSSP with transportation tasks performed by UGVs. Particularly, in this paper, the simulator incorporates UGV battery limitation, fleet size variation, and collision avoidance, which make it more realistic for studies and analysis.
- **Utilization of AIVs as transporters in JSSP:** unlike AGVs, which have been widely used in JSSPs, to our knowledge, AIVs have received little attention in JSSPs. Even in the previous work⁴⁹, battery limitation and fleet size were not addressed. Thus, this work introduces them and highlights their effectiveness compared with AGVs.
- **Evaluation of the additional constraints' impact on the job shop scheduling:** This work addresses JSSP by considering battery and collision avoidance constraints for each UGV type (AIV vs. AGV) and highlights AIVs' potential for efficiency and resilience.
- **Comparison of Scheduling Methods:** the paper assesses the performance of static and dynamic scheduling methods, demonstrating the advantages and disadvantages of each while comparing AGVs and AIVs in different scenarios.
- **Investigation on the effect of UGV Fleet Size with additional constraints:** The paper examines how the number of UGVs influences both transportation and production scheduling performance when collision avoidance and battery limitation constraints are considered.

3 Problem description

This paper focuses on the Job Shop Scheduling Problem (JSSP) that involves transportation tasks handled by Unmanned Grounded Vehicles (UGVs). It is assumed that the UGVs can operate until their batteries reach a certain threshold¹⁵. Once that happens, the UGV must stop all activities and head to a charging station. This aims to understand how these mandatory charging breaks during operation affect the overall production time (makespan) in the scheduling process. Besides, three types of energy consumption for UGVs are considered: idle consumption, empty consumption, and loaded consumption. Idle consumption refers to the energy used when the UGV is operational but not in motion. Empty consumption occurs when the UGV moves without carrying any load, while loaded consumption represents the energy usage when the UGV is transporting a job. These distinctions are defined for accurately modeling the UGV usage, as suggested in references^{15,50}.

The JSSP, involves organizing a set of jobs (denoted by $\mathcal{J} = \{J_1, J_2, \dots, J_I\}$) to be processed on a set of machines (denoted by $\mathcal{M} = \{M_1, M_2, \dots, M_M\}$). Each job, J_i , consists of a specific sequence of operations ($O_{i1}, O_{i2}, \dots, O_{in}$). These operations must be completed one after another. Each operation, O_{ij} (meaning operation j of job i), can only be performed on a designated machine, $M_k \in \mathcal{M}$, and takes τ_k time units to complete. An important aspect is that a machine can only handle one operation at a time, and once an operation starts, it must be finished without interruption. Additionally, each machine, M_k , has two buffers: an input buffer, B_k^I , for storing jobs waiting to be processed, and an output buffer, B_k^O , for jobs that have been processed. All jobs are located at a central loading/unloading (L/U) station at the beginning of the process. After all processing is finished, the jobs are returned to this same L/U station.

Each job, J_i , has a specific manufacturing process represented by an ordered sequence of operations. This sequence includes both transportation and processing operations: $(T_{i1}, O_{i1}, T_{i2}, O_{i2}, \dots, T_{in}, O_{in}, T_{in+1})$. Here, $T_{i,j}$ represents the movement of job J_i to the machine assigned for operation O_{ij} . For example, T_{i1} is the initial transport of J_i from the L/U station to the machine for its first operation, O_{i1} . Similarly, T_{in+1} signifies the final transport of J_i from its last operation's machine back to the L/U station. In this context, the makespan refers to the total time needed to complete all jobs. This duration is measured from the moment the first job starts its process until the very last job returns to the L/U station.

It is assumed that the battery level of the UGVs decreases gradually as they work. Three types of consumption are distinguished: idle consumption, empty consumption, and loaded consumption. All jobs have equal weight and are lower than the maximum capacity of the UGVs. The workshop has one charging station for all UGVs, but it has enough space and power to charge them all simultaneously. Once the battery level of a UGV reaches the set threshold, it stops all its activities and heads towards the charging station. However, if it reaches the threshold while transporting a job, it first completes the delivery before going to recharge. It remains in charge for a certain time and then returns to work.

4 Simulator description

This section delves into a detailed exploration of the simulator. It begins by presenting the framework upon which it is built. Next, it investigates the various elements of the user interface that allow interaction with the simulator. Finally, it delves into the core components that constitute the backbone of the simulator program.

4.1 Framework

Our simulator was developed with Netlogo*, which is both a multi-agent programming language and a modeling environment for simulating natural and social phenomena⁵¹. Designed to simulate real-world interactions, NetLogo is specifically built for modeling complex systems with multiple agents. These “agents”, which can number in the hundreds or thousands, act independently and concurrently, allowing users to explore how individual behaviors shape larger, long-term patterns. NetLogo goes beyond just running simulations - it fosters exploration. Users can “play” with simulations, adjusting conditions to see how the system reacts. Furthermore, as evoking⁵², NetLogo prioritizes ease of use, reflecting its roots as an educational tool. This is evident in its programming language, featuring high-level structures and primitives that minimize coding complexity. Additionally, comprehensive documentation is readily available.

A NetLogo program consists of 3 tabs:

- *Interface*: This is the tab for viewing and interacting with the simulation.
- *Info*: This tab provides information about the program being simulated. For example, you can enter a description of the phenomenon you are simulating, or explain how to use it or how it works.
- *Code*: this is the tab where the entire simulation code is written.

NetLogo uses *turtle* as its individual agents, and these agents are grouped into distinct categories called *breeds*. Parameters can be defined to the entire model or just to specific breeds. Importantly, these breed-specific characteristics cannot be accessed by other breeds. The simulation starts with the *setup* button, which initializes or resets the model and its environment. Once everything is prepared, the simulation can be launched using the *go* button. This button has two modes: “one time” and “forever”. In “one time” mode, the simulation runs for one iteration (action) and then stops. In “forever” mode, the simulation will continuously run until a stop condition is programmed into the code, or you press the *go* button again. The simulation time is counted in *ticks*, and each *tick* represents a simulation step.

The simulation environment, called *world*, is a 2D grid that allows the coded program to be visualized in real-time. This 2D grid is made up of tiny squares called *patches*. These patches act as the playing field for the *turtles*, and each *patch* has its own unique location, identified by *x* and *y* coordinates (*pxcor* and *pycor*). This allows us to track which agents are on specific *patches*, and vice versa, figuring out which *patch* a particular *turtle* is currently occupying.

For our purposes, it is assumed that 20 ticks correspond to 1 second, and 20 patches correspond to 1 meter. UGVs, machines, stocks (machine buffers and the L/U station), and jobs are considered as the four main *breeds* of our model.

4.2 Multi-agent system (MAS) model

The concept of MAS comes from the area of distributed artificial intelligence (DAI)⁵³. These systems do not need a central controller, but rather independent entities called “agents” that work on tasks at the same time. Therefore, MAS is characterized by a decentralized control and a parallel execution of processes^{54,55}. Each agent can act on its own to reach its goals, and/or they can collaborate to each other if they need information or skills they do not have themselves⁵⁶, which helps to study and analyze emergent behavior²³.

Our simulator is built on a MAS with four main types of agents: Transporters (UGVs), machines, stocks (machine buffers and the L/U station), and jobs. These agents work together as shown in Figure 1.

*<https://ccl.northwestern.edu/netlogo/>

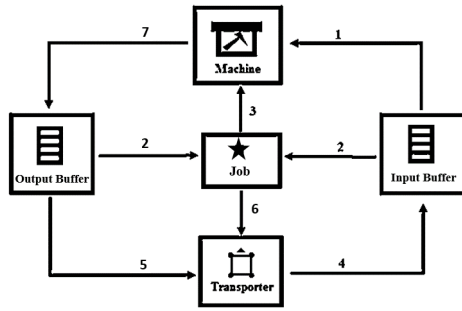


Figure 1. The MAS of the simulator

Interaction	Description
1	Machines retrieve Jobs from Stocks.
2	Stocks store Jobs.
3	Machines process Jobs.
4	Transporters deliver Jobs to Stocks.
5	Transporters pick Jobs from Stocks.
6	Transporters transport Jobs.
7	Machines put processed Products in Stocks.

Table 1. Description of the MAS interactions

4.3 Benchmark instances and workshop layouts

The experiments are based on the benchmark test instances proposed by Bilge and Ulusoy³⁴. These test instances involve four different workshop layouts, each with a loading/unloading (L/U) station and four machines. They also proposed 10 different sets of jobs, each set containing 5 to 8 jobs. Each job consists of several operations that require specific machines with corresponding processing times. The test cases are labeled "EX $\alpha\beta$," where α represents the job set and β represents the layout. Both travel times and processing times are measured in seconds. Each layout represents a physical workshop arrangement, with different locations of machines and the L/U station. This latter acts as a storage unit (stock) for raw materials and finished products. Two identical UGVs handle all transportation tasks, which always start and end at the L/U station. If there are multiple ways to get from one location to another, the fastest route is privileged.

In the original benchmark instances, the transportation tasks are carried out by two AGVs. Thus, the workshop layouts are characterized by specific designs, namely the travel orientations, as shown in Figures C.1-C.4, in Appendix C. Tables C.1-C.4 present the minimum required travel time for AGVs to travel between two locations. However, switching from AGVs to AIVs eliminates the need for pre-defined travel orientation. Unlike AGVs, AIVs can navigate and plan their routes independently. This has a significant impact on both workshop design and travel

times between locations. As shown in Figures 2-5, travel orientations are no longer required, and the resulting travel times are presented in Tables 2-5. For instance, the travel time between M2 and the L/U station in layout 4 is equal to 20s with AGVs, whereas this time drops to 8s with AIVs.

4.4 User interface

As illustrated in Figure 6, the user interface of our simulator is composed of several key elements. Each of these elements plays a specific role in facilitating interaction with the simulation:

- **Buttons:** The interface features buttons that trigger specific actions. The *setup* button functions like a game's "reset" button, initializing the simulation environment. The *go* button acts as a play/pause control, starting or stopping the simulation run.
- **Sliders:** sliders are used to vary the numerical values of a variable. In this case, they allow to set the number of jobs or transporters and to count the number of times a simulation has been run.
- **Choosers:** They enable the selection of a specific value from a predefined set for a variable. Supported value types include numbers, Boolean values (true/false), and strings. They are used, among other things, to select the layout and instance of the problem to be simulated.
- **Switches:** They enable the modification of a variable's state between two predefined options. Typically, these options can be true/false, on/off, left/right, etc. In the simulator, switches are used to activate the collision avoidance mechanism or to record the simulation details.
- **Monitors:** Once the simulation is running, you can display information about the simulated program on a monitor. In our case, this monitor keeps track of the completion time (in ticks) of each job.
- **The simulation environment (*world*):** It provides a window into the program's execution, allowing you to observe its behavior as it runs. In our simulation, the *world* represents the workshop layout with the different agents. To represent the workshop floor, the *patches* are colored differently. White patches are corridors where UGVs can move freely, while gray patches represent areas that are currently occupied.

The simulator allows users to specify the type of UGV for carrying transportation tasks. This selection is made via the chooser "robot" as shown in Figure 7:

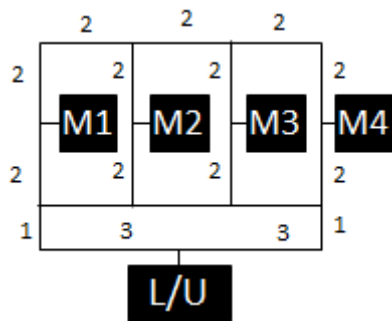


Figure 2. Layout 1 with AIVs

	L/U	M1	M2	M3	M4
L/U	0	6	8	8	6
M1	6	0	6	8	10
M2	8	6	0	6	8
M3	8	8	6	0	6
M4	6	10	8	6	0

Table 2. Travel times of AIVs on layout 1

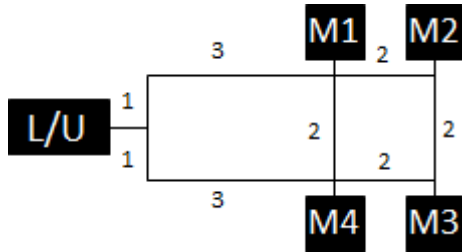


Figure 3. Layout 2 with AIVs

	L/U	M1	M2	M3	M4
L/U	0	4	6	6	4
M1	4	0	2	4	2
M2	6	2	0	2	4
M3	6	4	2	0	2
M4	4	2	4	2	0

Table 3. Travel times of AIVs on layout 2

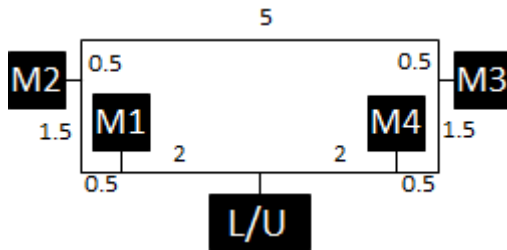


Figure 4. Layout 3 with AIVs

	L/U	M1	M2	M3	M4
L/U	0	2	4	4	2
M1	2	0	2	6	4
M2	4	2	0	6	6
M3	4	6	6	0	2
M4	2	4	6	2	0

Table 4. Travel times of AIVs on layout 3

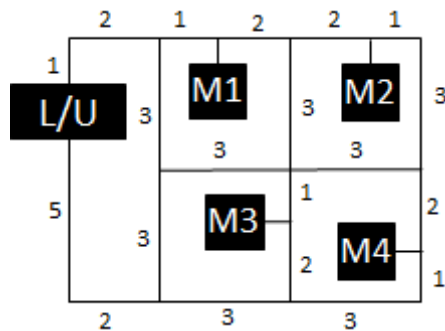


Figure 5. Layout 4 with AIVs

	L/U	M1	M2	M3	M4
L/U	0	4	8	10	14
M1	4	0	4	6	10
M2	8	4	0	6	6
M3	10	6	6	0	6
M4	14	10	6	6	0

Table 5. Travel times of AIVs on layout 4

- if *robot = agv*: the transportation tasks are performed by AGVs. In this case, the UGVs move along predefined paths dictated by the workshop’s layout.
- if *robot = aiv*: the transportation tasks are performed by AIVs. Unlike AGVs, AIVs are not restricted to following predetermined paths. They possess the intelligence to navigate and find the most suitable route on their own. This autonomy translates to greater flexibility and agility, allowing them to maneuver around both stationary and dynamic obstacles.

