mathematics

Article

Simulation of Heuristics for Automated Guided Vehicle Task Sequencing with Resource Sharing and Dynamic Queues

Jonas F. Leon [1,2], Mohammad Peyman [3], Xabier A. Martin [3] and Angel A. Juan [3,*]

[1] Department of Computer Science, Multimedia and Telecommunication, Universitat Oberta de Catalunya, 08018 Barcelona, Spain; jofule@uoc.edu
[2] Spindox España S.L., 08021 Barcelona, Spain
[3] Research Center on Production Management and Engineering, Universitat Politècnica de València, 03801 Alcoy, Spain; mpeyman@upv.es (M.P.); xamarsol@upv.es (X.A.M.)
[*] Correspondence: ajuanp@upv.es

Abstract: Automated guided vehicles (AGVs) stand out as a paradigmatic application of Industry 4.0, requiring the seamless integration of new concepts and technologies to enhance productivity while reducing labor costs, energy consumption, and emissions. In this context, specific industrial use cases can present a significant technological and scientific challenge. This study was inspired by a real industrial application for which the existing AGV literature did not contain an already well-studied solution. The problem is related to the sequencing of assigned tasks, where the queue formation dynamics and the resource sharing define the scheduling. The combinatorial nature of the problem requires the use of advanced mathematical tools such as heuristics, simulations, or a combination of both. A heuristic procedure was developed that generates candidate task sequences, which are, in turn, evaluated in a discrete-event simulation model developed in Simul8. This combined approach allows high-quality solutions to be generated and realistically evaluated, even graphically, by stakeholders and decision makers. A number of computational experiments were developed to validate the proposed method, which opens up some future lines of research, especially when considering stochastic settings.

Keywords: automated guided vehicles; heuristics; discrete-event simulation

MSC: 90-10; 90C59; 90B06

check for updates

Citation: Leon, J.F.; Peyman, M.; Martin, X.A.; Juan, A.A. Simulation of Heuristics for Automated Guided Vehicle Task Sequencing with Resource Sharing and Dynamic Queues. *Mathematics* **2024**, *12*, 271. https://doi.org/10.3390/math12020271

Academic Editors: Zsolt Tibor Kosztyán, Zoltán Kovács and Ioannis G. Tsoulos

Received: 20 November 2023
Revised: 11 January 2024
Accepted: 12 January 2024
Published: 14 January 2024

1. Introduction

Industry 4.0 presents great opportunities for improving productivity, but it also comes with plenty of technological challenges. These challenges mostly relate to the transformation of human-supervised processes into automated and machine-controlled processes, which, in turn, lead to distributed complex system coordination and connectivity issues [1]. In this context, the use of AGVs is becoming increasingly important, mainly due to the potential productivity enhancement and reduction in labor cost, energy consumption, and emissions [2]. Furthermore, AGVs are paradigmatic since their implementation requires the symbiotic integration of many of the Industry 4.0 key concepts and technologies, such as adaptive robotics, embedded systems, communication and networking, cloud systems, simulation, virtualization, data analytics, and artificial intelligence [3].

A great deal of research has been devoted to AGVs since they were initially introduced more than 25 years ago [4]. However, there are still open areas of development and specific use cases that pose a technological and scientific challenge. The case presented in this study was inspired by a real industrial application that was difficult to model following existing AGV problems found in the literature. The problem revolves around the dispatching, routing, and scheduling of tasks for multiple AGVs. However, these terms are not strictly equivalent [5]. In Vivaldini et al. [6] a possible distinction between them has been established: (i) dispatching is the process of selecting and assigning tasks to vehicles; (ii) routing is the selection of specific paths that each vehicle needs to execute in order to fulfill the assigned tasks; and (iii) scheduling is the definition of the arrival and departure times of the vehicles along the assigned routes for each one of its assigned tasks. Following these definitions, solving the scheduling problem assumes that the routing and dispatching problems have already been taken care of. Furthermore, in the present work, the scheduling is considered to be a consequence of the sequence in which the different tasks are carried out by the resources available at each point in time. This means that the queue formation dynamics and resource sharing are the key elements defining the schedule if handled by a sufficiently representative model. The problem presented in this form could be referred to as task sequencing, i.e., defining the order in which the (already assigned and planned) tasks are executed. The problem is trivial if the number of resources is infinite and the interaction between resources is neglected, but it becomes a true combinatorial challenge when the resources are constrained and the queue dynamics are accounted for. Hence, the coordination of multiple AGVs that are performing a given set of tasks, and that are considering also the interaction with other machines and the existence of resource sharing constraints, is a complex problem that can be modeled in many different ways. For instance, it could be modeled as a robotic task sequencing problem (RTSP) or as a resource-constrained scheduling problem (RCSP). Both the RTSP [7] and the RCSP [8] have been proven to be NP-hard. Thus, it is expected that any version that can be considered a particular case or combination of both problems is also NP-hard in nature. In other words, as the number of AGVs and workstations increases, capturing the interactions between the individual AGVs and workstations becomes nearly impossible using conventional mathematical methods, and it becomes necessary to resort to more advanced mathematical tools such as heuristics or simulation (or a combination of them).

In that sense, the use of algorithms and heuristics to find high-quality solutions in a reasonable time can be considered the preferred option to solve these types of problems, as opposed to exact procedures that aim for an optimal solution. The time it would take to find the optimal solution to a problem with such a vast solution space and the gap between the optimal solution found and the real optimal solution to the problem (e.g., considering the randomness present in the real system) makes heuristics a very sensible approach [9]. The approach followed in this study consists of identifying several algorithms typically used for sorting non-uniform tasks and resources, and then combining them in a heuristic procedure that generates candidate solutions for the task sequencing problem. Furthermore, to increase the fidelity of the proposed model, a discrete-event simulation model was developed in the Simul8 commercial software to evaluate the performance of the heuristic results. The use of simulators such as Simul8 can have many advantages in this context, such as the following: (i) aiding the graphical validation of results from stakeholders, especially technical personnel and decision makers; and (ii) facilitating the creation of complex and detailed models of real-life scenarios [10]. Without the simulation component, the evaluation of the candidate solutions obtained by the heuristic procedure is not possible due to the dynamic nature of the queues and shared resource availability (temporal or sequential dependencies).

There are many studies in the literature that, as reviewed in Section 2, approach scheduling or sequencing problems by disregarding or simplifying the complex interaction between the available resources, as well as by considering that the idealized solutions found can be executed in a realistic environment. The situation is even worse when the environment is stochastic, as is the case in all real-life industrial applications. Therefore, there is a gap in terms of scalable methodologies that can cope with those complex interactions and can be extended to more complicated settings where random variability is considered. Therefore, the main contribution of this paper is the development of a methodology that combines simulation and the use of simple heuristics to find high-quality solutions when the sharing of limited resources causes queuing and interactions. The effectiveness of the proposed methodology is proven, through various numerical experiments, to be more efficient than the NEH-based algorithm that uses the state-of-the-art NEH heuristic.

The rest of the paper is structured as follows: Section 2 reviews the relevant literature related to the problem studied. Afterward, Section 3 defines the problem precisely but also includes an example to clarify the concepts. The solving methodology is described in detail in Section 4, and the results are presented in Section 5 together with a brief discussion of the obtained results. Finally, Section 6 highlights the most important conclusions of the study and some future lines of research.

2. Literature Review

In this section, a brief literature review is conducted to present the current developments in the scientific community aimed at solving AGV task sequencing problems using heuristic and simulation methods.

2.1. AGVs in Industry 4.0

As mentioned in Section 1, the use of AGVs is increasingly regarded as an essential element in realizing the productivity improvements promised by the Industry 4.0 framework. The most basic challenge facing AGV technology is the efficient path planning for navigating manufacturing landscapes. Fransen and van Eekelen [11] discussed AGV control strategies and illustrated the use of the A* algorithm on a geometric graph to aid proficient path planning, thus influencing the flow layout within facilities. When multiple AGVs are used, traffic management strategies must be implemented to prevent collisions and ensure the smooth operation of AGVs. In this regard, Pratissoli et al. [12] proposed novel methods for coordinating a fleet of AGVs in an industrial setting, where they focused on developing complete traffic manager software based on a multi-level control architecture. Furthermore, the emergence of 5G connectivity has been identified as a technological trigger for AGV implementation, enhancing their collaborative capabilities via high-speed, low-latency networks. Vlachos et al. [13] used a case study to investigate the implications of AGVs integrated with the IoT in flexible manufacturing systems, depicting the technological synergy between AGVs and the IoT within a manufacturing enterprise. Also, Reis et al. [14] proposed a systematic literature review to identify the most relevant methodologies and technologies related to AGV position control, a critical aspect of manufacturing intralogistics and material handling, thereby aiding in defining optimal vehicle requirements and enhancing operational efficiency. For an overview of the current challenges and technological advances in AGVs, readers can refer to Oyekanlu et al. [15]. Conversely, for a survey on the design and control of AGV systems (flow layout, traffic management, vehicle requirements, etc.), readers can refer to Vis [4]. In the context of automated container terminals, Kim and Bae [16] introduced a dispatching method for AGVs to work with container cranes, enhancing efficiency by assigning future tasks and using simulation in various environments. Li et al. [17] tackled task assignment and sequencing for AGVs in manufacturing using a harmony search algorithm, which was tested in real-life scenarios. Zou et al. [18] employed a bee colony algorithm for a similar problem in a Chinese manufacturing company, focusing on a single depot like in our study.

2.2. Task Sequencing: Conceptual Clarifications and Evolution

Despite the extensive and often ambiguous terminology in its domain, scheduling, which is closely related to task sequencing, provides valuable insights for this work. As previously stated, terms such as dispatching, task scheduling, sequencing, and allocation are frequently used interchangeably in the literature [19]. Within the context of this study, it is crucial to underscore how shared resource availability influences task sequencing and scheduling. Since its introduction several decades ago [20], the RCSP has evolved into a well-studied optimization problem [21]. This problem involves scheduling a set of tasks or activities by assigning them specific execution times, aiming to minimize the total project duration or makespan while adhering to various constraints. These constraints are generally categorized based on the following: (i) the available resources required for completing different activities; and/or (ii) activity sequences (often referred to as precedence constraints) [22]. Due to its practical importance, the RCSP has been the focus of numerous studies. For instance, Hartman and Briskorn [23] surveyed RCSP and its extensions, highlighting the significance of efficient resource allocation for optimal scheduling solutions. Recent advances in computational techniques have encouraged the development of novel approaches to tackle the challenges posed by the RCSP. Notably, van der Beek et al. [24] introduced a hybrid differential evolution algorithm specifically tailored for the RCSP. This algorithm considers a flexible project structure along with the dynamics of resource consumption and production, showcasing the potential of evolutionary algorithms in navigating the complex challenges of resource allocation and task sequencing.

Equally pertinent to this study is reviewing the literature for the RTSP. The RTSP has undergone extensive study as it aims to optimize the sequence of tasks performed by robots in diverse industrial settings. Foundational work by Maimon [25] laid the groundwork, addressing the efficient utilization of robot task flexibility characteristics and providing a framework and classification for such problems. Over time, a plethora of approaches and methodologies have been proposed to address the challenges posed by the RTSP. For example, Suárez-Ruiz et al. [26] made a significant contribution by devising a method to optimize the sequence in which a robot visits multiple targets in industrial applications, resulting in fast RTSP solutions, which are especially beneficial in time-sensitive operations. Subsequently, Li et al. [27] introduced an efficient approach employing a decoupling strategy to determine an optimal sequence of collision-free motions for robots executing a set of repetitive tasks, thus emphasizing the continuous efforts to enhance efficiency and safety in robotic operations, particularly in environments where collision avoidance is critical. Furthermore, Chen et al. [28] investigated optimizing joint-space tour scheduling to align with manufacturing criteria for each task point visit, highlighting the significance of planning the sequence of robotic tasks in attaining operational excellence in industrial settings. Some authors have translated RTSP concepts to AGV applications, focusing on a single AGV unit. For instance, Li et al. [29] converted the RTSP into a traveling salesman problem, a common practice in robotics. They proposed a heuristic to reduce its complexity from $O(N^3)$ to $O(N^2)$ by utilizing auxiliary data structures. In another approach, Martin et al. [30] proposed a reactive solution for a dynamic RTSP, where the AGV resets after each task, akin to this study's application. They focused on task sequencing, using simulation to manage resource constraints, workstation availability, and queue dynamics, thus leading to effective scheduling.

Another significant problem family that extensively explores optimizing task sequences executed by robots is the multi-robot task allocation (MRTA) problem. Korsah et al. [31] presented a comprehensive taxonomy for MRTA, establishing a framework that aids in the efficient utilization of robot task flexibility characteristics. This taxonomy assists in understanding the features and complexity of MRTA problems, which have been extensively investigated in several studies. For example, Khamis et al. [32] studied multi-robot systems executing collective behaviors for tackling complex tasks. Similarly, Alshaboti and Baroudi [33] discussed state-of-the-art algorithms and strategies employed in tackling MRTA problems, providing insights into the evolving landscape of robot task allocation. Additionally, Chakraa et al. [34] reviewed optimization techniques for multi-robot task allocation problems, showcasing current methods and offering insights into future directions for optimizing task allocation among multi-robots.

2.3. Application of Discrete-Event Simulation

Most of the articles mentioned above employ simulation to varying extents to validate algorithms developed for their AGV-related problems. Simulation facilitates a better understanding of reality's complexity, reducing the necessity for the oversimplifications and unrealistic assumptions sometimes required in analytical models [35]. Discrete-event simulation, in particular, has proven to be a valuable tool for modeling AGV-based transport systems, such as AGV traffic management policies, or, as in the current study, AGV task dispatching/sequencing rules [36]. For example, in Inoue [37] a discrete-event simulation model was established to simulate AGV behavior and to identify the best control policy in a real factory use case. Similarly, bin Md Fauadi et al. [38] proposed a discrete-event simulation approach to determine vehicle requirements in a manufacturing environment. Notably, the authors emphasized simulation's role in accurately modeling queuing behavior when multiple AGVs interact. Also, Kühn et al. [39] introduced a two-stage genetic algorithm for generating sequencing heuristics in stochastic, decentralized, multi-project scheduling with limited resources. The algorithm emphasizes efficient heuristic generation with lower simulation efforts and uses deterministic values initially for robust solutions, followed by a more intensive computation stage incorporating stochastic values. The PyScOp framework facilitates this simulation and optimization process. Equally, Azimi and Sholekar [40] presented a novel approach to solving the multi-objective multi-mode resource-constrained project scheduling problem with stochastic duration. This approach employs a simulation-based optimization method, SIMSUM1, which integrates discrete-event simulation with a relaxation technique for binary decision variables. Given the advantages that discrete-event simulation offers in modeling multi-AGV scenarios, the use of commercial simulators can expedite model creation, ensure its validity, and simplify result communication to stakeholders.