Users are also able to activate the collision avoidance constraint during simulations using the “*without-collisions?*” switch, as depicted in Figure 8. When this constraint is activated, collisions between UGVs are not tolerated. Reference¹¹ explains how this mechanism works with AGVs. However, when the transportation tasks are performed by AIVs, they employ two safety zones⁴⁹: a larger zone for obstacle detection and speed reduction, and a smaller zone for immediate stopping. Within the larger zone, both AIVs halve their speed. Inside the smaller zone, they determine which one has the right of way according to the priority

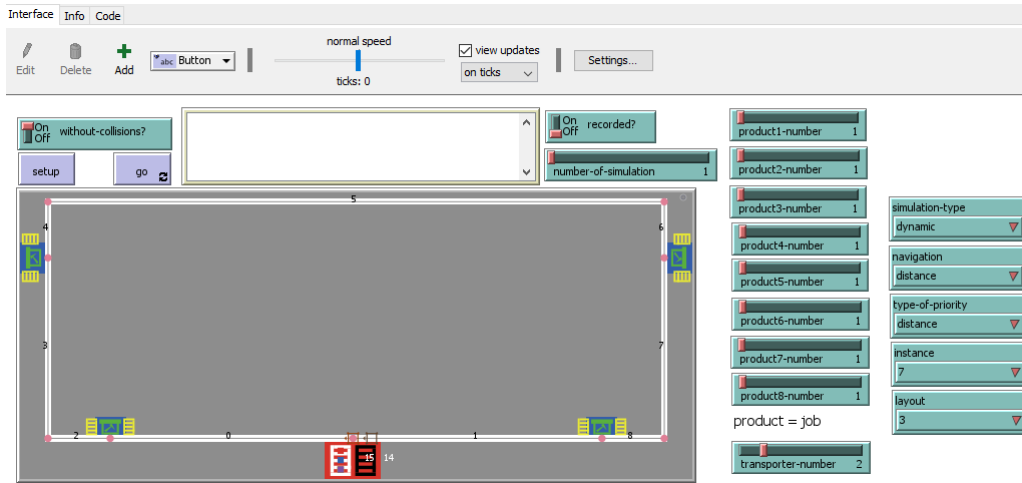


Figure 6. The simulation user interface

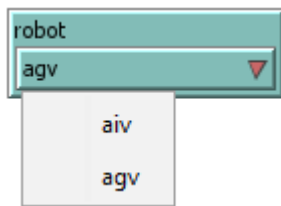


Figure 7. The choice of UGV for carrying the transportation tasks



Figure 8. Activation of the collision avoidance constraint during simulations

management system. The one with the priority does a go-around maneuver, further reducing its speed, while the other has to stop. Once the path is clear, both AIVs gradually accelerate back to their cruising speed and resume their tasks as normal.

4.5 Simulation types

The simulator is designed to simulate two types of scheduling:

- Advanced scheduling: the simulator takes the results of a schedule that has been defined beforehand. In this scenario, task assignments are predetermined, and the simulator acts as an execution tool, running the simulation based on this pre-defined plan.
- Dynamic scheduling: the scheduling is generated dynamically by a simulation-optimization (sim-optim) approach through a dynamic scheduling optimization algorithm embedded in the simulator.

The type of scheduling simulation is selected using the chooser “simulation-type”. For example, in Figure 9, the

simulator is set on dynamic scheduling simulation. However,

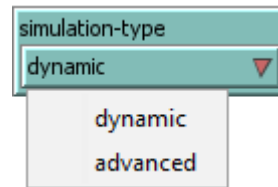


Figure 9. The choice of the scheduling simulation type

in both cases, the UGVs have to transport the jobs from one place to another. When a UGV receives an instruction (an order), it first locates the job to be transported before moving to find it. Then, once it has retrieved it, it sends it to its destination.

Algorithms 1-6 describe how the simulation program works.

Algorithm 1 Pseudocode of the simulation main program

Require: available jobs, machines, stocks and robots

Ensure: simulate the job-shop scheduling

```

1: procedure GO
2:   TRANSPORT-JOBS
3:   PROCESS-JOBS
4:   if all jobs have been processed then
5:     print makespan      ▷ or any other relevant
                             information
6:     stop the simulation
7:   end if
8: end procedure

```

Algorithm 2 Pseudocode of the transportation task program**Require:** available jobs and robots**Ensure:** Execute the transportation task scheduling

```

1: procedure TRANSPORT-JOBS
2:   for each available transporter  $T_p$  do
3:     if simulation-type = "advanced" then
4:       if  $orders\_list$  is empty then
5:         FIND-ORDERS-FROM-
SCHEDULING-MATRIX
6:       else
7:         if  $current\_order$  is empty then
8:            $current\_order \leftarrow orders\_list[0]$  ▷
get the first element of orders-list
9:         end if
10:        end if
11:       else
12:         if  $current\_order$  is empty then
13:           FIND-JOB-TO-TRANSPORT ▷ run
the dynamic scheduling algorithm
14:         end if
15:       end if
16:       if  $status = idle$  and  $battery\_level > threshold$ 
then
17:         PLAN-TRAVEL( $current\_robot$ )
18:       else
19:         go-to  $charge\_station$ 
20:       end if
21:     end for
22: end procedure

```

Algorithm 3 Pseudocode of job transportation program**Require:** the robot and its current task**Ensure:** the robot to execute the transportation task

```

1: procedure PLAN-TRAVEL( $current\_robot$ )
2:   read  $current\_order$  ▷ to get the job location and its
next destination
3:   if  $job\_picked? = false$  then
4:     MOVE-TO-PICK-JOB
5:   else
6:     MOVE-TO-DESTINATION
7:   end if
8: end procedure

```

Algorithm 4 Pseudocode of job picking program**Require:** the robot and the job location**Ensure:** the robot to pick the job

```

1: procedure MOVE-TO-PICK-JOB( $current\_robot$ )
2:   if reach the job location then
3:     if the job is being processed then
4:       wait
5:     else
6:       pick-job
7:        $job\_picked? \leftarrow true$ 
8:     end if
9:   else
10:    move-on
11:     $job\_picked? \leftarrow false$ 
12:  end if
13: end procedure

```

Algorithm 5 Pseudocode of job delivering program**Require:** the robot and the job destination**Ensure:** the robot to deliver the job

```

1: procedure MOVE-TO-DESTINATION( $current\_robot$ )
2:   if reach the destination then
3:     deliver-job
4:      $job\_picked? \leftarrow false$ 
5:      $current\_order \leftarrow []$  ▷ empty list
6:   else
7:     move-on
8:      $job\_picked? \leftarrow true$ 
9:   end if
10: end procedure

```

Algorithm 6 Pseudocode of job processing program**Require:** available machines**Ensure:** the jobs processing

```

1: procedure PROCESS-JOBS
2:   for each machine  $M_k$  do
3:     if input-buffer is not empty then
4:       if simulation-type = "advanced" then
5:         process job according to the machine
processing schedule
6:          $status \leftarrow occupied$ 
7:       else
8:         process job in FIFO mode
9:          $status \leftarrow occupied$ 
10:      end if
11:    end if
12:    if the job processing is completed then
13:      put job in the output-buffer
14:       $status \leftarrow free$ 
15:    end if
16:  end for
17: end procedure

```

4.5.1 Advanced scheduling

When the simulator is set on advanced scheduling simulation, the simulator takes the advanced scheduling matrix as input. This comprehensive matrix serves as a central hub for all task-related information. It details all tasks involved, specifies who is assigned to each task, and defines the order in which they should be executed. As an

J_1	J_1	J_2	J_1	J_2	J_2	J_1	J_2
M_1	M_2	M_1	M_4	M_3	M_2	L/U	L/U
T_2	T_1	T_1	T_2	T_1	T_2	T_1	T_2
O_{11}	O_{12}	O_{21}	O_{33}	O_{22}	O_{23}	R	R

Table 6. A structure of an advanced scheduling matrix

illustration, Table 6 presents an advanced scheduling matrix for a scenario with two jobs, each involving three operations. This table provides a detailed overview of the schedule. The first row identifies the jobs (denoted as J_i). The second row specifies the destination for each operation, which can be a machine (denoted as M_k) or the loading/unloading (L/U) station. The third row indicates the transporter (denoted as T_p) responsible for moving jobs between locations. Finally, the fourth row details the specific operations (denoted as O_{ij}) that each job must undergo. The letter "R" signifies the job's return to the L/U station after processing. The information in this table is best understood by reading each column. Take the first column, for instance. It tells us that transporter 2 has to move job 1 to machine 1 for its first operation. Following this logic, we can extract all the transportation tasks for each transporter and create their individual order lists. Similarly, production tasks can be extracted for each machine from the schedule.

For this paper, the VNS (Variable Neighborhood Search) algorithm developed by Abderrahim *et al.*⁵⁷ was utilized. This approach relies on switching between different neighborhoods to escape local optima. It uses two types of local searches: vertical and horizontal. The vertical search aims to improve the schedule within the current task allocation for machines. The horizontal search explores different transportation assignments for vehicles. The algorithm works by representing schedules as two strings. One string encodes the tasks assigned to each machine, and the other encodes the transportation tasks for the vehicles. It then performs a two-step local search:

1. Local Search on Transportation String: The algorithm searches within the current set of possible transportation assignments to find a better option.
2. Local Search on Production String: Given the new transportation string, the algorithm performs a local search on the machine task assignments to find the best

combination that minimizes the makespan considering the current transportation plan.

This two-step search process is repeated for all possible transportation options until a stopping criterion is met (e.g., no improvement found) or a better combination of production and transportation tasks is identified. The entire process is then iterated with new transportation options until another stopping criterion is reached. Readers are invited to refer to⁵⁷ for a more detailed explanation of the VNS algorithm.

In this study, we opted to use the VNS algorithm in its unmodified form due to its suitability for our requirements. Notably, the algorithm facilitated the exploration of varying UGV fleet sizes and enabled us to address benchmark instances encompassing both scenarios with and without job returns to the L/U station.

4.5.2 Dynamic scheduling

For dynamic scheduling, the simulator employs a sim-optim approach, where it simulates and optimizes the schedule simultaneously. Whenever a transporter becomes available, the simulator triggers the optimization algorithm. This algorithm analyzes the current workshop state and determines the next job for the transporter. Transportation tasks are represented as triplets (J_i, d, O_{ij}) , where J_i is the job to transport, d is its designated destination (a machine or the L/U station), and O_{ij} (or R) is the operation to undergo. Meanwhile, the machines process jobs in FIFO (first in, first out) mode according to the order in which they arrive in their input buffer.

In this paper, the dynamic optimization method developed by Yiyi *et al.*³⁸ was employed and adapted for our needs. This method, developed for multi-objective optimization, lies within the Pareto-based approaches. All objectives are improved at the same time using a dynamic lexicographic ranking system. However, this ranking system requires decision-makers (the transporters) to figure out which objectives are most important before starting the process. In practice, it can be tricky to come up with a definitive ranking of all the objectives. Additionally, lexicographic rankings do not allow for small improvements in a more important objective to be balanced out by bigger drawbacks in a less important objective. To overcome these drawbacks, the dynamic lexicographic ranking system is combined with entropy and goal programming (to rank the objectives). When assigning transportation tasks, the system considers how uncertain (how much randomness there is) in achieving different objectives. This uncertainty is reflected in the

entropy value of each objective. Based on real-time data, like a job's next steps, locations of UGVs, or backlog in buffers, the entropy values are constantly updated. Next, decision-makers typically set desired achievement levels for each objective beforehand. Then, the system finds a solution that comes closest to meeting those targets by using goal programming. This approach is an improvement on the work carried out by Eloundou⁵⁸.

Our approach differs from Yiyi *et al.*'s original method in two ways. First, the focus is on just three objectives: minimizing the makespan (Cmax), minimizing the transportation time, and minimizing the distance to the destination. Second, the objectives ranking and their entropy values calculation were adapted. Our adjustments now take into account the physical layout of the workshop area and the processing steps for each job, ensuring alignment with our benchmark instances. However, the structure of the method and the main algorithms remain the same. For more details, please refer to³⁸.

5 Experiments

This work aims to conduct a comparative study between AIVs and AGVs within integrated JSSP with transportation tasks. To do so, the experiments conducted during this study focus on three main objectives:

- **Incorporating Transporter Battery Constraint:** The battery limitation of UGVs was integrated into our simulation model. This allows us to investigate the impact of this constraint on scheduling decisions within the manufacturing system.
- **Comparing Scheduling Strategies (or Types):** We compare the performance of static and dynamic scheduling approaches when applied to the same test instances. This comparison will help us understand how each approach handles the additional constraint of UGVs.
- **Varying the UGV fleet size:** We compare the execution of each scenario with a different number of UGVs, ranging from 2 to 5. This allows us to observe the influence of the fleet size in the makespan minimization for each instance. It should be noticed that adding more than 5 UGVs won't significantly impact the makespan. There might even be situations where there are more UGVs than tasks, making some of them useless.

To achieve our research goals, the following experimental protocol is adopted:

1. Develop an algorithm that specifically addresses the UGVs' battery management constraint, as explained in Section 3. UGVs in our simulation require recharging when their battery level drops to 20%. This choice is justified by the study of Qazi S. K. and Yoshinori S.⁵⁹ that suggest the 20% threshold both preserves UGV battery's life cycle and limits the impact of recharging on overall system performance. Upon reaching the charging station, they undergo a 5-second charging process, where their battery level rises by 20%. This duration aligns with the timescale observed in the benchmark used. Following the completion of this charging period, the UGV will prioritize the execution of any pending transportation tasks. In the absence of such tasks, it will resume the charging process until its battery level reaches 80%. Indeed, it is recommended to maintain the UGV's battery state-of-charge (SoC) within a range of 20% to 80% to optimize battery health and lifespan⁶⁰. Moreover, based on references^{15,50}, it was assumed three different battery depletion rates: 1mA (milliamper) per second while idle, 5mA per second while moving empty, and 7mA per second when transporting a job. As assumed in the work of Moussa Abderrahim *et al.*¹⁵, the UGV has a maximal capacity of 100Ah (ampere-hour), corresponding to 100% of battery level. Thus, a UGV standing idle for an hour consumes $0.001A/s \times 3600s = 3.6A$, i.e. 3.6% of the maximum battery level.
2. Using the well-known benchmark instances of Bilge *et al.*³⁴, the VNS method developed by Abderrahim *et al.*⁵⁷ is employed to generate a set of advanced schedules. These schedules varied the number of UGVs from 2 to 5. It was decided not to explore scenarios with more than 5 UGVs for two reasons. First, such scenarios could lead to situations where there are more UGVs than actual jobs. Second, in most of all cases, the makespan exhibited minimal difference between using 4 and 5 UGVs. However, varying the number of UGVs allows us to better analyze how battery recharging affects the initial schedules.
3. Employ the dynamic scheduling approach developed by Yiyi *et al.*³⁸ to the same scenarios and settings used for advanced schedules. First, this allowed us to directly compare the performance of static scheduling versus dynamic scheduling. Second, by introducing the robot battery recharging constraint, we could

observe how the dynamic scheduling algorithm adapts to this disturbance.