Since the majority of the reviewed papers did not address the specific use of simulation software in our problem context, namely the complex interaction between shared resources, the purpose of this study is to fill this gap in the literature. To provide a novel approach in this area, we combined a heuristic algorithm with discrete-event simulation software. This combination is intended to improve problem-solving effectiveness in our particular problem scenario. While previous studies have utilized commercial simulation software like ARENA [41] or FlexSim [42] for warehouse AGV applications, Simul8 was selected for this research due to its capabilities and alignment with the study's objectives. Simul8 (www.simul8.com, accessed on 22 December 2023) is a commercial simulator renowned for its flexibility in modeling complex systems, where its focus is on being intuitive, fast, and effective. Although Simul8 has been successfully employed in other applications such as healthcare systems [43], supply chain optimization [44], or transportation [45], to the best of the authors' knowledge, it has not been previously utilized for AGV sequencing optimization.

3. Problem Description

In this section, the industrial case that inspired this study is presented. Subsequently, an illustrative example of the problem is provided to enhance comprehension of the application under consideration. Finally, the problem is formulated in mathematical terms.

3.1. Motivation and Industrial Context

The problem delineated in this study draws inspiration from a genuine industrial scenario at a manufacturing company in Spain. The company operates a production facility equipped with several AGVs responsible for collecting and transporting products between designated plant locations. Initially, the products awaiting processing are stored temporarily in a specific warehouse location, known as the depot. Then, these products are transported to various workstations, each dedicated to a distinct manufacturing process. Following the transportation of the product, the AGV stationed there travels back to the starting point to transport the next product. The number of AGVs available at the starting point remains fixed and are shared among the different streams of work. Visiting a workstation involves a travel time (to and from) and a processing time. Notice that each workstation has the capacity to process only one product at a time. Consequently, if an AGV intends to visit an already occupied workstation, it must wait in the queue until the station becomes available. Figure 1 provides a schematic representation of the aforementioned industrial scenario. The primary objective of the company is to establish the sequence for the execution of various tasks by the AGVs to minimize the total production time, commonly referred to as the makespan.

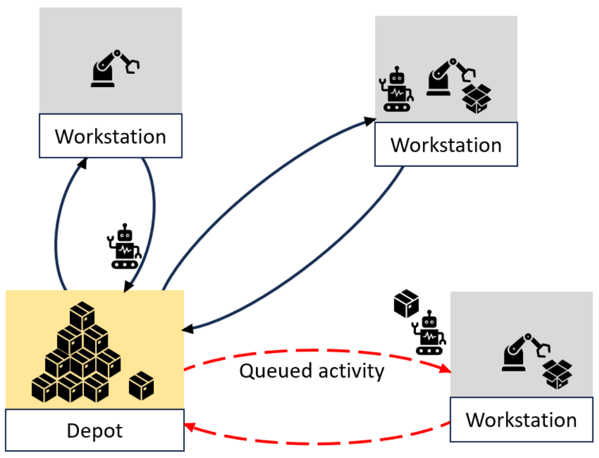

Figure 1. Schematic representation of the different elements present in the problem.

In this case study, one should differentiate between scheduling and sequencing, as outlined in Sections 1 and 2. The task allocation to workstations and AGVs is predetermined by the company. Each task is already assigned to a specific location and AGV. This means that the products designated for processing are predetermined, along with their respective travel and processing times. All the products to be processed are assumed to be available at the start time. The problem can be likened to uniform machine scheduling, but with a unique aspect: the resources are limited and shared among the different tasks to be executed (i.e., each AGV can only handle one task at a time). This characteristic creates a variant of the resource-constrained scheduling problem previously discussed in Section 2. Additionally, typical assumptions for such scheduling problems are incorporated, such as the absence of preemption (tasks cannot be interrupted) and considering resources as renewable (AGVs can perform another task once they finish the current task). The approach in this study is deterministic, implying fixed processing times for each AGV when handling a job at a specific workstation.

3.2. An Illustrative Numerical Example

Consider a scenario where there are five tasks allocated across three distinct workstations, with the assignment of tasks to workstations already predetermined. All three workstations have equal processing times. Specifically, task 1 is designated for workstation 1, tasks 2 and 4 for workstation 2, and tasks 3 and 5 for workstation 3. In this setup, there are two available AGVs. AGV A is responsible for tasks 1, 2, and 3, while AGV B handles tasks 4 and 5. The illustration of this scenario is depicted on the left side of Figure 2.

Let us consider that, for the sake of simplicity, the travel time is negligible compared to the processing time in the workstations; thus, it is not considered. In the simplest scenario with infinite resources, where there are no constraints on the AGVs, tasks can be executed whenever the workstations are available. In such cases, scheduling becomes straightforward as the sequence remains unchanged, and a first-in-first-out (FIFO) approach suffices. The total makespan is determined by the maximum number of tasks within a workstation, given they share the same processing time. This situation is depicted in the timeline (a) in Figure 2, illustrating a total makespan of two times the workstation processing time.

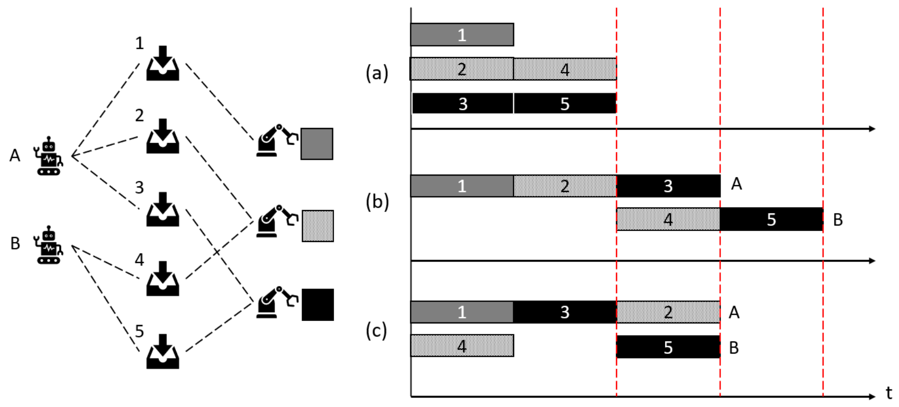

Figure 2. Illustrative example of the problem. In case (**a**), no resource constraint is considered; in case (**b**), the resource constraints are active, and the task sequence is {1, 2, 3, 4, 5}; and, in case (**c**), the resource constraints are active, and the task sequence is {1, 4, 3, 2, 5}.

When resource sharing constraints are taken into account, as shown in timelines (b) and (c) of Figure 2, the task sequence becomes crucial in determining the total makespan. In timeline b), if the task sequence {1, 2, 3, 4, 5} remains unchanged, tasks 1, 2, and 3 are executed sequentially as AGV A becomes available. Task 4 can occur concurrently with task 3 since both AGV B and the corresponding workstation are available. Task 5 is the final activity after AGV B becomes free. This results in a total makespan of four times the workstation processing time, which is twice as long as the situation with infinite resources. The resource constraint causes tasks to queue until the assigned resource is available. As the sequence is strictly adhered to, rearranging tasks can improve the total makespan.

In timeline (c) of Figure 2, the task execution order was altered to {1, 4, 3, 2, 5}. Here, tasks 1 and 4 can be performed concurrently because they are allocated to different AGVs and workstations. Following this, task 3 is executed, and although AGV B is free, the workstation is occupied, causing task 5 to queue and be executed simultaneously with task 2. This sequence change optimizes the makespan, reducing it to only three times the workstation processing time.

3.3. Generic Formulation for the Sequencing Problem with Shared Resources

Let $M = \{m_1, m_2, \ldots, m_M\}$ be the set of workstations such that workstation $m_k \in M$ has an associated processing time p_k, which could be deterministic or, in the most generic version of the problem, stochastic. Let R be the set of available resources such that AGV $r_j \in R = \{r_1, r_2, \ldots, r_R\}$. The set of tasks can be defined as the binary relation T such that the following applies:

$$T \subset R \times M = \left\{(r_j, m_k) \mid r_j \in R, m_k \in M\right\}, \tag{1}$$

with task $t_i \in T = \{t_1, t_2, \ldots, t_T\}$. The goal of the RCSP is to produce the totally ordered set $S(T, \leq)$ such that the total makespan C_{max} is minimized. The total makespan can be defined as follows:

$$C_{max} = max\Big(E(t_i) + p_k\Big), \quad \forall t_i \in S, \tag{2}$$

where $E(t_i)$ denotes the start time of a given task. The total makespan is, therefore, calculated as the maximum start time plus its corresponding processing time p_k for all the tasks in S. The activities carried out are constrained by the available resources and workstations. The current activity, $C(S, t) \subset T$, for a given sequence S at time t, is a subset of the set of tasks T that have to comply with both the resource constraint

$$\sum_{\forall t_i \in C(S,t)} \#r_j \leq 1, \tag{3}$$

and the workstation constraint

$$\sum_{\forall t_i \in C(S,t)} \#m_k \leq 1, \tag{4}$$

thus meaning that the number of times (represented by the symbol '#') that the resource r_j or the workstation m_k appears in the subset of current activities, C is less or equal than 1. In other words, a resource or a workstation can only be used once at any given point in time t.

4. Methods and Methodology

In this section, the methodology employed to address the problem introduced in Section 3 is delineated and the sequential steps and their principal components are detailed. Additionally, the various heuristic algorithms developed in this study and the instances developed for the computational experiments are described.

4.1. A Two-Step Method for Sequencing Tasks

To tackle the problem described in Section 3, a two-step process integrating heuristics and simulation was adopted. During the first step, the heuristic component aims to generate high-quality candidate sequences, minimizing the total makespan by ordering tasks efficiently. On the second step, the performance of the proposed sequence is evaluated. It is important to note that evaluating each sequence entails considering the intricate and dynamic queuing resulting from the idle time of AGVs when traveling to occupied workstations. As evidenced in the simplified example provided in Section 3.2, minor alterations in the sequence significantly impact task interferences. This complexity amplifies with an increasing number of tasks, and it becomes nearly impossible to handle by traditional methods for big instances or when stochastic variables are present. Hence, the proposal is to model the queuing phenomenon directly within Simul8. This simulation software is apt for managing complex queuing scenarios, thereby allowing for a robust evaluation of the interaction between shared resources. Moreover, utilizing Simul8 enables the easy creation of a stochastic version of the problem by introducing stochastic processing times instead of fixed processing times. Figure 3 visually illustrates the described methodology.

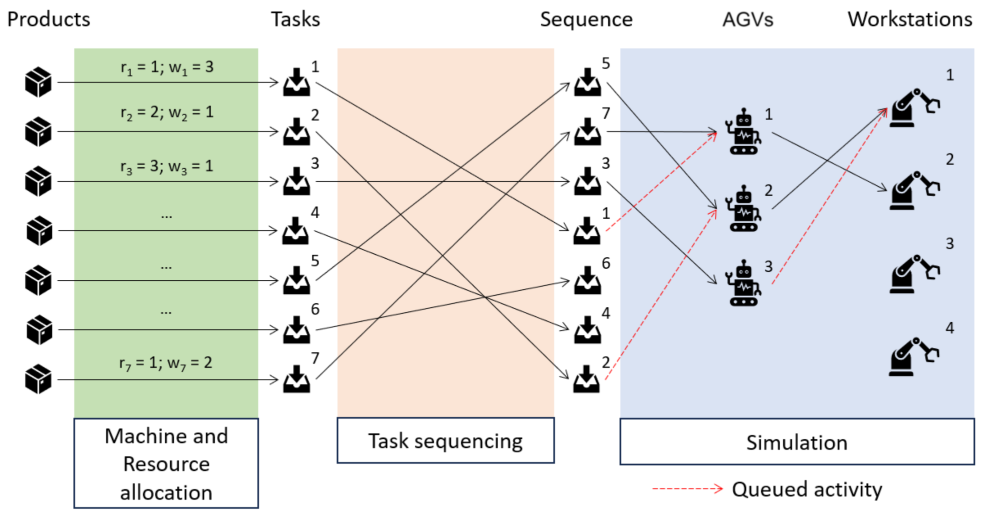

Figure 3. Schematic representation of the proposed solving methodology involving the task sequencing heuristic and Simul8 simulation of the dynamic queues.

The initial phase illustrated in Figure 3, involving the assignment of workstations and AGVs to tasks, is considered external to the core problem and is assumed as an input. Task-specific information (traveling and processing times, as well as AGV-to-workstation assignments) is encapsulated within the problem instances, which are detailed further below in Section 4.3. Essentially, the company predefines both task-to-workstation and task-to-AGV assignments. Subsequently, the task sequencing begins (the first step of the method), which make use of the heuristic algorithms outlined in Section 4.2, by focusing on determining the order of task execution. After that, at the second step of the method, the simulation component evaluates the optimized sequence, considering the dynamic queue formations as the simulation progresses and the resources are shared between the different streams of tasks. Figure 4 depicts the implementation of one of the problem instances in Simul8. It must be highlighted that the simulation strictly follows the optimized sequence, and it automatically manages the shared AGV availability depending on the dynamically generated queues.