4. The simulations (of advanced and dynamic schedules) were conducted under the four (04) following scenarios to achieve the targeted contributions previously mentioned:
 - **Unconstrained Scenario (US):** neglecting both collision avoidance and battery constraints;
 - **Collision Avoidance Scenario (CAS):** considering solely the collision avoidance constraint, but not the battery constraint;
 - **Battery Limited Scenario (BLS):** considering solely the battery constraint, but not the collision avoidance constraint;
 - **Constrained Scenario (CS):** considering both collision avoidance and battery constraints.
5. When the battery constraint is considered, the average initial SoC across the UGVs fleet is assumed to be 50%. In other words, the sum of the initial SoC of each UGV divided by the fleet size gives the value of 50%. This assumption is inspired by the study of Ozan Yesilyurt *et al.*⁶¹ where they set the initial SoC of each UGV to 50%. However, to prevent all the UGVs from needing to recharge simultaneously, and then avoid any downtime in the manufacturing process, it was decided in this work to set different initial SoC for each UGV.
6. To realize a comprehensive performance comparison, all experiments within each of the aforementioned scenarios were conducted with both AGVs and AIVs, under the same conditions with the same assumptions, while varying the fleet size.
7. Each simulation was replicated 30 times, with the makespan recorded for each run. Subsequently, the average and standard deviation of the makespan values were calculated.

6 Results and discussion

The results of the experiments are presented in separate tables in Appendix A. Tables A.1-A.4 detail the findings for AGVs, and Table A.5 presents the results for AIVs. The VNS column shows the outcome after optimization using the VNS algorithm developed by Abderrahim *et al.*⁵⁷. We labeled the simulations based on the scheduling technique (AS for Advanced Schedules and DS for Dynamic Schedules) and the experiment scenario. For example, AS-US refers to the simulation of advanced schedules in the unconstrained scenario. However, it should be noted that

there is no unconstrained scenario with AIVs because collision avoidance is inherent to their operation. This explains the absence of these columns in Table A.5. In addition, the results presented are the average of the makespan recorded over 30 runs, followed by the standard deviation in brackets.

In the AGV simulations, the AS-US cases exhibit minimal variation (standard deviation below 1). This is because the advanced schedule is repeated without external disruptions, leading to consistent results. However, the AS-CAS cases show occasional spikes in standard deviation. This indicates deadlocks encountered and resolved during the simulations. As presented in the previous work¹¹, an algorithm is developed for deadlock resolution by permuting the remaining transportation tasks of two AGVs, causing a deviation from the initial schedule. This issue is alleviated with AIVs due to their inherent collision avoidance capabilities. Moreover, the AS-BLS case exhibits significant variations in AGV simulation results, with a few exceptions. Indeed, despite a uniform initial SoC of 50% for the fleet, individual UGVs have different initial SoC. Consequently, the frequency of recharging for each AGV fluctuates across simulations, contributing to the observed variations. This phenomenon is also present in AIV simulations, as incorporating the battery constraint inherently increases standard deviation.

In contrast, dynamic scheduling simulations, regardless of UGV type (AGV or AIV), exhibit significantly higher result variability. This stems from the inherent characteristic of dynamic scheduling, where each UGV's schedule adapts based on real-time conditions during each simulation run. Consequently, the inclusion of additional constraints exacerbates the overall volatility inherent to dynamic scheduling.

Complementary analysis of the results reveals the following key observations:

- **Globally, in the unconstrained scenario (US), advanced schedules (AS) yield better makespan than dynamic schedules (DS):** AS utilizes an optimization process to establish a comprehensive plan for all tasks from initiation to completion. This method identifies the most efficient configuration, resulting in a fully optimized schedule. Each UGV and machine is assigned all its tasks within this predefined plan. Conversely, DS employs an adaptive approach where task optimization occurs concurrently with the manufacturing process. UGVs operate based on real-time data, lacking prior knowledge of subsequent tasks.

- **AIVs can mitigate DS drawbacks:** Interestingly, a comparison of the AGVs AS-US case (advanced scheduling without constraints) with the AIVs DS-CAS case (dynamic scheduling with collision avoidance) in Figure 10 reveals comparable average performance. This suggests that even under DS with the additional constraint of collision avoidance, AIVs maintain performance levels similar to AS with AGVs. This highlights the potential of AIVs to mitigate the drawbacks of DS, potentially offering a balance between efficiency and adaptability.

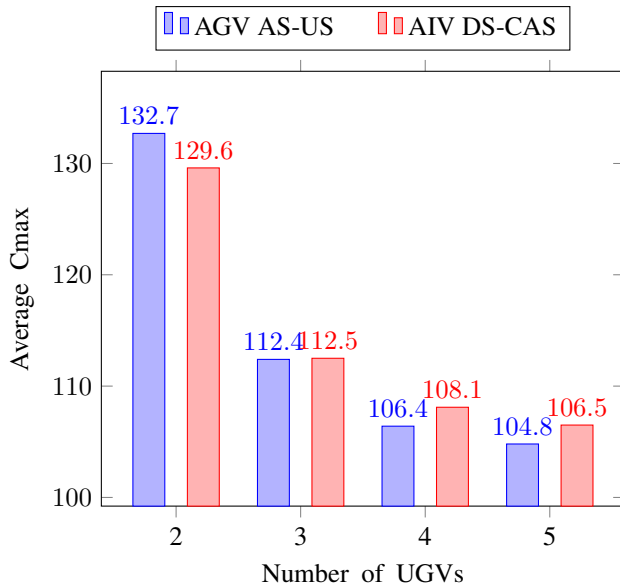


Figure 10. Comparison between the AGV AS-US scenario and the AIV DS-CAS scenario.

- **Considering additional constraints typically leads to a longer makespan:** As mentioned in reference¹¹ regarding collision avoidance, and now evident for battery limitations as well, including these additional constraints generally increases makespan, regardless of the type of UGV used for transportation tasks. Figure 11 shows this perfectly. It combines the Gantt diagrams of EX22 simulated with two AGVs, under the unconstrained scenario (US) and the battery limitation scenario (BLS). In particular, it can be seen that under the US scenario, AGV1 starts $T_{O_{4,3}}$ 2 seconds after completing $T_{O_{6,2}}$, i.e. the minimum required time to move from machine 2 to machine 3 in layout 2. But under the BLS, there is a nearly 30-second delay before AGV1 starts $T_{O_{4,3}}$ because its battery needs to be recharged. This unexpected operation disturbs the initial schedule. In contrast, making such observations with DS is challenging due to the inherent variability in UGV schedules across simulations. However, a notable increase in makespan

is consistently observed in cases involving at least one additional constraint, regardless of the UGV type (AGV or AIV).

- **AIVs exhibit better performance than AGVs:** The observation of the simulation results across all instances reveals a noteworthy improvement in makespan when AGVs are replaced by AIVs in the same experimental scenario. Under the CAS, for example, the simulation of advanced schedules with two AIVs improves the makespan by an average of 9% across all instances, compared to the simulation with two AGVs. This translates to an average time savings of approximately 12 seconds when switching from AGVs to AIVs. The observed reduction in makespan persists even with an increase in the number of UGVs, as evidenced in Figure 12.
- **AIVs are better suited to constrained environments than AGVs:** Excluding the AS simulations with only two (02) UGVs, simulations employing AIVs under the constrained scenario (CS) consistently demonstrate a better average makespan compared to simulations utilizing AGVs within the CAS. This observation underscores the enhanced flexibility inherent to AIVs and their aptitude for navigating in more complex and constrained environments. Furthermore, we observe that employing DS with AGVs increases the makespan when the number of AGVs exceeds 4 (see Figure 12b). However, with AIVs, the reduction in makespan continues even with 5 AIVs, albeit at a diminished rate. This shows the potential to utilize a greater number of AIVs due to their enhanced routing flexibility and inherent collision avoidance capability.
- **The influence of fleet size depends on UGV type:** Consistent with prior observations, increasing the number of UGVs generally reduces makespan as observed in Figure 12. However, a noteworthy finding emerges when comparing AGVs and AIVs. In the AS-CAS case, a fleet of only 2 AIVs achieves almost identical results (avg. Cmax = 121.6s) than a fleet of 4 AGVs (avg. Cmax = 121s). Even a fleet of 5 AGVs demonstrates a marginal efficiency gain of just 3.2% compared to 2 AIVs. Next, in the AS-CS case, a fleet of 3 AIVs achieves better results (avg. Cmax = 117.3s) than a fleet of 5 AGVs (avg. Cmax = 130.3s). Even more impressive, a fleet of 3 AIVs in the AS-CS case outperforms a fleet of 5 AGVs in the AS-CAS case (avg. Cmax = 117.7s). This suggests that the battery constraint, even when considered, does not hinder the

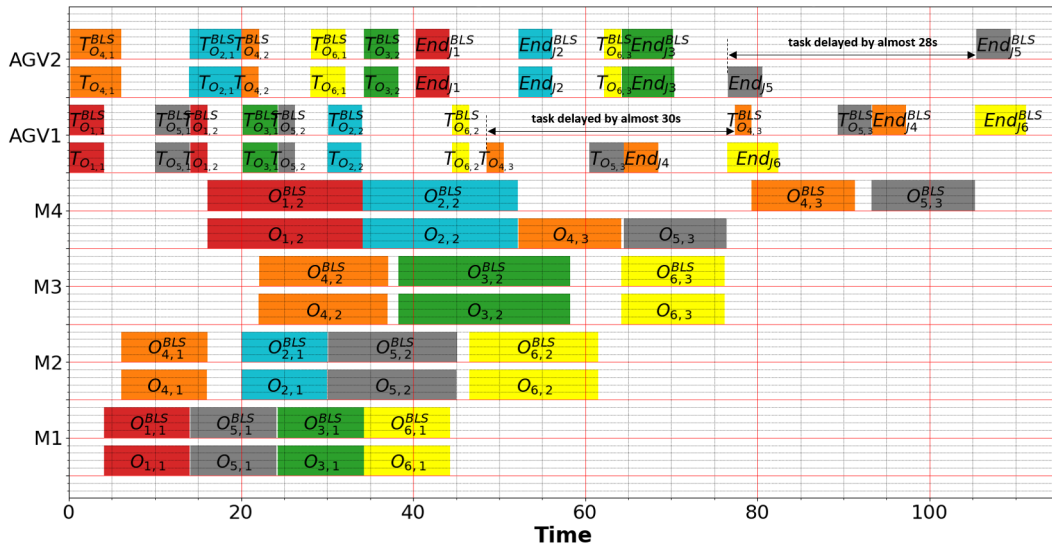


Figure 11. Combined EX22 Gantt diagrams in both unconstrained and battery limitation scenarios

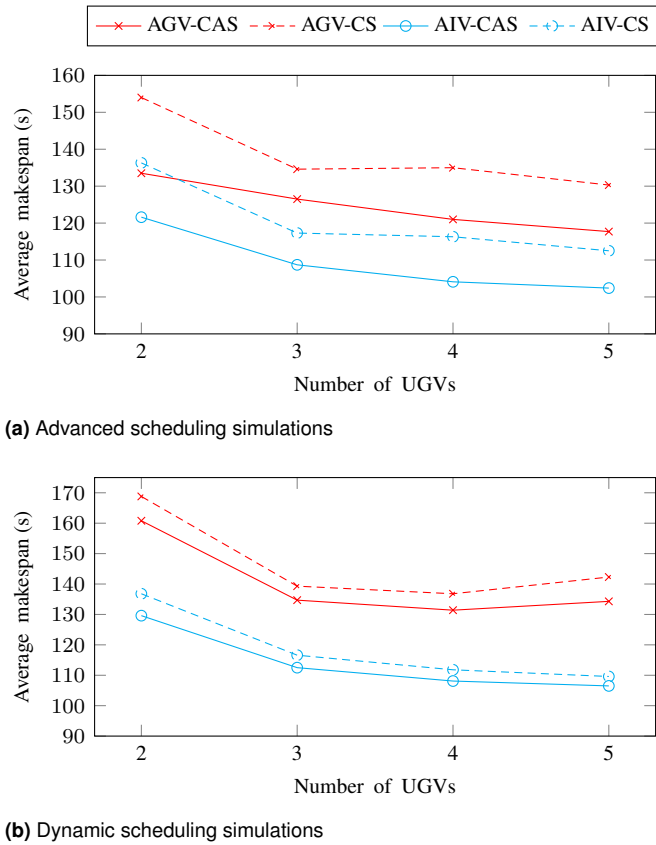
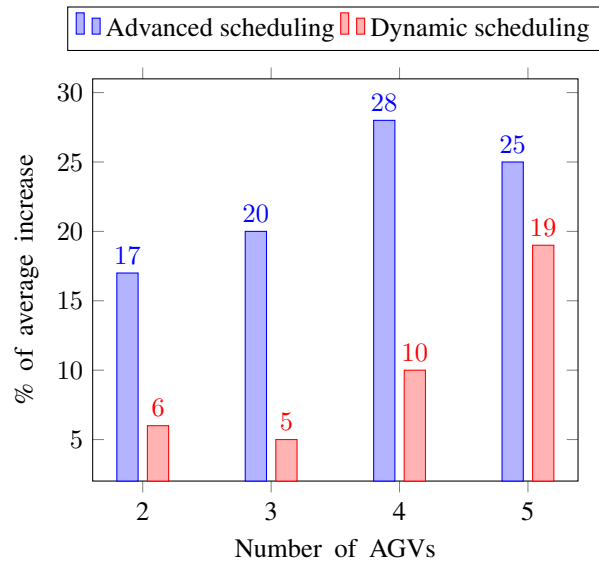


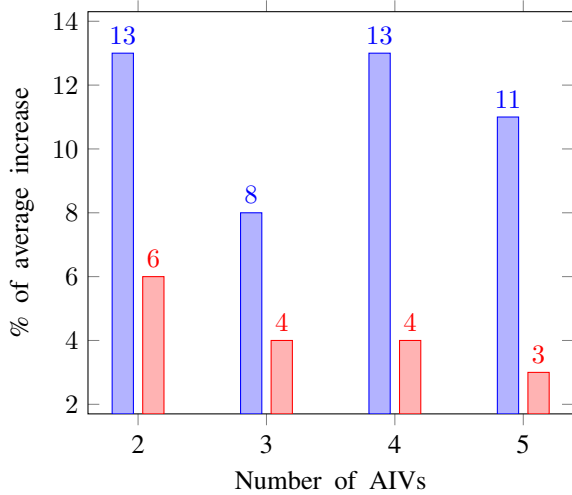
Figure 12. Evolution of the average makespan based on the number of UGVs for both dynamic and advanced schedules, under collision avoidance and constrained scenarios.

ability of AIVs to outperform AGVs. This advantage becomes even more pronounced under DS, where a fleet of 3 AIVs in the DS-CS case demonstrates significantly greater efficiency (avg. Cmax = 116.6s) compared to a fleet of 5 AGVs in the DS-CAS case (avg. Cmax = 134.3s).