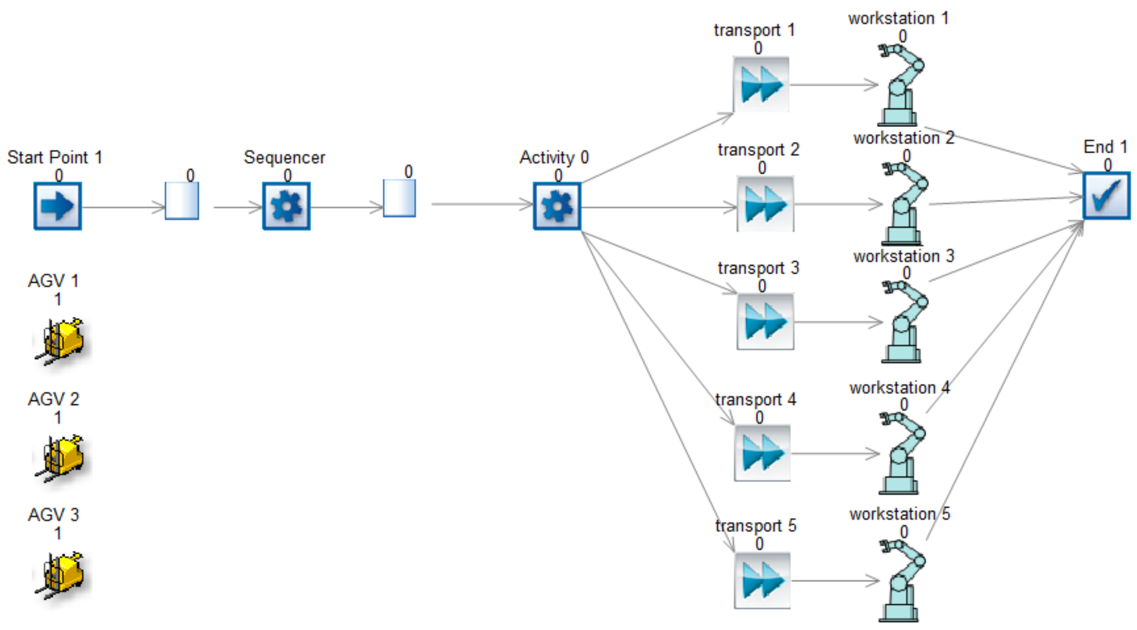

Figure 4. Screenshot of the Simul8 model for the instances with three AGVs and five workstations. The simulation model follows the sequence and automatically manages the queues for both workstations and AGVs.

4.2. Algorithm Combination and the NEH-Based Benchmark

To derive efficient task sequences minimizing the total makespan, several well-known sequencing algorithms were combined in this study to generate a number of efficient sequencing heuristics for ordering the given tasks. As elucidated in Section 1, the application of heuristics is well founded due to the impracticality of finding an optimal solution given the simplifications considered along with the inherent complexity of the problem. The heuristic algorithms were devised using the information available within the instances, including the number of workstations, AGVs, tasks, traveling and processing times for each task, and task-specific workstation and AGV allocations. The partial sequencing algorithms considered in this study were the following:

- Most loaded resource first (MLRF): This algorithm sorts AGVs, prioritizing the most loaded AGV first based on either the total processing time the AGV is engaged or the number of times it is required by a task. This study used total processing time for sorting.
- Less loaded resource first (LLRF): Similar to MLRF, LLRF sorts AGVs from lower workload to higher workload.
- Shortest processing time first (SPTF): This algorithm prioritizes tasks with the shortest processing times, sorting workstations from the least to the greatest processing time for tasks.
- Longest processing time first (LPTF): Contrary to SPTF, LPTF sorts workstations in reverse order, from lower to higher processing times.

These were referred to as partial sequencing algorithms because each one of them orders resources (i.e., AGVs) or workstations but not tasks, which are the specific combination of an AGV and a workstation, with its corresponding travel and process time. These algorithms can be viewed as individual components that can be integrated into a comprehensive heuristic procedure that generates candidate solutions for the task sequencing problem while considering shared resources. The heuristic procedure initiates by arranging the AGVs and workstations. This process involves employing the partial algorithms discussed earlier, such as sorting AGVs (MLRF and LLRF) and workstations (SPTF and LPTF). The combination of these sorting algorithms gives rise to four distinct heuristics: (i) MLRF/LPTF, (ii) MLRF/SPTF, (iii) LLRF/SPTF, and (iv) LLRF/LPTF. After organizing the AGVs and workstations, the tasks are incorporated into the final sequence through a round-robin approach. This involves commencing from the top-ranked workstation and AGV, selecting a workstation from the sorted list, and then choosing the first AGV from the corresponding list. If the pairing is not among the tasks to be performed, the next AGV is examined. This process continues until all tasks are accommodated in the final candidate sequence. Algorithm 1 showcases the MLRF/LPTF version of the heuristic. To derive other versions of the heuristic, modifications in lines 4 to 5 are required, while the rest of the procedure remains consistent. The details of the heuristic process are provided in a graphical manner in Figure 5.

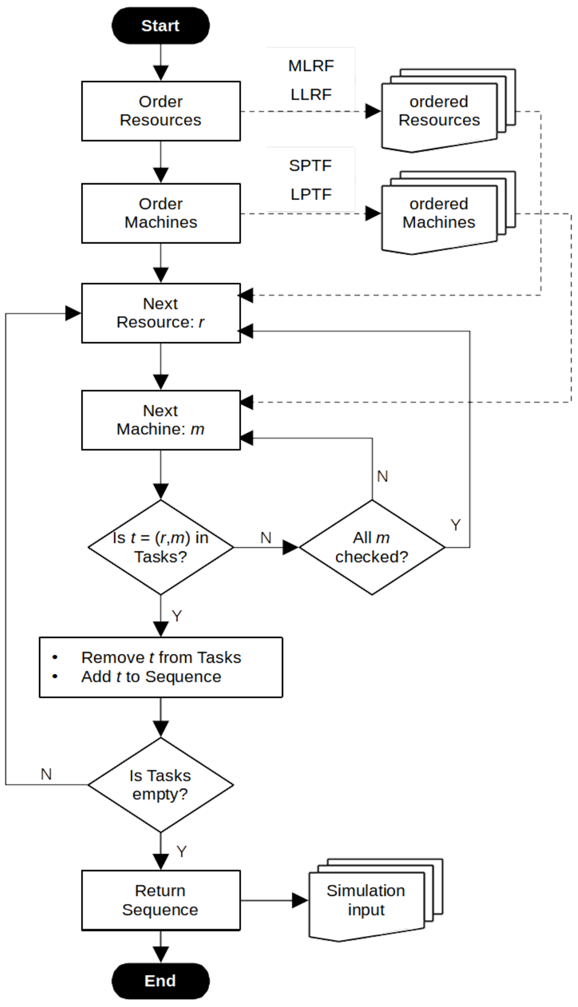

Figure 5. Schematic of the heuristic followed to generate the candidate solutions, i.e., the task sequences that were used as simulation inputs.

In order to measure the performance of our solving methodology, we provide a benchmark algorithm that uses the state-of-the-art NEH heuristic to serve as a reference baseline for comparison purposes [46]. Algorithm 2 outlines the main steps of the denoted NEH-based algorithm. Initially, the list of tasks to schedule was divided into several sub-problems based on their workstation–AGV combination. In other words, tasks with equal workstation and AGV allocation were grouped together. Each of these sub-problems were solved independently using the well-known NEH heuristic without considering resource sharing. Once a sub-sequence was generated for each sub-problem, the final sequence of tasks was built-up, combining each sub-sequence into the final task sequence. Specifically, the sub-sequences were first sorted by their number of tasks in descending order, and one task from each sub-sequence was added to the final sequence until all tasks were added to the sequence.