- **Dynamic scheduling outperforms advanced scheduling when multiple jobs share identical processing steps:** As presented in Appendix B, job sets 6 and 8 have a striking characteristic: a significant portion of jobs share identical processing sequences. In job set 8. For instance, all jobs follow the $M_2 \rightarrow M_3 \rightarrow M_4$ manufacturing process. This phenomenon fosters deadlocks, bottleneck formation, and increased robot encounters during the simulation of advanced schedules. In contrast, simulations of dynamic schedules demonstrate the absence of deadlocks and a lower risk of bottlenecks. Furthermore, dynamic scheduling outperforms advanced scheduling in 91% of the simulations conducted on job sets 6 and 8, employing AIVs under the CS. This success rate diminishes to 59% when AGVs are utilized.
- **Dynamic scheduling is more adaptable to additional constraints than advanced scheduling:** Regardless of the scheduling method, additional constraints lead to a makespan increase. However, this does not occur equally in both cases. The impact of collision avoidance and battery limitations on scheduling is demonstrably greater for the advanced schedules, as shown in Figure 13. For example, in Figure 13a, it can be seen that for a scenario with 4 AGVs, these constraints increased the makespan by 28% in advanced scheduling, while the impact on dynamic scheduling was only 10%.
- **Workshop layouts significantly impact the behavior of UGV fleets:** Beyond the impact of UGV fleet size, the workshop layout itself plays a significant role in influencing their effectiveness as shown in Figures 14



(a) Simulations conducted with AGVs

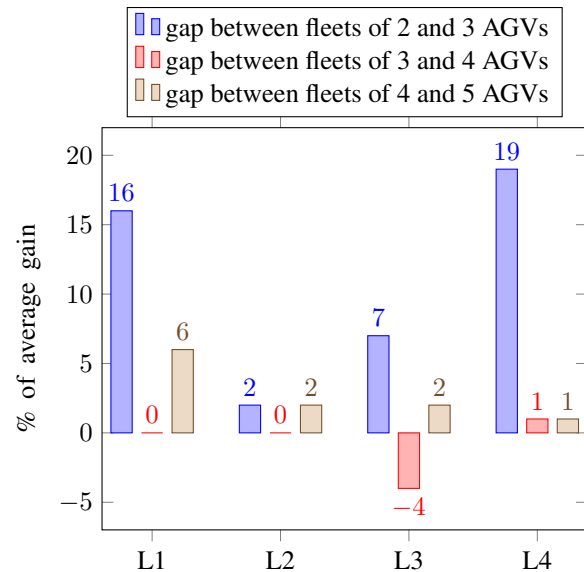


(b) Simulations conducted with AIVs

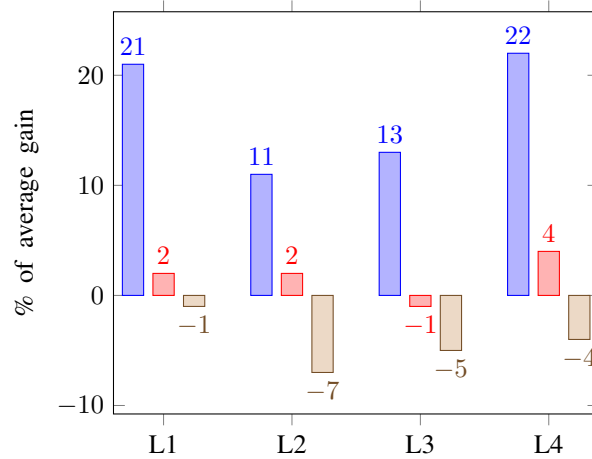
Figure 13. Evolution of makespan increase according to the fleet size for advanced scheduling and dynamic scheduling methods.

and 15. In the AS-CS case, increasing the number of AGVs from 2 to 3 yielded an average productivity gain of 16% for layout 1, 19% for layout 4, but only 2% for layout 2, and 7% for layout 3. This suggests that layouts 1 and 4, with greater routing flexibility compared to layouts 2 and 3, can better accommodate the increased AGV traffic. However, further increasing the fleet size from 3 AGVs to 4 generally led to a decrease in productivity. Notably, on layout 3, adding an extra AGV resulted in a -4% decline compared to 3 AGVs. This observation is further corroborated by analyzing the DS-CS case results. Here, increasing the fleet size from 4 AGVs to 5 leads to a productivity decline across all layouts. However, the impact is more pronounced in layouts 2 and 3, where losses reach -7% and -5% respectively. Layouts 1 and 4, with their

inherent flexibility, mitigate the decline to -1% and -4%, respectively. Likewise, the influence of layout on AIV performance aligns with the observations for AGVs. Increasing the AIV fleet size from 3 to 4 in the AS-CS case results in productivity gains of 2% and 9% for layouts 1 and 4, respectively. However, layouts 2 and 3 exhibit performance degradation, with productivity losses of -4% and -5%, respectively. Nevertheless, in DS-CS cases, increasing the number of AIVs does not lead to productivity losses, even though the gains observed tend to diminish.



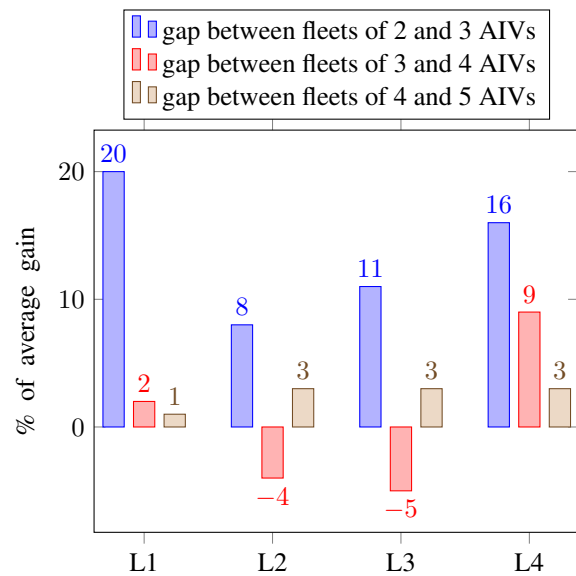
(a) AS-CS simulations with AGVs



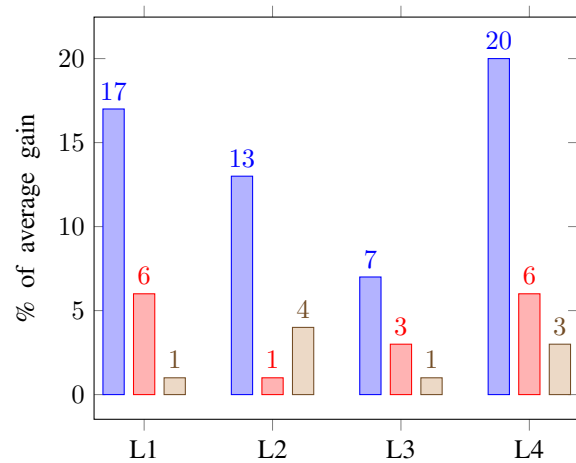
(b) DS-CS simulations with AGVs

Figure 14. Variation in system productivity based on AGV fleet size for each layout.

Beyond the technical considerations explored in the analyses above, a significant challenge associated with adopting any new technology in industries is overcoming resistance due to change aversion and uncertainty⁶². In this regard, the simulator presented in this paper offers a valuable decision-support tool. It allows for the



(a) AS-CS simulations with AIVs



(b) DS-CS simulations with AIVs

Figure 15. Variation in system productivity based on AIV fleet size for each layout.

exploration of AIVs within JSSPs and the quantification of potential productivity gains. Our findings, for example, demonstrate that a fleet of 3 AIVs outperforms a fleet of 5 AGVs. This information can be instrumental for decision-makers, guiding future investments and optimizing resource allocation. Nevertheless, despite potentially higher initial investment costs compared to AGVs, AIVs can offer a more favorable overall cost structure. This stems from two factors: reduced infrastructure modifications required for AIVs due to their inherent flexibility, and the possibility of achieving similar performance with a smaller AIV fleet size compared to AGVs. However, implementing AIVs necessitates investment in personnel training and education, as these UGVs share the workspace with human workers. This potential intrusion into the human worker's environment may lead to initial acceptance challenges.

7 Conclusion

This paper introduced a novel multi-agent system-based simulator for addressing the Job Shop Scheduling Problem (JSSP) with Unmanned Ground Vehicles (UGVs) handling transportation tasks. The simulator incorporates real-world constraints often overlooked in previous research, such as collision avoidance, the UGV fleet size, and the limited battery capacity of UGVs. By enabling users to explore the impact of these factors on both advanced and dynamic scheduling with different robot types (AGVs vs. AIVs), this simulator provides valuable insights for optimizing JSSP in constrained manufacturing environments.

Our findings reveal that while advanced scheduling shows generally a better makespan than dynamic scheduling, the latter is more resilient to additional constraints such as collision avoidance and battery limitations. Besides, dynamic scheduling is more efficient when dealing with instances where several jobs have identical processing steps. Additionally, when both AGVs and AIVs are employed under the same scenario, simulations typically show that AIVs outperform AGVs regardless of the scheduling method. Even when they are employed under the constrained scenario, they still outperform the AGVs used in the collision avoidance scenario solely.

In addition, the transition from AGVs to AIVs constitutes a strategic technological shift necessitating managerial precaution. To ensure well-informed decision-making, a data-driven approach is paramount. This paper presents a comparative analysis to highlight the efficacy of AIVs relative to AGVs. We quantify the productivity enhancements achievable through AIVs, even within the limitations of real-world operating environments. Therefore, both the simulator employed in this study and the study itself serve as valuable tools to empower managers in their pursuit of optimizing production system performance.

Future research directions could involve expanding the simulator's capabilities to handle even more complex scenarios, such as integrating machine breakdowns or rush orders and increasing variability in the workshop. Besides, exploring alternative scheduling algorithms within the simulator framework could offer further optimization opportunities for JSSP with UGVs. We could also explore the integration of battery constraints as an optimization criterion. This approach would have the potential to enhance the energy efficiency of UGVs and mitigate disruptions to the overall production system caused by recharging requirements. Besides, as mentioned in the discussion, acceptance can be a significant issue that managers must

consider when implementing AIVs. This aspect should be addressed in future research efforts.


References


1. Lehman R. *Computer Simulation and Modeling: An Introduction*. Lawrence Erlbaum Associates, 1977. ISBN 9780470992968.
2. Pritsker AAB. Compilation of definitions of simulation. *SIMULATION* 1979; 33(2): 61–63. DOI:10.1177/003754977903300205.
3. Chen R, Yang B, Li S et al. A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem. *Computers & Industrial Engineering* 2020; 149: 106778. DOI:https://doi.org/10.1016/j.cie.2020.106778.
4. Ahmadian MM, Salehipour A and Cheng T. A meta-heuristic to solve the just-in-time job-shop scheduling problem. *European Journal of Operational Research* 2021; 288(1): 14–29. DOI:https://doi.org/10.1016/j.ejor.2020.04.017.
5. Cronin C, Conway A and Walsh J. State-of-the-art review of autonomous intelligent vehicles (aiv) technologies for the automotive and manufacturing industry. In *2019 30th Irish Signals and Systems Conference (ISSC)*. pp. 1–6. DOI:10.1109/ISSC.2019.8904920.
6. Hu H, Jia X, He Q et al. Deep reinforcement learning based agvs real-time scheduling with mixed rule for flexible shop floor in industry 4.0. *Computers & Industrial Engineering* 2020; 149: 106749. DOI:10.1016/j.cie.2020.106749.
7. Martin L, González-Romo M, Sahnoun M et al. Effect of human-robot interaction on the fleet size of aiv transporters in fms. In *2021 1st International Conference On Cyber Management And Engineering (CyMaEn)*. pp. 1–5. DOI: 10.1109/CyMaEn50288.2021.9497273.
8. Aizat M, Azmin A and Rahiman W. A survey on navigation approaches for automated guided vehicle robots in dynamic surrounding. *IEEE Access* 2023; 11: 33934–33955. DOI: 10.1109/ACCESS.2023.3263734.
9. Destouet C, Tlahig H, Bettayeb B et al. Flexible job shop scheduling problem under industry 5.0: A survey on human reintegration, environmental consideration and resilience improvement. *Journal of Manufacturing Systems* 2023; 67: 155–173. DOI:10.1016/j.jmsy.2023.01.004.
10. Jian Z, Guofu D, Yisheng Z et al. Review of job shop scheduling research and its new perspectives under industry 4.0. *Journal of Intelligent Manufacturing* 2019; 30. DOI: https://doi.org/10.1007/s10845-017-1350-2.
11. Sanogo K, Mekhalef Benhafssa A, Sahnoun M et al. A multi-agent system simulation based approach for collision avoidance in integrated job-shop scheduling problem with transportation tasks. *Journal of Manufacturing Systems* 2023; 68: 209–226. DOI:10.1016/j.jmsy.2023.03.011.
12. Parveen S and Ullah H. Review on job-shop and flow-shop scheduling using. *Journal of Mechanical Engineering* 2011; 41. DOI:10.3329/jme.v41i2.7508.
13. Rodrigues RP, de Pinho AF and Sena DC. Application of hybrid simulation in production scheduling in job shop systems. *SIMULATION* 2020; 96(3): 253–268. DOI:10.1177/0037549719861724.
14. Moussa A. *Approach for energy management in industrial systems: use case: scheduling of automated guided vehicles in job-shop manufacturing system with energy constraints*. PhD Thesis, Oran University 1, 2021.
15. Abderrahim M, Bekrar A, Trentesaux D et al. Manufacturing 4.0 operations scheduling with agv battery management constraints. *Energies* 2020; 13(18). DOI:10.3390/en13184948.
16. Vis IF. Survey of research in the design and control of automated guided vehicle systems. *European Journal of Operational Research* 2006; 170(3): 677–709. DOI:https://doi.org/10.1016/j.ejor.2004.09.020.
17. De Ryck M, Versteijhe M and Debrouwere F. Automated guided vehicle systems, state-of-the-art control algorithms and techniques. *Journal of Manufacturing Systems* 2020; 54: 152–173. DOI:https://doi.org/10.1016/j.jmsy.2019.12.002.
18. Pritsker A and Pegden C. *Introduction to Simulation and SLAM*. A Halsted Press book, Wiley, 1979. ISBN 9780470265888.
19. Ruiz-Martin C. Theory and foundations of modeling and simulation. *SIMULATION* 2023; 99(5): 431–432. DOI:10.1177/00375497231171140.
20. Zambrano Rey G, Bonte T, Prabhu V et al. Reducing myopic behavior in fms control: A semi-heterarchical simulation–optimization approach. *Simulation Modelling Practice and Theory* 2014; 46: 53–75. DOI:https://doi.org/10.1016/j.simpat.2014.01.005. Simulation–Optimization of Complex Systems: Methods and Applications.
21. Dunke F and Nickel S. Day-ahead and online decision-making for collaborative on-site logistics. *Journal of Simulation* 2019; 13(2): 138–151. DOI:10.1080/17477778.2018.1485616.
22. de Mooij J, Bhattacharya P, Dell’Anna D et al. A framework for modeling human behavior in large-scale agent-based epidemic simulations. *SIMULATION* 2023; 99(12): 1183–1211. DOI: 10.1177/00375497231184898.
23. Adam C and Arduin H. Agent-based epidemics simulation to compare and explain screening and vaccination prioritization strategies. *SIMULATION* 2024; 100(4): 335–355. DOI: 10.1177/00375497231178303.