Algorithm 1 MLRF/LPTF algorithm.

```
Input: Tasks = $\{t_i : i \leq T\}$, Resources = $\{r_j : j \leq R\}$, Machines = $\{m_k : k \leq M\}$
Output: Sequence
1:  for Tasks do
2:      $(r_j, m_k) \leftarrow t_i$
3:      $r_j$.load ← $sum$($m_k$.processTime)          ▷ Calculate total workload per resource
4:  Resources ← $orderBy$($r_j$.load, desc)          ▷ From max. to min.
5:  Machines ← $orderBy$($m_k$.processTime, desc)    ▷ From max. to min.
6:  m ← $first$(Machines)
7:  while Tasks ≠ ∅ do
8:      r ← $next$(Resources)                        ▷ Using Round-Robin
9:      t ← (r, m)
10:     if t ∈ Tasks then
11:         Sequence ← $add$(t)
12:         Tasks ← $remove$(t)
13:         m ← $next$(Machines)
14:     else
15:         for Machines do
16:             m ← $next$(Machines)                 ▷ Using Round-Robin
17:             t ← (r, m)
18:             if t ∈ Tasks then
19:                 Sequence ← $add$(t)
20:                 Tasks ← $remove$(t)
21: $return$ Sequence
```

Algorithm 2 NEH-based algorithm.

```
Input: Tasks = $\{t_i : i \leq T\}$, Resources = $\{r_j : j \leq R\}$, Machines = $\{m_k : k \leq M\}$
Output: Sequence
1:  Combinations ← ∅
2:  for $t$ ∈ Tasks do                             ▷ Split tasks into combinations
3:      Combinations(t.machine, t.resource) ← $add$(t)
4:  SubSequences ← ∅
5:  for c ∈ Combinations do                        ▷ Solve Combinations using NEH
6:      SortedTasks ← $orderBy$(c.totalTimes, desc)
7:      s ← $HeuristicNEH$(SortedTasks)
8:      SubSequences ← $add$(s)
9:  SortedSequences ← $orderBy$(SubSequences.length, desc)
10: while SortedSequences ≠ ∅ do                   ▷ Combine Sub-Sequences
11:     for s ∈ SortedSequences do
12:         if s ≠ ∅ then
13:             t ← $removeFirst$(s)
14:             Sequence ← $add$(t)
15: $return$ Sequence
```

4.3. Computational Experiment Instances

The computational experiments were conducted on a Windows 10 operating system, utilizing an i7-4500U CPU 2.40 GHz with 6 GB RAM. All algorithms were implemented using Python v3.8.10. Simul8 2022 version 29.0 was employed for the simulation model. To evaluate the efficacy of the various heuristic algorithms developed for AGV task scheduling, multiple problem instances were generated. Table 1 presents all instances used in this study along with their key characteristics. An instance is defined by the following factors: the number of workstations, the number of AGVs available, the number of tasks to be performed, the travel and processing times for each task, and the allocation of tasks to workstations and AGVs. In the following section, both the AGV travel time and the workstation process time will be referred to as "processing time". As a matter of fact, two distinct instance types were considered for each workstation–AGV–task combination regarding processing time. The first type features processing times, derived from a uniform random distribution, tightly centered around an average value, resulting in instances with low variability in processing times (LVPT). In contrast, the second type showcases a broader range between minimum and maximum processing times, leading to instances with high variability in processing times (HVPT). Similarly, for assigning tasks to workstation–AGV combinations, two instance types were created. The first type ensures a balanced assignment, resulting in low variability in the task assignment (LVTA) to each workstation–AGV combination. Conversely, the second type introduces an unbalanced assignment, causing some workstation–AGV combinations to handle more work than others, thereby leading to high variability in task assignments (HVTAs). Combining these characteristics results in the following four distinct instance types for assessing the performance of the algorithms: (i) low variability in processing times and low variability in task assignment (LVPT/LVTA); (ii) high variability in processing times and low variability in task assignment (HVPT/LVTA); (iii) low variability in processing times and high variability in task assignment (LVPT/HVTA); and (iv) high variability in processing times and task assignment (HVPT/HVTA).

Table 1. Instances for the computational experiments.

Instance	**Workstations**	**AGVs**	**Tasks**	**Type**
$m5r3t100$_01	5	3	100	LVPT/LVTA
$m5r3t100$_02	5	3	100	HVPT/LVTA
$m5r3t100$_03	5	3	100	LVPT/HVTA
$m5r3t100$_04	5	3	100	HVPT/HVTA
$m5r3t200$_01	5	3	200	LVPT/LVTA
$m5r3t200$_02	5	3	200	HVPT/LVTA
$m5r3t200$_03	5	3	200	LVPT/HVTA
$m5r3t200$_04	5	3	200	HVPT/HVTA
$m10r5t300$_01	10	5	300	LVPT/LVTA
$m10r5t300$_02	10	5	300	HVPT/LVTA
$m10r5t300$_03	10	5	300	LVPT/HVTA
$m10r5t300$_04	10	5	300	HVPT/HVTA
$m10r5t600$_01	10	5	600	LVPT/LVTA
$m10r5t600$_02	10	5	600	HVPT/LVTA
$m10r5t600$_03	10	5	600	LVPT/HVTA
$m10r5t600$_04	10	5	600	HVPT/HVTA
$m20r10t1200$_01	20	10	1200	LVPT/LVTA
$m20r10t1200$_02	20	10	1200	HVPT/LVTA
$m20r10t1200$_03	20	10	1200	LVPT/HVTA
$m20r10t1200$_04	20	10	1200	HVPT/HVTA

5. Results and Discussion

This section presents the numerical results obtained from the computational experiments, followed by an analysis and discussion of the obtained results. Table 2 includes the output of the simulation for each instance and for each of the heuristic algorithms considered. The first column identifies the instances, while subsequent columns show the total makespan required to complete all tasks within the instance. First, we present the results obtained from the NEH-based approach, reflecting the reference baseline to be used for comparison purposes. Following this, the rest of the columns in the table display the results for the various combinations of sequencing algorithms. Additionally, the average makespan is computed for each set of instances. Finally, the percentage gaps with respect to the NEH-based approach is provided as per the following:

$$Gap[\%] = \frac{C^{i}_{max} - C^{NEH}_{max}}{C^{NEH}_{max}} \times 100, \qquad (5)$$

where C_{max} is the total makespan for each one of the $i = \{$MLRF/LPTF, MLRF/SPTF, LLRF/SPTF, LLRF/LPTF$\}$ algorithms considered. At this point, it is of relevance to focus on those situations in which the sharing of resources results in inefficiencies for a given task sequence. The inefficiencies in the sequence occur when a task is supposed to start its processing but it has to first queue (a) because there is no AGV available for transportation or (b) because the workstation to which it is traveling is already occupied by a previous task. Let consider the case in which the resources are not shared among different AGV–workstation pairs, or in which there are sufficiently abundant resources to be shared between workstations. In the context of this study, these types of situations are considered as inefficiency-free scenarios. In these cases, the NEH-based approach is expected to perform extremely well and minimize the total makespan for the different sizes and types of instances. However, for the instances analyzed, the NEH-based approach yields the highest total makespan across almost all cases since it lacks a dedicated strategy for minimizing the overall makespan when resources are shared. It can be observed that all approaches presented in this work obtain better solutions than the NEH-based approach, as can be seen by the lower makespan values and negative percentage gaps. It should be noticed that the more negative the gap is, the more substantial the improvement in the makespan relative to the NEH-based strategy. This is attributed to the improved sequence, which minimizes the possible inefficiencies and, in turn, decreases the overall sequence makespan. Apart from the obvious increment in makespan as the size of the instance increases (a larger number of tasks requires more time for completion), the algorithm performance tends to improve with larger instances. For scenarios involving more AGVs, workstations, and tasks, the algorithm consistently yields better solutions in comparison to the baseline. This trend might be attributed to the limited potential for improvement in smaller instances, where altering task sequencing has minimal impact due to less scope for enhancement.