24. Ibrahim Y and Hassan R. A revolutionary crowd model: Implemented to contrast oscillating to consistent media influence on crowd behavior. *SIMULATION* 2017; 93(11): 951–971. DOI:10.1177/0037549717702722.
25. Datsaris G, Vahdati AR and DuBois TC. Agents.jl: a performant and feature-full agent-based modeling software of minimal code complexity. *SIMULATION* 2022; : 00375497211068820 DOI:10.1177/00375497211068820.
26. Collins AJ and Frydenlund E. Strategic group formation in agent-based simulation. *SIMULATION* 2018; 94(3): 179–193. DOI:10.1177/0037549717732408.
27. Yao B, Wang Z, Cao Q et al. Express bus fare optimization based on passenger choice behavior. *SIMULATION* 2016; 92(7): 617–625. DOI:10.1177/0037549715620501.
28. García-Suárez A, Guisado-Lizar JL, Diaz-del Rio F et al. A cellular automata agent-based hybrid simulation tool to analyze the deployment of electric vehicle charging stations. *Sustainability* 2021; 13(10). DOI:10.3390/su13105421.
29. Benhafssa AM, Sahnoun M, Bettayeb B et al. Optimizing energy-conscious dynamic flexible job shop scheduling: Multi-agent simulation approach. In *2021 1st International Conference On Cyber Management And Engineering (CyMaEn)*. pp. 1–6. DOI:10.1109/CyMaEn50288.2021.9497301.
30. Sahnoun M, Xu Y, Abdelaziz FB et al. Optimization of transportation collaborative robots fleet size in flexible manufacturing systems. In *2019 8th International Conference on Modeling Simulation and Applied Optimization (ICMSAO)*. IEEE, pp. 1–5.
31. Sanogo K, Mekhalef Benhafssa A, Sahnoun M et al. Multi-agent simulation for flexible job-shop scheduling problem with traffic-aware routing. In Borangiu T, Trentesaux D, Leitão P et al. (eds.) *Service Oriented, Holonic and Multi-agent Manufacturing Systems for Industry of the Future*. Cham: Springer International Publishing, pp. 573–583.
32. Riaz F and Niazi MA. Enhanced emotion enabled cognitive agent-based rear-end collision avoidance controller for autonomous vehicles. *SIMULATION* 2018; 94(11): 957–977. DOI:10.1177/0037549717742203.
33. Zaidi L, Bettayeb B and Sahnoun M. Optimisation and simulation of transportation tasks in flexible job shop with multi-robot systems. In *2021 1st International Conference On Cyber Management And Engineering (CyMaEn)*. IEEE, pp. 1–6.
34. Bilge Ü and Ulusoy G. A time window approach to simultaneous scheduling of machines and material handling system in an fms. *Operations Research* 1995; 43(6): 1058–1070.
35. Ham A. Transfer-robot task scheduling in job shop. *International Journal of Production Research* 2021; 59(3): 813–823.
36. Malega P, Gazda V and Rudy V. Optimization of production system in plant simulation. *SIMULATION* 2022; 98(4): 295–306. DOI:10.1177/00375497211038908.
37. Heidary MH, Aghaie A and Jalalimanesh A. A simulation–optimization approach for a multi-period, multi-objective supply chain with demand uncertainty and an option contract. *SIMULATION* 2018; 94(7): 649–662. DOI:10.1177/0037549718761588.
38. Xu Y, Sahnoun M, Abdelaziz FB et al. A simulated multi-objective model for flexible job shop transportation scheduling. *Annals of Operations Research* 2020; : 1–22.
39. Ouelhadj D and Petrovic S. A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling* 2009; 12: 417–431.
40. Zhang M, Lu Y, Hu Y et al. Dynamic scheduling method for job-shop manufacturing systems by deep reinforcement learning with proximal policy optimization. *Sustainability* 2022; 14(9). DOI:10.3390/su14095177.
41. Zhang J, Ding G, Zou Y et al. Review of job shop scheduling research and its new perspectives under industry 4.0. *Journal of intelligent manufacturing* 2019; 30: 1809–1830.
42. Meng L, Zhang C, Shao X et al. Milp models for energy-aware flexible job shop scheduling problem. *Journal of Cleaner Production* 2019; 210: 710–723. DOI:https://doi.org/10.1016/j.jclepro.2018.11.021.
43. Dai M, Tang D, Giret A et al. Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints. *Robotics and Computer-Integrated Manufacturing* 2019; 59: 143–157. DOI:https://doi.org/10.1016/j.rcim.2019.04.006.
44. McHANEY R. Modelling battery constraints in discrete event automated guided vehicle simulations. *International Journal of Production Research* 1995; 33(11): 3023–3040. DOI:10.1080/00207549508904859.
45. Boccia M, Masone A, Sterle C et al. The parallel agv scheduling problem with battery constraints: A new formulation and a matheuristic approach. *European Journal of Operational Research* 2023; 307(2): 590–603. DOI:https://doi.org/10.1016/j.ejor.2022.10.023.
46. Boccia M, Mancuso A, Masone A et al. Exact and heuristic solution approaches for the multi-objective agv scheduling problem with battery constraints. *Transportation Research Procedia* 2024; 78: 369–376. DOI:https://doi.org/10.1016/j.trpro.2024.02.047. 25th Euro Working Group on Transportation Meeting.
47. Commission E, for Research DG, Innovation et al. *Industry 5.0 – Towards a sustainable, human-centric and resilient European industry*. Publications Office of the European Union, 2021.

- DOI:doi/10.2777/308407.
48. Maddikunta PKR, Pham QV, B P et al. Industry 5.0: A survey on enabling technologies and potential applications. *Journal of Industrial Information Integration* 2022; 26: 100257. DOI: <https://doi.org/10.1016/j.jii.2021.100257>.
 49. Kader S, M'hammed S and Abdelkader MB. Multi-agent simulation for scheduling and path planning of autonomous intelligent vehicles. In José-Luis GL, Agustín RN, María-José MF et al. (eds.) *Simulation Tools and Techniques*. Cham: Springer Nature Switzerland. ISBN 978-3-031-57523-5, pp. 195–205.
 50. McHaney R. Modelling battery constraints in discrete event automated guided vehicle simulations. *International journal of production research* 1995; 33(11): 3023–3040.
 51. Tisue S and Wilensky U. Netlogo: A simple environment for modeling complexity. In *International conference on complex systems*, volume 21. Citeseer, pp. 16–21.
 52. Railsback SF, Lytinen SL and Jackson SK. Agent-based simulation platforms: Review and development recommendations. *SIMULATION* 2006; 82(9): 609–623. DOI:10.1177/0037549706073695.
 53. Leitão P. Agent-based distributed manufacturing control: A state-of-the-art survey. *Engineering Applications of Artificial Intelligence* 2009; 22(7): 979–991. DOI:10.1016/j.engappai.2008.09.005. Distributed Control of Production Systems.
 54. Hosseini S, Azgomi MA and Torkaman AR. Agent-based simulation of the dynamics of malware propagation in scale-free networks. *SIMULATION* 2016; 92(7): 709–722. DOI: 10.1177/0037549716656060.
 55. Sanz V and Urquia A. Combining pdevs and modelica for describing agent-based models. *SIMULATION* 2023; 99(5): 455–474. DOI:10.1177/00375497221094873.
 56. Zutshi A, Grilo A, Nodehi T et al. Simulation and forecasting of digital pricing models for an e-procurement platform using an agent-based simulation model. *Journal of Simulation* 2018; 12(3): 211–224. DOI:10.1057/s41273-016-0045-6.
 57. Abderrahim M, Bekrar A, Trentesaux D et al. Bi-local search based variable neighborhood search for job-shop scheduling problem with transport constraints. *Optimization Letters* 2020; .
 58. Eloundou J. *Modélisation multi-contraintes d'un système de production flexible*. PhD Thesis, INSA ROUEN, 2016.
 59. Kabir QS and Suzuki Y. Comparative analysis of different routing heuristics for the battery management of automated guided vehicles. *International Journal of Production Research* 2019; 57(2): 624–641. DOI:10.1080/00207543.2018.1475761.
 60. Niestrój R, Rogala T and Skarka W. An energy consumption model for designing an agv energy storage system with a pemf stack. *Energies* 2020; 13(13). DOI:10.3390/en13133435.
 61. Yesilyurt O, Kurrle M, Schlereth A et al. Modelling agv operation simulation with lithium batteries in manufacturing. *Anwendungen und Konzepte der Wirtschaftsinformatik* 2022; 36(16): 7–7.
 62. Skobelev PO and Borovik SY. On the way from industry 4.0 to industry 5.0: From digital manufacturing to digital society. *Industry* 40 2017; 2(6): 307–311.

Author biographies

Kader SANOGO  is currently pursuing a PhD in Computer Science and Industrial Engineering at *École Nationale des Arts et Métiers (ENSAM, Paris, France)*, focusing on Industry 5.0. His research explores optimizing transportation task scheduling for intelligent and collaborative robots using simulation, algorithms, game theory, and multi-agent systems. Before his PhD, he earned a Master's degree in Algorithms and Modeling from the University of Paris-Saclay (France), and another one in Computer security and web technologies from *École Supérieure Africaine des TIC (ESATIC, Abidjan, Côte d'Ivoire)*.

Abdelkader MEKHALEF BENHAFSSA  is an Associate Professor at CESI School of Engineering in Angoulême, France, and is affiliated with the LINEACT research laboratory. His research focuses on sustainable industrial processes, complex system management, and dynamic scheduling in the context of Industry 4.0 and 5.0. He holds a PhD in electrotechnics from the University of Sidi Bel Abbès and the University of Poitiers, and his work has been widely recognized through publications in prestigious journals such as *IEEE Transactions on Industry Applications*, *Journal of Manufacturing Systems*, and *Waste Management*. In addition to his research, Dr. Mekhalef serves as a peer reviewer for several scientific journals, contributing to advancements in intelligent manufacturing, energy optimization, and recycling technologies.

M'hammed SAHNOUN  is a Research Director at CESI LINEACT. His research primarily focuses on the modeling, simulation, and optimization of complex systems, particularly in the areas of manufacturing systems and renewable energy. He holds his Habilitation to Supervise Research (HDR) in Industrial Engineering from Normandy University, a PhD in Automation, and a Master's degree in Organization from the University of Metz, as well as a Master's degree in Robotics and Intelligent Systems from the University of Paris 6. He also earned an Engineering

degree in Automation from the National Polytechnic School of Algiers. Dr. Sahnoun previously worked as a Research Engineer at the Grenoble Institute of Technology and served as a Lecturer at the University of Lorraine. He has supervised several PhD students, with research topics covering industrial data exploitation, complex systems management, and dynamic scheduling of flexible manufacturing systems. Additionally, he manages and coordinates various European and national research projects, including Optiman, CoRoT, AntiHpert, and UV-Robot.

A Simulations results

Instance	2 AGVs								
	VNS	AS-US	DS-US	AS-CAS	DS-CAS	AS-BLS	DS-BLS	AS-CS	DS-CS
EX11	120	119.9 (0.1)	156.5 (5.2)	120.0 (0.1)	152.7 (1.1)	126.3 (6.2)	164.2 (7.7)	128.1 (5.6)	166.6 (8.3)
EX21	122	122.4 (0.1)	147.3 (8.9)	125.6 (0.1)	141.4 (11.9)	135.1 (4.6)	149.7 (7.8)	164.6 (51.3)	152.6 (7.1)
EX31	132	132.0 (0.1)	155.4 (7.6)	132.2 (0.1)	160.9 (8.0)	157.5 (12.5)	166.1 (10.6)	155.1 (6.9)	175.5 (11.4)
EX41	147	147.0 (0.1)	162.7 (12.4)	147.1 (0.1)	155.2 (6.4)	161.2 (1.5)	173.4 (7.6)	175.2 (5.2)	169.7 (8.5)
EX51	116	116.0 (0.1)	140.2 (10.2)	116.2 (0.1)	138.0 (7.2)	131.3 (7.6)	151.2 (11.4)	132.9 (10.9)	153.5 (9.8)
EX61	138	138.0 (0.1)	182.3 (17.7)	142.8 (0.1)	192.5 (10.4)	218.9 (4.0)	193.2 (10.3)	226.4 (15.9)	204.7 (14.6)
EX71	159	159.1 (0.1)	175.2 (11.0)	160.2 (0.1)	173 (7.4)	166.1 (2.3)	190.8 (13.3)	172.2 (8.4)	190.7 (13.0)
EX81	167	167.3 (0.1)	177 (17.3)	171.5 (0.1)	177.1 (15.4)	176.4 (8.6)	182.1 (7.9)	175.1 (4.2)	186.4 (8.3)
EX91	130	130.0 (0.1)	163.5 (17.5)	130.1 (0.1)	162.4 (6.9)	150.7 (14.9)	176.5 (11.1)	164.3 (34.1)	178.6 (10.1)
EX101	169	169.0 (0.1)	200.4 (7.5)	173.5 (0.1)	196.8 (6.8)	181.6 (8.2)	195.8 (9.3)	198.8 (14.9)	200.6 (11.5)
EX12	99	99.2 (0.1)	122.8 (10.5)	99.2 (0.1)	127.6 (10.3)	117.8 (4.8)	121.3 (6.1)	121.6 (8.7)	123.7 (11.0)
EX22	82	82.3 (0.1)	111.9 (8.7)	82.6 (0.1)	114.4 (10.1)	90.7 (11.5)	118.9 (10.6)	90.7 (11.4)	121.2 (8.3)
EX32	95	95.1 (0.1)	127.4 (10)	95.5 (0.1)	127.9 (10.1)	104.0 (1.0)	126.3 (14.7)	105.3 (0.8)	132.7 (9.5)
EX42	109	109.2 (0.1)	137.1 (9.7)	109.6 (0.1)	134.3 (8.6)	124.4 (0.1)	139.9 (12.8)	124.6 (0.1)	144.7 (12.3)
EX52	84	84.2 (0.1)	118 (10.1)	84.2 (0.1)	117.4 (6.6)	98.7 (0.6)	117.8 (10.9)	98.8 (0.6)	118.1 (9.7)
EX62	102	102.4 (0.1)	137.0 (9.2)	102.7 (0.1)	140.3 (13.6)	112.8 (1.6)	140.1 (7.4)	114.6 (0.6)	140.8 (10.9)
EX72	101	101.1 (0.1)	140.7 (11.3)	101.2 (0.1)	143.2 (6.9)	120.7 (0.1)	147.6 (13.5)	120.7 (0.0)	146.9 (16.7)
EX82	155	155.7 (0.1)	167.2 (15.7)	155.7 (0.1)	155.7 (11.9)	156.4 (1.6)	157.5 (17.2)	163.5 (10.0)	166.7 (12.6)
EX92	106	106.5 (0.1)	129.8 (8.9)	106.5 (0.1)	131.1 (12.5)	115.5 (0.1)	135.3 (9.0)	115.9 (0.2)	140.8 (7.5)
EX102	145	145.2 (0.1)	164.5 (8.4)	145.6 (0.1)	161.2 (9.5)	160.6 (1.4)	168.5 (8.5)	162.4 (0.3)	173.9 (12.2)
EX13	99	99.2 (0.1)	148 (10.4)	99.2 (0.1)	147.9 (7.6)	102.6 (8.2)	153 (8.7)	116.2 (8.3)	140.2 (13.3)
EX23	92	92.0 (0.1)	129.2 (5.6)	92.3 (0.1)	128.3 (7.5)	108.1 (0.3)	130.4 (8.4)	116.7 (11.9)	130.8 (5.2)
EX33	104	104.1 (0.1)	138.5 (8.9)	104.2 (0.1)	139.6 (6.8)	120.0 (0.7)	139.1 (13.8)	120.7 (1.6)	139.9 (11.9)
EX43	112	112.0 (0.1)	138.2 (6.7)	112.2 (0.1)	136.0 (6.5)	124.6 (7.2)	142.4 (10.6)	118.2 (8.9)	145.8 (9.5)
EX53	93	93.0 (0.1)	122.2 (6.7)	93.1 (0.1)	126.2 (6.2)	126.8 (10.3)	122.6 (2.7)	128.0 (9.2)	131.0 (8.5)
EX63	110	110.4 (0.1)	126.2 (14.0)	117.8 (0.1)	160.5 (15.5)	115.4 (5.0)	162.7 (11.9)	126.4 (7.0)	167.2 (14.8)
EX73	112	112.1 (0.1)	122.7 (15.8)	112.1 (0.1)	160.2 (19.6)	127.3 (6.0)	165.7 (17.6)	119.4 (5.8)	164.5 (12.1)
EX83	155	155.0 (0.1)	156.6 (12.4)	155.2 (0.1)	156.4 (6.3)	162.0 (0.1)	158.5 (9.5)	183.1 (1.9)	162.3 (13.9)
EX93	109	109.0 (0.1)	139.5 (6.9)	109.2 (0.1)	143.5 (8.2)	119.9 (7.7)	143.0 (6.3)	123.5 (9.7)	144.5 (9.8)
EX103	148	148.0 (0.1)	161.0 (10.3)	148.1 (0.1)	176.6 (6.8)	163.4 (2.0)	166.6 (7.1)	168.0 (8.4)	176.2 (8.2)
EX14	144	144.0 (0.1)	172.8 (7.3)	144.3 (0.2)	187.3 (4.8)	161.4 (2.1)	180.1 (7.5)	164.6 (1.1)	182.4 (8.4)
EX24	156	156.2 (0.1)	192.9 (11.9)	156.3 (0.1)	178.8 (9.3)	179.8 (3.0)	184.4 (11.6)	190.5 (7.0)	190.8 (13.7)
EX34	160	160.2 (0.2)	195.3 (13.3)	162.2 (0.2)	198 (11.9)	176.2 (0.2)	178.6 (11.6)	177 (0.1)	185.8 (13.9)
EX44	180	180.0 (0.1)	201.1 (12.0)	180.1 (0.1)	193.4 (11.9)	206.7 (12.3)	204.9 (11.1)	211.2 (10.1)	214.2 (11.6)
EX54	140	140.0 (0.2)	171 (14.0)	140.2 (0.2)	166.2 (10.4)	185 (9.2)	176.2 (14.1)	185.3 (9.2)	180.6 (13.9)
EX64	168	168.0 (0.1)	210 (15.0)	168.2 (0.1)	203.9 (12.6)	171.4 (9.6)	213.4 (16.7)	175.8 (11.5)	222.2 (18.0)
EX74	191	191.3 (0.1)	225 (16.6)	191.3 (0.1)	213.5 (13.1)	210.9 (0.2)	221.8 (15.3)	210.9 (0.1)	216.7 (9.5)
EX84	190	190.2 (0.3)	204.9 (9.8)	190.3 (0.3)	202.5 (10.6)	200.5 (2.9)	217 (13.5)	203.7 (3.1)	219.7 (10.8)
EX94	158	158.1 (0.1)	187.3 (7.3)	158.1 (0.1)	190.6 (4.3)	173.6 (8.8)	204 (11.0)	185.5 (8.9)	209.7 (10.7)
EX104	202	202.1 (0.2)	211.4 (8.9)	203.3 (0.2)	217.7 (7.9)	218.4 (6.4)	245.3 (17.3)	224.2 (5.6)	240.9 (19.0)

Table A.1. Makespan results of different simulation experiments with 2 AGVs across all the benchmark instances.

Instance	3 AGVs								
	VNS	AS-US	DS-US	AS-CAS	DS-CAS	AS-BLS	DS-BLS	AS-CS	DS-CS
EX11	96	96.2 (0.1)	118.6 (3.9)	108.2 (0.0)	127.4 (26.4)	103.7 (12.7)	124.8 (8.6)	121.5 (8.4)	123.6 (9.3)
EX21	99	99.2 (0.1)	113.2 (6.3)	119.7 (0.0)	115.4 (5.3)	124.2 (7.6)	119.9 (10.9)	120 (6.0)	120.0 (13.3)
EX31	105	105.2 (0.1)	138 (8.3)	110.4 (0.1)	133.2 (9.3)	111.5 (8.2)	133.2 (7.8)	121.4 (7.1)	136.0 (10.3)
EX41	115	115.7 (0.2)	129.6 (6.8)	131.0 (3.0)	134.1 (3.2)	133.6 (11.4)	135.6 (10.5)	142.9 (9.9)	143.3 (8.9)
EX51	87	87.3 (0.1)	109.9 (8.8)	96.2 (0.1)	105.2 (7.9)	104.5 (15.1)	118.5 (18.5)	106.1 (9.1)	116.8 (12.8)
EX61	116	116.1 (0.1)	128.1 (6.2)	124.3 (0.1)	137.2 (6.8)	155.8 (36)	139.4 (18.8)	148.5 (18.4)	142.7 (12.5)
EX71	116	116.5 (0.2)	134.2 (10.6)	123.2 (1.4)	134.7 (13.2)	138.9 (15.4)	136.4 (16.1)	137.8 (14.9)	138.9 (12.3)
EX81	167	167.3 (0.1)	175.8 (8.0)	220.7 (2.8)	182.6 (6.1)	174.3 (7.0)	148.4 (9.1)	200.3 (18.1)	167.1 (11.8)
EX91	115	115.4 (0.1)	134.6 (8.9)	144.9 (0.2)	136.9 (6.9)	129.8 (10.6)	140.4 (9.6)	143.6 (24.9)	151.0 (18.6)
EX101	147	147.7 (0.1)	160.0 (8.0)	176.6 (2.6)	161.8 (6.3)	159.9 (9.8)	165.8 (7.1)	159.8 (4.2)	165.5 (8.7)
EX12	84	84.7 (0.1)	101.2 (8.5)	93.3 (0.0)	106.2 (9.1)	91.6 (3.5)	118.8 (19.8)	99.2 (5.9)	115.3 (7.2)
EX22	81	81.9 (0.1)	98.2 (8.0)	103.6 (0.2)	114.4 (10.3)	90.9 (10.4)	102.0 (8.6)	111.5 (9.5)	115.3 (13.2)
EX32	84	84.9 (0.0)	110.8 (7.9)	89.5 (0.2)	112.3 (5.6)	96.3 (10.3)	116.0 (9.2)	91.8 (0.1)	116.7 (6.7)
EX42	93	93.5 (0.1)	118.9 (10.2)	100.5 (0.1)	120.4 (8.4)	107.2 (14.3)	119.0 (12.6)	105.2 (8.0)	125.0 (10.9)
EX52	72	72.8 (0.1)	92.5 (7.2)	84.7 (2.1)	94.0 (7.1)	79.9 (1.3)	104.4 (10.5)	86.0 (3.5)	100.1 (9.2)
EX62	102	102.7 (0.1)	112.0 (6.0)	115.8 (6.1)	116.9 (6.9)	106.1 (6.1)	117.8 (7.6)	128.9 (6.4)	123.8 (9.0)
EX72	83	83.6 (0.2)	106.4 (13.2)	99.4 (0.1)	102.4 (10.8)	98.0 (0.6)	107.7 (11.6)	109.2 (2.7)	110.6 (13.1)
EX82	155	155.8 (0.1)	165.7 (6.2)	160.4 (4.2)	165.6 (6.7)	161.6 (6.6)	158.2 (6.4)	198.1 (7.4)	168.7 (8.3)
EX92	104	104.8 (0.1)	118.2 (7.0)	105.8 (0.1)	121.1 (7.5)	117.0 (8.1)	122.9 (6.8)	115.6 (2.4)	131.8 (9.4)
EX102	135	135.2 (0.1)	135.8 (9.3)	139.9 (0.0)	140.5 (8.8)	137.8 (2.6)	138.8 (7.1)	146.9 (7.4)	144.4 (10.3)
EX13	90	90.2 (0.1)	113.2 (13.6)	96.4 (3.5)	118.5 (13.0)	95.3 (5.0)	111.9 (9.3)	114.8 (8.3)	119.5 (10.3)
EX23	86	86.3 (0.1)	101.3 (9.5)	89.6 (0.0)	108.3 (10.7)	94.1 (6.3)	111.5 (12.0)	93.1 (9.0)	112.8 (12.0)
EX33	88	88.6 (0.1)	122.3 (8.3)	94.2 (0.0)	118.0 (6.0)	94.5 (8.2)	129.8 (9.5)	103.7 (6.2)	122.0 (10.5)
EX43	92	92.6 (0.1)	112.7 (8.1)	115.2 (0.9)	119.8 (8.5)	104.5 (1.6)	121.4 (11.6)	108.1 (0.1)	121.7 (9.9)
EX53	77	77.0 (0.1)	100.7 (8.0)	98.1 (0.1)	102.2 (10.1)	80.8 (5.2)	109.9 (7.8)	107.4 (7.7)	107.3 (12.2)
EX63	102	102.8 (0.1)	116.0 (6.8)	110.3 (0.1)	121.1 (11.4)	108.7 (6.3)	119.1 (6.5)	117.5 (5.2)	125.6 (10.6)
EX73	91	91.1 (0.1)	118.2 (10.3)	107.4 (0.1)	113.2 (7.0)	104.3 (5.9)	121.9 (7.1)	117.3 (9.2)	132.4 (11.4)
EX83	155	155.4 (0.1)	162.3 (10)	182.4 (0.0)	167.7 (10.5)	172.4 (7.8)	130.0 (10.8)	182.5 (0.1)	176.6 (9.4)
EX93	102	102.6 (0.1)	122.9 (3.6)	116.4 (2.5)	134.9 (8.1)	126.4 (0.8)	127.2 (9.1)	128.0 (9.8)	132.8 (6.6)
EX103	136	136.1 (0.1)	139.5 (8.8)	146.1 (0.0)	153.1 (10.1)	153.1 (0.1)	143.9 (12.6)	161.0 (11.1)	155.4 (11.7)
EX14	110	110.1 (0.3)	148.5 (8.0)	121.6 (0.1)	142.5 (11.9)	118.6 (6.6)	141.0 (9.9)	131.5 (11.2)	144.1 (9.7)
EX24	119	119.6 (0.3)	153.3 (6.9)	125.6 (0.2)	136.9 (9.3)	122.7 (2.8)	142.5 (11.5)	136.2 (10.6)	139.4 (7.0)
EX34	132	132.5 (0.4)	164.1 (9.8)	131.7 (0.1)	158.3 (10.7)	134.6 (5.3)	154.4 (13.1)	143.1 (11.3)	162.2 (8.4)
EX44	136	136.8 (0.3)	163.1 (10.3)	157.0 (0.1)	151.8 (8.6)	151.1 (5.2)	160.0 (9.8)	162.0 (4.6)	154.4 (12.6)
EX54	104	104.4 (0.1)	141.2 (9.6)	104.2 (0.1)	129.1 (9.8)	114.8 (5.8)	134.3 (9.0)	115.3 (5.7)	139.4 (9.6)
EX64	137	137.3 (0.8)	176.7 (13.4)	156.1 (0.1)	154.4 (6.7)	143.6 (8.9)	156 (8.6)	156.1 (0.0)	156.8 (9.3)
EX74	150	150.3 (0.3)	173.5 (10.5)	148.1 (0.2)	167.9 (11.7)	163.3 (1.8)	172.3 (14.3)	172.6 (5.3)	166.4 (11.2)
EX84	177	117.4 (0.1)	181.1 (8.0)	194.7 (0.0)	182.8 (8.6)	182.6 (5.1)	180.2 (5.7)	208.2 (12.6)	188.9 (11.6)
EX94	131	131.6 (0.4)	164.2 (7.0)	138.1 (5.8)	150.4 (6.5)	144.2 (3.9)	150.1 (5.2)	145.3 (6.0)	161.2 (8.8)
EX104	167	167.7 (0.6)	189.2 (7.1)	177.2 (0.1)	193.3 (11.9)	175.4 (6.0)	192.4 (5.7)	187.7 (8.4)	195.4 (14.5)

Table A.2. Makespan results of different simulation experiments with 3 AGVs across all the benchmark instances.