Regarding the different types of instances, it is noteworthy that the ones that exhibit significant variability in task assignment (HVTA) result in an imbalanced usage of the system resources (the AGVs and workstations), meaning that some resources tend to have more tasks assigned to them than other resources. In turn, a lower number of interactions between the AGVs and workstations occurs in the system, and the NEH-based algorithm is able to obtain solutions that have an acceptable performance. On the other hand, instances with low variability in task assignment (LVTA) tend to result in a higher number of interactions between shared resources, in which the proposal algorithm performs better, as observed in the comparatively larger percentage gaps. One extreme case of this situation is depicted in the results obtained for the instance *m*5*r*3*t*100_04, which had a high variability in processing times and task assignment. Notably, the NEH-based approach outperformed our proposed algorithms, obtaining percentage gaps between 4.09% and 7.42%. After a comprehensive examination of the instance in the simulation, we observed that most tasks were assigned to a particular workstation–AGV combination. Subsequently, the number of interactions due to the sharing of resource was significantly low. Thus, the NEH heuristic was able to construct a high-quality sequence for that combination that minimized the makespan, and, in turn, built a better task sequence than our proposed methods.

The observed pattern suggests that our proposed algorithms are less effective when high variability is present in task assignment, meaning that the resources are not uniformly shared but assigned in an unbalanced way, as described in Section 4.3. Interestingly, this phenomenon is not encountered in the largest set of instances. This can be attributed to the increase in the number of tasks compared to the increase in number of AGVs and workstations. As the number of tasks increases, the variability in task assignment tends to be mitigated due to a higher number of tasks available for sequencing. Hence, the interaction between resources tend to take place regardless of the level of variability in task assignment. These results point toward a promising scalability of our proposed methodology, especially when dealing with a substantial number of tasks in an environment with shared resources. Additionally, it appears that the algorithm performance also diminishes in instances characterized by high variability in processing times (HVPT). This could be explained by the inefficiencies imposed by exceptionally large processing times for certain workstations, leading to unavoidable queues and subsequently reducing the efficacy of sequence alterations. The boxplot depicted in Figure 6 illustrates the average percentage gaps observed for the MLRF/LPTF, MLRF/SPTF, LLRF/SPTF, and LLRF/LPTF strategies in comparison to the NEH-based approach. As indicated in Section 4.2, the NEH-based approach was selected as the reference baseline for assessing other strategies, as its performance can be considered at the state-of-the-art level, at least in scenarios without resource sharing interactions. Hence, the NEH-based heuristic can help when evaluating the relative performance of alternative strategies. Notice that the average percentage gap consistently fell below zero, indicating an improvement in the average makespan compared to the NEH-based approach across all instances, where the average percentage gaps ranged between −16.82% and −19.49%. The large gap with respect to the NEH-based approach suggests that the improvement could be due to the round-robin sequencing rather than due to the combination of different sorting algorithms for the AGVs and workstations. The conclusion is that, for situations in which resource sharing becomes critical for the application, the balancing of resources is more important than the quality of the sequencing for a given stream of work. In the case presented, the combination of simple sequencing strategies outperforms the NEH algorithm when naively used in a context in which resources are limited and have to be shared among different streams of work.

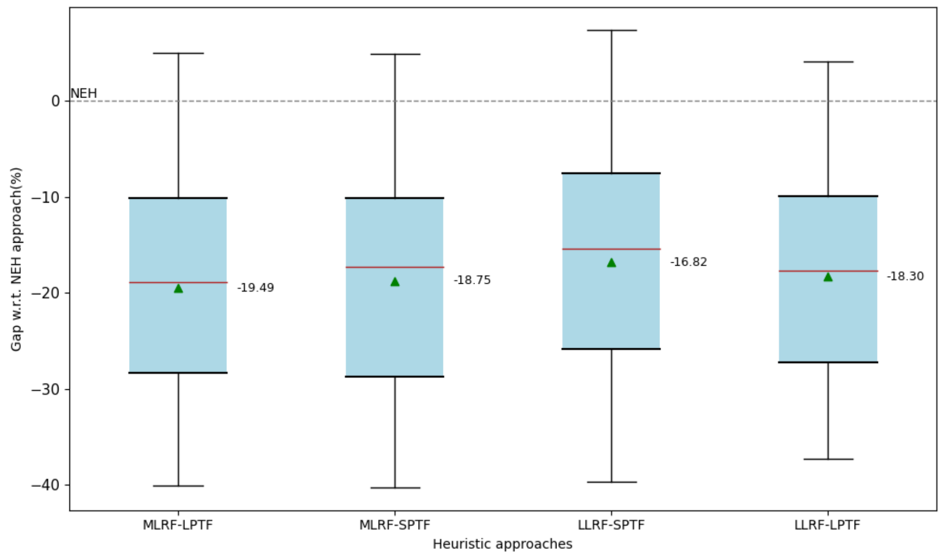

Figure 6. Boxplot of the gaps of different heuristic approaches with respect to the NEH approach. The red line indicates the median and the green triangle the mean of the data set.

Table 2. Results of the computational experiments in terms of the total makespan and percentage gap with respect to the NEH strategy.