Instance	4 AGVs								
	VNS	AS-US	DS-US	AS-CAS	DS-CAS	AS-BLS	DS-BLS	AS-CS	DS-CS
EX11	88	88.2 (0.2)	113.0 (6.7)	92.2 (0.9)	120.2 (16.7)	104.7 (4.6)	130.4 (13.0)	102.7 (8.3)	123.2 (9.1)
EX21	92	92.6 (0.2)	108.0 (6.8)	128.1 (4.5)	116.7 (4.9)	110.7 (13.0)	115.2 (10)	121.8 (10.9)	123.9 (12.1)
EX31	95	95.6 (0.1)	121.4 (9.4)	108.1 (4.9)	125.4 (19.2)	112.7 (1.2)	134.0 (13.2)	122.5 (8.6)	143.6 (10.5)
EX41	100	100.2 (0.2)	116.9 (8.8)	119.6 (0.1)	119.1 (10.1)	124.2 (10.6)	128.6 (23.6)	131.5 (9.8)	140.8 (27)
EX51	75	75.8 (0.2)	102.4 (10.5)	75.5 (0.0)	106.3 (13.6)	84.4 (11.3)	107.5 (7.7)	99.3 (11.4)	107.6 (7.5)
EX61	114	114.5 (0.2)	129.1 (4.8)	140.7 (0.2)	127.0 (8.9)	122.1 (9.9)	126.5 (7.6)	147.0 (18.1)	134.9 (10.6)
EX71	99	99.1 (0.2)	104.4 (8.0)	129.0 (0.1)	116.3 (14.0)	119.8 (2.5)	118.4 (11.7)	142.9 (13.9)	122.5 (10.8)
EX81	167	167.4 (0.2)	172.9 (9.2)	178.9 (0.0)	175.2 (8.1)	190.0 (8.9)	149.6 (13.5)	204.7 (26.3)	168.6 (9.0)
EX91	112	112.6 (0.2)	122.6 (7.9)	137.3 (0.1)	128.5 (9.7)	130.5 (8.6)	136.6 (16.0)	153.9 (13.1)	147.3 (15.0)
EX101	142	142.1 (0.2)	142.8 (6.6)	159.5 (3.6)	145.2 (7.6)	163.3 (6.2)	144.3 (11.1)	181.4 (25.4)	161.9 (10.3)
EX12	84	84.8 (0.2)	102.1 (6.4)	90.4 (0.1)	115.9 (8.0)	97.5 (12.3)	103.0 (10.2)	98.4 (4.6)	117.0 (7.7)
EX22	81	81.6 (0.2)	95.4 (6.4)	82.4 (0.1)	108.4 (9.4)	91.5 (5.1)	103.7 (6.8)	87.5 (1.3)	110.3 (11.5)
EX32	84	84.9 (0.1)	105.9 (5.9)	100.5 (0.1)	116.9 (7.5)	98.7 (1.1)	117.3 (9.6)	109.5 (7.3)	121.8 (16.9)
EX42	79	79.5 (0.2)	108.5 (5.8)	82.0 (0.0)	119.7 (12.1)	89.8 (2.8)	110.8 (7.1)	92.8 (2.8)	124.5 (12.1)
EX52	72	72.3 (0.2)	89.7 (8.2)	87.8 (0.1)	96.8 (11.3)	85.1 (5.3)	96.1 (13.1)	94.9 (9.8)	90.0 (11.4)
EX62	102	103.0 (0.1)	105.7 (5.9)	115.3 (0.5)	114.7 (7.9)	116.7 (1.5)	102.1 (4.9)	125.7 (4.1)	118.4 (10.1)
EX72	80	80.1 (0.2)	101.5 (9.7)	101.5 (0.1)	106.6 (11.3)	98.7 (6.1)	103.1 (8.0)	104.1 (2.4)	104.7 (12.4)
EX82	155	156.0 (0.2)	165.1 (7.0)	156.8 (0.0)	165.3 (10.4)	163.2 (4.1)	159.5 (7.2)	163.0 (8.6)	158.1 (7.3)
EX92	102	102.3 (0.2)	112.5 (8.7)	129.5 (0.1)	134.5 (17.2)	119.4 (5.2)	113.4 (7.4)	148.0 (13.1)	136.9 (8.0)
EX102	132	132.6 (0.2)	133.7 (11.7)	133.5 (0.1)	146.7 (10)	147.7 (4.5)	137.9 (12.1)	153.8 (3.4)	147.7 (9.0)
EX13	84	84.3 (0.2)	106.6 (7.1)	98.9 (0.0)	117.9 (8.2)	100.1 (4.3)	112.7 (9.5)	110.7 (9.6)	129.1 (15.2)
EX23	84	84.7 (0.2)	97.2 (5.7)	97.9 (0.1)	110.6 (10.8)	92.7 (5.8)	97.2 (8.4)	103.7 (3.4)	109.7 (9.0)
EX33	84	84.9 (0.1)	108.0 (6.2)	84.9 (0.0)	119.7 (15.7)	96.8 (2.8)	112.1 (7.0)	103.4 (1.3)	131.6 (9.5)
EX43	84	86.7 (0.2)	114.5 (8.8)	109.6 (0.0)	129.8 (8.2)	104.2 (1.2)	120.8 (10.8)	129.2 (1.2)	127.3 (10.5)
EX53	73	73.8 (0.2)	95.6 (9.1)	75.6 (0.0)	104.8 (9.4)	87.1 (3.5)	96.4 (6.8)	89 (3.2)	109.5 (12.0)
EX63	102	102.7 (0.2)	110.3 (6.1)	143.1 (2.6)	120.3 (6.8)	138.3 (8.4)	119.3 (10.3)	146.8 (8.4)	127.6 (10.7)
EX73	80	80.2 (0.2)	106.5 (7.3)	82.9 (0.1)	112 (6.6)	97.1 (0.9)	111.7 (6.7)	98.5 (1.6)	115.3 (10.1)
EX83	155	155.6 (0.2)	157.3 (7.0)	159.5 (0.0)	160.2 (11.2)	158.3 (5.0)	162.0 (11.2)	173.1 (4.0)	164.6 (15.4)
EX93	102	102.5 (0.2)	112.7 (6.5)	148.3 (0.0)	141.7 (14.4)	118.4 (2.1)	116.5 (7.1)	152.5 (10.3)	141.8 (14.1)
EX103	135	135.8 (0.2)	139.8 (9.5)	150.0 (0.1)	158.4 (10.7)	145.6 (1.2)	144 (10.6)	168.8 (6.4)	163.4 (11.2)
EX14	100	100.4 (0.2)	137.1 (11.5)	117.7 (0.1)	147.6 (22.1)	123.0 (8.9)	139.3 (9.4)	134.5 (11.5)	144.6 (16.2)
EX24	102	101.9 (0.2)	140.5 (8.5)	109.7 (0.1)	132.9 (11.0)	129.9 (6.8)	129.7 (10.9)	129.5 (14.8)	139.8 (9.9)
EX34	107	106.9 (0.2)	145.5 (6.7)	121.0 (0.1)	143.8 (22.6)	134.4 (10)	152.4 (19.7)	159.0 (12.6)	150.0 (14.5)
EX44	114	114.9 (0.2)	147.8 (10)	127.6 (0.0)	133.7 (9.7)	137.3 (16.4)	143.6 (10.2)	139.2 (17.6)	164.6 (26.5)
EX54	89	89.3 (0.2)	126.8 (9.4)	99.6 (0.1)	118.4 (5.9)	118.2 (12.8)	116.5 (10.8)	122.6 (6.6)	125.5 (10.6)
EX64	119	119.1 (0.2)	155.9 (10.8)	138.9 (5.0)	145.3 (5.0)	159.7 (1.6)	147.4 (7.5)	148.3 (14.7)	157.7 (16.1)
EX74	120	119.9 (0.2)	149.6 (10.6)	129.1 (0.1)	139.9 (10.7)	141.8 (9.7)	138.0 (10)	165.8 (12.4)	141.3 (7.9)
EX84	177	177.3 (0.2)	181.3 (9.9)	204.5 (0.1)	179.1 (10.1)	186.2 (14.5)	172.7 (15.8)	210.5 (3.5)	176.4 (13.4)
EX94	116	116.4 (0.2)	151.7 (7.9)	135.0 (0.1)	154.5 (8.4)	123.9 (16.6)	150.8 (8.8)	141.2 (14.0)	161.9 (13.4)
EX104	152	152.5 (0.2)	173.0 (7.0)	158.0 (4.7)	181.5 (21.6)	174.9 (16.9)	175.5 (14.4)	186.5 (0.1)	186.2 (11.7)

Table A.3. Makespan results of different simulation experiments with 4 AGVs across all the benchmark instances.

Instance	5 AGVs								
	VNS	AS-US	DS-US	AS-CAS	DS-CAS	AS-BLS	DS-BLS	AS-CS	DS-CS
EX11	86	86.6 (0.2)	106 (7.1)	93.6 (0.1)	125.2 (9.4)	108.6 (20.6)	111.6 (6.4)	105.8 (9.9)	136.8 (11.4)
EX21	92	92.9 (0.2)	105.2 (6.1)	103.6 (0.1)	114.6 (9.0)	104.8 (7.7)	112.3 (10.3)	115.7 (7.8)	119.5 (11.6)
EX31	94	94.9 (0.2)	122.7 (12.3)	101.3 (0.1)	141.7 (22)	108.5 (13.5)	138.5 (15.4)	110.8 (13.3)	138.6 (12.3)
EX41	94	94.7 (0.2)	118.4 (10.6)	100.9 (0.1)	141.0 (17.7)	115.2 (5.8)	126.9 (15.1)	125.9 (11.8)	135.4 (11.3)
EX51	75	75.6 (0.3)	90.8 (7.8)	80.0 (0.0)	102.8 (8.1)	85.9 (2.4)	98.4 (12.5)	90.1 (4.4)	103.7 (7.1)
EX61	114	114.7 (0.2)	120.4 (7.5)	119.0 (0.1)	132.5 (10.3)	121.3 (9.6)	128.7 (22.3)	137.2 (8.0)	139.6 (7.5)
EX71	91	91.3 (0.2)	112.2 (7.5)	107.7 (1.1)	112.6 (6.4)	109 (8.0)	110.8 (9.5)	117.0 (9.8)	119.2 (8.2)
EX81	167	167.7 (0.2)	172.1 (8.4)	203.7 (0.0)	17.6 (10.3)	179.8 (8.4)	142.7 (10.4)	208.0 (14.2)	162.3 (8.6)
EX91	111	111.7 (0.2)	124.1 (8.7)	129.1 (4.1)	133.5 (11.7)	135.9 (8.0)	129.4 (19.6)	144.5 (18.9)	168.2 (24.2)
EX101	139	139.7 (0.2)	142.6 (7.5)	147.3 (0.1)	169.0 (20.3)	156.7 (15.6)	150.8 (16.9)	162.5 (17.1)	165.8 (18.9)
EX12	84	84.8 (0.2)	97.5 (9.7)	94.3 (0.1)	127.7 (12.3)	93.8 (6.3)	101.6 (11.9)	97.4 (12.0)	126.0 (13.0)
EX22	81	82.2 (0.2)	91.8 (6.5)	82.8 (0.0)	111.1 (9.0)	91.9 (4.8)	94.0 (6.0)	96.5 (4.0)	117.3 (9.6)
EX32	84	85.1 (0.1)	110.8 (8.9)	96.0 (0.0)	127.0 (12.8)	99.7 (0.8)	114.9 (13.3)	100.6 (2.7)	129.0 (18.3)
EX42	79	79.6 (0.2)	107.2 (9.6)	88.9 (0.1)	130.3 (11.1)	95.2 (0.2)	111.2 (8.0)	102.1 (1.9)	132.5 (14.0)
EX52	72	72.3 (0.2)	81.4 (8.1)	75.2 (0.0)	101.8 (9.3)	78.9 (5.1)	90.5 (10.3)	77.8 (0.4)	104.1 (11.5)
EX62	102	102.1 (0.2)	103.0 (5.3)	121.2 (0.1)	129.5 (9.9)	112.2 (8.0)	106.4 (5.9)	124.8 (4.9)	143.2 (15.8)
EX72	80	80.7 (0.2)	95.2 (9.9)	90.0 (0.1)	103.7 (9.8)	92.8 (1.6)	98.0 (9.8)	100.2 (4.1)	104.0 (7.5)
EX82	155	156.3 (0.2)	160.3 (7.9)	156.1 (0.0)	158.7 (12.1)	159.6 (4.7)	157.5 (8.0)	161.2 (7.1)	157.6 (8.6)
EX92	102	102.6 (0.2)	115.1 (10.5)	120.9 (0.1)	147.7 (19.0)	111.9 (7.9)	112.0 (5.1)	123.0 (8.5)	134.7 (6.1)
EX102	132	132.1 (0.3)	133.6 (7.3)	146.1 (0.1)	161.6 (12.3)	145.4 (6.5)	128.1 (5.2)	166.8 (7.8)	167.2 (15.0)
EX13	84	84.9 (0.2)	99.3 (8.9)	100.7 (0.1)	139.9 (8.4)	96.3 (7.0)	103.2 (9.4)	115.0 (8.1)	137.5 (9.3)
EX23	84	84.2 (0.3)	93.2 (5.5)	111.5 (0.1)	131.4 (14.6)	113.5 (3.6)	96.9 (5.7)	118.6 (11.5)	127.9 (16.1)
EX33	84	84.4 (0.2)	114.4 (10.5)	95.5 (0.0)	144.0 (15.2)	95.6 (5.0)	118.7 (6.9)	112.7 (7.3)	136.4 (11.3)
EX43	79	79.9 (0.2)	110.2 (8.8)	81.6 (0.2)	134.4 (13.5)	95.5 (3.2)	115.3 (9.7)	98.8 (1)	147.9 (13.6)
EX53	73	75.3 (0.2)	84.8 (8.4)	85.3 (0.1)	107.4 (13.2)	85.1 (5.6)	97.2 (11.3)	94.7 (4.3)	108.4 (15.7)
EX63	102	102.6 (0.3)	105.9 (6.3)	118.5 (0.1)	139.7 (12.9)	110.2 (5.3)	114.2 (7.3)	134.3 (8.7)	138.7 (12.8)
EX73	80	80.3 (0.3)	99.6 (7.2)	86.8 (0.1)	108.5 (9.0)	89.9 (8.7)	103.6 (7.3)	90.5 (8.2)	115.1 (9.2)
EX83	155	156.2 (0.2)	158.5 (8.2)	163.7 (0.1)	162.9 (13.6)	164.8 (8.6)	159.7 (10.5)	163.7 (9.8)	162.5 (12.2)
EX93	102	102.6 (0.2)	113.0 (6.7)	127.5 (0.1)	131.3 (7.1)	102.2 (0.1)	114.5 (8.5)	138.1 (23.6)	121.3 (10.6)
EX103	135	135.3 (0.2)	137.7 (7.4)	160.5 (0.1)	157.5 (14.0)	149.2 (7.9)	131.8 (8.7)	164.7 (4.8)	183.4 (15.8)
EX14	96	96.0 (0.2)	135.0 (7.9)	113.6 (0.1)	144.1 (15.5)	113.5 (12.4)	131.8 (11.9)	132.8 (9.3)	171.6 (27.9)
EX24	98	98.1 (0.2)	127.2 (9.0)	135.3 (0.1)	127.5 (9.9)	132.2 (7.7)	127.1 (10.2)	133.3 (7.7)	147.2 (19.9)
EX34	102	102.1 (0.2)	159.7 (13.0)	121.0 (2.4)	151.9 (17.8)	108.6 (7.2)	141.1 (15.9)	144.2 (15.8)	156.2 (15.0)
EX44	105	105.4 (0.2)	125.8 (9.9)	118.1 (0.0)	162.3 (18.7)	131.8 (8.4)	139.5 (13.6)	143.8 (17.0)	170.8 (13.3)
EX54	82	82.1 (0.2)	118.1 (10.5)	86.2 (0.1)	122.3 (12.8)	116.1 (6.2)	117.5 (14.9)	116.4 (12.0)	136.5 (25.7)
EX64	118	118.5 (0.2)	125.2 (8.0)	134.3 (0.1)	149.3 (13.0)	135.9 (12.1)	140.8 (9.7)	152.4 (19.3)	159.5 (12.9)
EX74	103	108.0 (0.2)	128.2 (10)	119.8 (0.1)	131.6 (11.0)	133.5 (8.2)	129.7 (11.8)	141.5 (7.1)	145.6 (25.2)
EX84	177	177.9 (0.2)	181.4 (10.2)	184.8 (0.0)	180.4 (13.2)	206.1 (15.2)	174.4 (15.2)	194.7 (1.2)	177.2 (14.1)
EX94	116	116.1 (0.2)	153.9 (7.8)	139.6 (5.4)	186.1 (25)	158.1 (0.2)	142.7 (13.5)	162.0 (16.2)	149.4 (12.0)
EX104	152	152.6 (0.2)	168.2 (9.4)	164.8 (0.0)	168.9 (7.5)	178.2 (12.4)	172.3 (18.9)	190.4 (16.7)	192.4 (21.5)

Table A.4. Makespan results of different simulation experiments with 5 AGVs across all the benchmark instances.

Instance	2 AIVs		3 AIVs		4 AIVs		5 AIVs	
	AS-CAS	DS-CS	AS-CAS	DS-CS	AS-CAS	DS-CS	AS-CAS	DS-CS
EX11	120	118.5 (0.5)	131.1 (5.1)	127.1 (9.6)	147.4 (11.5)	147.4 (11.5)	103.2 (20.9)	109.0 (11.2)
EX21	122	119.1 (0.2)	135.1 (8.5)	138.8 (26.8)	140.8 (17.6)	140.8 (17.6)	111.2 (12.5)	107.1 (12.7)
EX31	132	122.2 (0.3)	135.1 (4.7)	136.7 (3.5)	148.1 (11.9)	148.1 (11.9)	109.8 (10.3)	125.9 (18.0)
EX41	147	140.1 (0.2)	140.4 (9.9)	152.4 (6.1)	154.2 (12.8)	154.2 (12.8)	107.6 (7.8)	121.4 (13.8)
EX51	116	103.1 (0.1)	124.8 (14.4)	123.4 (8.2)	136.1 (14.3)	136.1 (14.3)	91.5 (17.4)	98.4 (26.3)
EX61	138	135.6 (0.4)	147.9 (9.8)	215.8 (8.6)	170.8 (20.0)	170.8 (20.0)	131.9 (19.6)	120.9 (14.9)
EX71	159	144.8 (0.3)	161.0 (15.9)	173.3 (8.1)	171.1 (15.3)	171.1 (15.3)	99.9 (8.3)	101.9 (12.1)
EX81	167	154.8 (0.9)	155.2 (7.1)	179.9 (16.4)	172.7 (10.6)	172.7 (10.6)	180.0 (7.6)	179.9 (17.1)
EX91	130	123.5 (0.2)	145.7 (11.8)	140.3 (15.1)	154.6 (14.0)	154.6 (14.0)	129.4 (11.5)	127.2 (17.5)
EX101	169	157.9 (0.3)	132.3 (8.5)	178.9 (9.8)	187.1 (10.4)	187.1 (10.4)	152.6 (13.1)	140.2 (13.9)
EX12	99	76.5 (0.1)	106.0 (4.4)	98.4 (5.8)	113.5 (8.3)	113.5 (8.3)	78.1 (7.7)	84.7 (10.1)
EX22	82	82.8 (0.1)	97.8 (6.9)	97.7 (1.7)	103.7 (6.6)	103.7 (6.6)	91.3 (4.1)	83.1 (7.4)
EX32	95	87.7 (0.1)	103.4 (10.6)	99.1 (6.8)	109.3 (7.7)	109.3 (7.7)	92.1 (5.4)	91.3 (8.4)
EX42	109	94.8 (0.1)	97.0 (8.2)	105.6 (5.8)	107.5 (6.3)	107.5 (6.3)	80.5 (7.0)	86.2 (7.2)
EX52	84	68.5 (0.1)	88.3 (3.5)	80.6 (3.4)	97.4 (10.6)	97.4 (10.6)	73.9 (5.9)	73.8 (6.1)
EX62	102	102.8 (0.1)	107.1 (5.9)	111.7 (1.1)	111.8 (10.6)	111.8 (10.6)	66.1 (4.9)	66.1 (4.9)
EX72	101	96.3 (0.1)	108.5 (8.7)	109.5 (0.6)	110.6 (9.2)	110.6 (9.2)	109.7 (5.7)	91.4 (5.2)
EX82	155	155.6 (0.1)	166.4 (7.0)	155.8 (0.0)	154.9 (6.2)	154.9 (6.2)	91.2 (7.3)	82.1 (7.9)
EX92	106	99.5 (0.1)	103.6 (3.9)	99.7 (0.1)	112.3 (7.1)	112.3 (7.1)	156.0 (0.0)	155.6 (7.2)
EX102	145	143.6 (0.1)	132.3 (8.5)	143.5 (0.0)	140.3 (8.5)	140.3 (8.5)	106.8 (7.6)	95.8 (8.1)
EX13	99	80.7 (0.0)	89.2 (7.8)	91.1 (2.0)	92.7 (6.5)	92.7 (6.5)	141.3 (7.8)	121.7 (7.6)
EX23	92	84.4 (0.0)	83.1 (4.0)	96.0 (2.8)	91.4 (3.8)	91.4 (3.8)	81.8 (3.4)	87.0 (6.5)
EX33	104	89.3 (0.1)	97.4 (5.9)	102.4 (2.7)	104.2 (11.6)	104.2 (11.6)	93.6 (6.1)	81.8 (7.4)
EX43	112	87.8 (0.1)	88.4 (5.3)	105.0 (1.3)	95.2 (8.2)	95.2 (8.2)	91.6 (4.9)	95.1 (8.0)
EX53	93	90.3 (0.0)	80.3 (6.4)	96.5 (7.3)	86.2 (10.7)	86.2 (10.7)	87.5 (5.6)	92.1 (10.4)
EX63	110	100.9 (0.0)	94.5 (7.4)	106.8 (6.9)	98.7 (5.6)	98.7 (5.6)	70.2 (5.8)	70.1 (7.2)
EX73	112	88.9 (0.1)	104.6 (9.4)	100.9 (1.7)	102.1 (6.7)	102.1 (6.7)	109.8 (7.4)	89.2 (5.2)
EX83	155	155.5 (0.0)	155.3 (6.9)	156.5 (1.0)	155.2 (6.6)	155.2 (6.6)	84.6 (7.0)	78.0 (5.1)
EX93	109	103.0 (0.1)	103.8 (5.7)	110.4 (5.2)	109.9 (7.8)	109.9 (7.8)	158.0 (3.1)	154.1 (6.6)
EX103	148	137.9 (0.0)	140.6 (4.1)	139.9 (4.4)	136.4 (8.3)	136.4 (8.3)	97.7 (0.0)	93.0 (3.8)
EX14	144	128.7 (0.1)	146.0 (9.1)	139.4 (5.6)	153.8 (7.2)	153.8 (7.2)	140.1 (6.0)	121.8 (10.4)
EX24	156	138.2 (0.1)	134.5 (10.1)	147.0 (6.7)	144.6 (11.9)	144.6 (11.9)	95.9 (8.3)	107.5 (8.9)
EX34	160	151.6 (0.3)	149.6 (11.8)	159.9 (0.0)	152.4 (10.2)	152.4 (10.2)	108.9 (9.9)	105.9 (11.2)
EX44	180	149.3 (0.1)	150.3 (14.1)	160.7 (9.7)	165.2 (12.8)	165.2 (12.8)	110.6 (10.8)	123.2 (14.0)
EX54	140	112.6 (0.1)	132.9 (12.6)	158.9 (5.0)	139.5 (14.1)	139.5 (14.1)	110.6 (2.8)	123.2 (12.3)
EX64	168	154.4 (0.1)	168.3 (13.0)	174.0 (11.9)	176.8 (11.9)	176.8 (11.9)	88.7 (3.4)	94.5 (10.7)
EX74	191	169.6 (0.1)	176.8 (10.8)	184.4 (5.1)	189.2 (11.9)	189.2 (11.9)	124.9 (8.1)	117.8 (8.8)
EX84	190	166.5 (0.3)	170.3 (11.4)	196.3 (1.1)	175.2 (6.8)	175.2 (6.8)	117.3 (5.8)	115.7 (10.0)
EX94	158	148.4 (0.1)	156.3 (6.1)	154.9 (0.1)	167.5 (10.9)	167.5 (10.9)	189.7 (7.2)	155.2 (12.5)
EX104	202	197.2 (0.1)	195.7 (9.1)	202.4 (2.8)	191.4 (9.1)	191.4 (9.1)	134.0 (8.1)	125.8 (9.1)
							165.2 (7.6)	143.9 (9.0)

Table A.5. Makespan results of different simulation experiments with AIVs across all the benchmark instances.

B Job sets proposed by Bilge and Ulusoy

Job set 1

Job 1: M1(8); M2(16); M4(12)
 Job 2: M1(20); M3(10); M2(18)
 Job 3: M3(12); M4(8); M1(15)
 Job 4: M4(14); M2(18)
 Job 5: M3(10); M1(15)

Job set 2

Job 1: M1(10); M4(18)
 Job 2: M2(10); M4(18)
 Job 3: M1(10); M3(20)
 Job 4: M2(10); M3(15); M4(12)
 Job 5: M1(10); M2(15); M4(12)
 Job 6: M1(10); M2(15); M3(12)

Job set 3

Job 1: M1(16); M3(15)
 Job 2: M2(18); M4(15)
 Job 3: M1(20); M2(10)
 Job 4: M3(15); M4(10)
 Job 5: M1(8); M2(10); M3(15); M4(17)
 Job 6: M2(10); M3(15); M4(8); M1(15)

Job set 4

Job 1: M4(11); M1(10); M2(7)
 Job 2: M3(12); M2(10); M4(8)
 Job 3: M2(7); M3(10); M1(9); M3(8)
 Job 4: M2(7); M4(8); M1(12); M2(6)
 Job 5: M1(9); M2(7); M4(8); M2(10); M3(8)

Job set 5

Job 1: M1(6); M2(12); M4(9)
 Job 2: M1(18); M3(6); M2(15)
 Job 3: M3(9); M4(3); M1(12)
 Job 4: M4(6); M2(15)
 Job 5: M3(3); M1(9)

Job set 6

Job 1: M1(9); M2(11); M4(7)
 Job 2: M1(19); M2(20); M4(13)
 Job 3: M2(14); M3(20); M4(9)
 Job 4: M2(14); M3(20); M4(9)
 Job 5: M1(11); M3(16); M4(8)
 Job 6: M1(10); M3(12); M4(10)

Job set 7

Job 1: M1(6); M4(6)
 Job 2: M2(11); M4(9)
 Job 3: M2(9); M4(7)
 Job 4: M3(16); M4(7)
 Job 5: M1(9); M3(18)
 Job 6: M2(13); M3(19); M4(6)
 Job 7: M1(10); M2(9); M3(13)
 Job 8: M1(11); M2(9); M4(8)

Job set 8

Job 1: M2(12); M3(21); M4(11)
 Job 2: M2(12); M3(21); M4(11)
 Job 3: M2(12); M3(21); M4(11)
 Job 4: M2(12); M3(21); M4(11)
 Job 5: M1(10); M2(14); M3(18); M4(9)
 Job 6: M1(10); M2(14); M3(18); M4(9)

Job set 9

Job 1: M3(9); M1(12); M2(9); M4(6)
 Job 2: M3(16); M2(11); M4(9)
 Job 3: M1(21); M2(18); M4(7)
 Job 4: M2(20); M3(22); M4(11)
 Job 5: M3(14); M1(16); M2(13); M4(9)

Job set 10

Job 1: M1(11); M3(19); M2(16); M4(13)
 Job 2: M2(21); M3(16); M4(14)
 Job 3: M3(8); M2(10); M1(14); M4(9)
 Job 4: M2(13); M3(20); M4(10)
 Job 5: M1(9); M3(16); M4(18)
 Job 6: M2(19); M1(21); M3(11); M4(15)

C Original workshop layouts

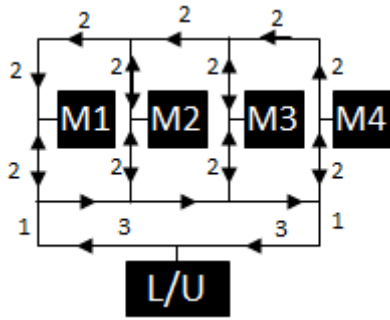


Figure C.1. Layout 1

	L/U	M1	M2	M3	M4
L/U	0	6	8	10	12
M1	12	0	6	8	10
M2	10	6	0	6	8
M3	8	8	6	0	6
M4	6	10	8	6	0

Table C.1. Travel times on layout 1

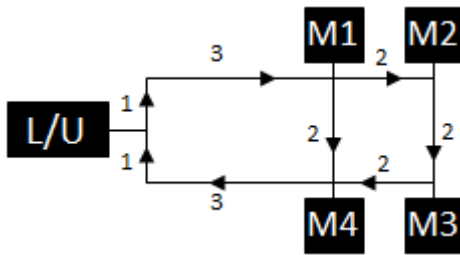


Figure C.2. Layout 2

	L/U	M1	M2	M3	M4
L/U	0	4	6	8	6
M1	6	0	2	4	2
M2	8	12	0	2	4
M3	6	10	12	0	2
M4	4	8	10	12	0

Table C.2. Travel times on layout 2

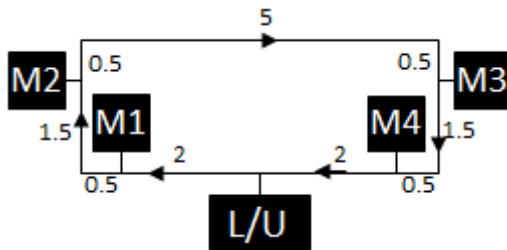


Figure C.3. Layout 3

	L/U	M1	M2	M3	M4
L/U	0	2	4	10	12
M1	12	0	2	8	10
M2	10	12	0	6	8
M3	4	6	8	0	2
M4	2	4	6	12	0

Table C.3. Travel times on layout 3

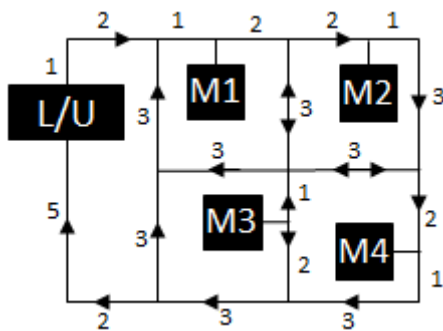


Figure C.4. Layout 4

	L/U	M1	M2	M3	M4
L/U	0	4	8	10	14
M1	18	0	4	6	10
M2	20	14	0	8	6
M3	12	8	6	0	6
M4	14	14	12	6	0

Table C.4. Travel times on layout 4