Instance	NEH	MLRF/LPTF	Gap (%)	MLRF/SPTF	Gap (%)	LLRF/SPTF	Gap (%)	LLRF/LPTF	Gap (%)
$m5r3t100$_01	6594	5248	−20.41%	5377	−18.46%	5480	−16.89%	5310	−19.47%
$m5r3t100$_02	6838	5674	−17.02%	6061	−11.36%	6414	−6.20%	5536	−19.04%
$m5r3t100$_03	8644	8125	−6.00%	8115	−6.12%	8568	−0.88%	8277	−4.25%
$m5r3t100$_04	8688	9121	4.98%	9115	4.91%	9333	7.42%	9043	4.09%
Average:	7691	7042	−9.61%	7167	−7.76%	7448.75	−4.14%	7041.50	−9.67%
$m5r3t200$_01	13,818	9752	−29.43%	10,090	−26.98%	10,405	−24.70%	9946	−28.02%
$m5r3t200$_02	14,761	11,016	−25.37%	11,509	−22.03%	11,895	−19.42%	11,350	−23.11%
$m5r3t200$_03	16,896	14,528	−14.02%	14,623	−13.45%	15,048	−10.94%	14,921	−11.69%
$m5r3t200$_04	17,236	15,986	−7.25%	16,410	−4.79%	16,607	−3.65%	16,402	−4.84%
Average:	15,677.75	12,820.50	−19.02%	13,158	−16.81%	13,488.75	−14.68%	13,154.75	−16.91%
$m10r5t300$_01	14,966	11,249	−24.84%	10,613	−29.09%	11,866	−20.71%	10,918	−27.05%
$m10r5t300$_02	14,719	11,393	−22.60%	11,786	−19.93%	12,243	−16.82%	12,226	−16.94%
$m10r5t300$_03	24,557	22,198	−9.61%	22,438	−8.63%	22,589	−8.01%	22,453	−8.57%
$m10r5t300$_04	23,817	21,690	−8.93%	21,828	−8.35%	22,546	−5.34%	21,905	−8.03%
Average:	19,514.75	16,632.50	−16.49%	16,666.25	−16.50%	17,311	−12.72%	16,875.50	−15.15%
$m10r5t600$_01	36,203	23,095	−36.21%	22,665	−37.39%	22,094	−38.97%	24,633	−31.96%
$m10r5t600$_02	36,759	26,424	−28.12%	25,766	−29.91%	25,414	−30.86%	26,826	−27.02%
$m10r5t600$_03	53,791	47,427	−11.83%	47,565	−11.57%	47,684	−11.35%	47,628	−11.46%
$m10r5t600$_04	54,801	49,184	−10.25%	48,980	−10.62%	49,037	−10.52%	49,149	−10.31%
Average:	45,388.50	36,532.50	−21.60%	36,244	−22.37%	36,057.25	−22.93%	37,059	−20.19%
$m20r10t1200$_01	52,242	31,338	−40.01%	31,198	−40.28%	31503	−39.70%	32,768	−37.28%
$m20r10t1200$_02	54,533	38,646	−29.13%	38,945	−28.58%	37,442	−31.34%	37,821	−30.65%
$m20r10t1200$_03	75,115	45,027	−40.06%	45,039	−40.04%	47,707	−36.49%	47,524	−36.73%
$m20r10t1200$_04	78,231	54,280	−30.62%	55,022	−29.67%	55,259	−29.36%	54,306	−30.58%
Average:	65,030.25	42,322.75	−34.95%	42,551	−34.64%	42,977.75	−34.22%	43,104.75	−33.81%

In detail, the approaches that give priority to the most loaded resource (MLRF/LPTF and MLRF/SPTF) seem to give as good or even better results compared to the other two approaches. This can be attributed to the way the studied benchmark instances are specified, as the number of AGVs is always lower than the number of workstations. Thus, the most important resource to take into account when generating an improved sequence of tasks is to consider the process times in the AGVs in order to prioritize the most loaded ones. Moreover, the strategies that prioritize workstations with longer processing times (LPTF) also tend to perform very well since they prevent excessive workstation occupancy during the later stages of task processing. Thus, sequences favoring the LPTF strategy over SPTF exhibit lower percentage gaps, underscoring its effectiveness across nearly all instances. The MLRF/LPTF algorithm emerges as the most effective sequencing heuristic for addressing the proposed problem. As already discussed, its superiority primarily stems from the fact that it strategically places the most heavily utilized AGVs and most loaded workstations at the beginning of the sequencing, thereby mitigating potential queue formations and inefficiencies that might arise later in the process.

6. Conclusions

This study addressed the complex challenge of optimizing task sequencing within manufacturing environments. The primary objective was to minimize the total production time, also known as the makespan, by determining the most efficient order for executing tasks assigned to AGVs and workstations. The methodology proposed in this study employed a two-step process integrating heuristic algorithms and simulation techniques. The heuristics were designed to generate high-quality candidate sequences considering resource utilization, task assignment, and processing times. Simul8, a discrete-event simulation software, was utilized to model the complex queuing dynamics arising from the AGV task sequencing, providing a robust evaluation framework for the proposed sequencing strategies. From the developed heuristic algorithms, MLRF/LPTF emerged as a superior strategy for task sequencing. This algorithm strategically prioritized the most loaded AGVs and workstations with the highest processing times, resulting in substantial improvements in the makespan compared to the default sequencing strategy, which was based on the well-known NEH algorithm. The performance of the proposed algorithms demonstrated consistent enhancements, especially for larger instances, emphasizing their effectiveness in improving AGV task sequencing efficiency in the context of limited and shared resources. Nonetheless, the analysis revealed a correlation between the variability in the task assignment and processing times, as well as in the algorithms' performance. Instances characterized by high variability exhibited reduced improvement potential due to dominating factors such as non-uniformly load machines with very long processing times, thus limiting the algorithms' effectiveness.

Building upon the insights gained from this study, several research lines emerge. A future line of research that naturally extends from this study involves the development of a stochastic version of the presented problem. This stochastic behavior can be achieved by transforming the fixed task/activity process times into random variables selected from specific probability distributions, such as log-normal or Weibull distributions. This direction benefits from the methodology established in this study, where the evaluation of the final scheduling utilizes a discrete-event simulator like Simul8. Leveraging such a simulator enables the realistic simulation of dynamic queues resulting from stochastic processing times. Furthermore, enhancing the interaction between the algorithmic component and simulation could allow for more integrated simulation–optimization frameworks, often referred to as simheuristics [35,47]. This integration holds promise for efficiently deriving high-quality and robust solutions that consider the stochastic nature inherent in warehouse and production processes. Another research line could involve extending the constructive heuristics into biased-randomized algorithms [48] capable of providing enhanced results, and also utilizing the simulation developed in this study as an environment for a reinforcement learning agent. Such an agent could learn optimal sequencing policies by being dynamically based on the current production state. By adapting to queues and resource sharing in a changing environment, the agent could propose improved sequencing strategies over time. Finally, investigating the scalability and applicability of the developed heuristic algorithms to diverse industrial scenarios could be an area for further investigation.

Author Contributions: Conceptualization, J.F.L. and A.A.J.; methodology, J.F.L.; validation, J.F.L., M.P. and X.A.M.; writing—original draft preparation, J.F.L., M.P. and X.A.M.; writing—review and editing, J.F.L. and A.A.J.; supervision, J.F.L. and A.A.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the European Commission (AIDEAS HORIZON-CL4-2021-TWIN-TRANSITION-01-07-101057294 and SUN HORIZON-CL4-2022-HUMAN-01-14-101092612), Simul8, Spindox, AGAUR (2020-DI-116), and the Spanish Ministry of Science and Innovation (PRE2020-091842).

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: Author J.F.L. was employed by the company Spindox España S.L. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Klingenberg, C.O.; Borges, M.A.V.; Antunes, J.A.d.V. Industry 4.0: What makes it a revolution? A historical framework to understand the phenomenon. *Technol. Soc.* **2022**, *70*, 102009. [CrossRef]
2. Bechtsis, D.; Tsolakis, N.; Vlachos, D.; Iakovou, E. Sustainable supply chain management in the digitalisation era: The impact of Automated Guided Vehicles. *J. Clean. Prod.* **2017**, *142*, 3970–3984. [CrossRef]
3. Salkin, C.; Oner, M.; Ustundag, A.; Cevikcan, E. A Conceptual Framework for Industry 4.0. In *Industry 4.0: Managing The Digital Transformation*; Ustundag, A., Cevikcan, E., Eds.; Springer Series in Advanced Manufacturing; Springer International Publishing: Cham, Switzerland, 2018; pp. 3–23.
4. Vis, I.F.A. Survey of research in the design and control of automated guided vehicle systems. *Eur. J. Oper. Res.* **2006**, *170*, 677–709. [CrossRef]
5. Ruiz, R. Scheduling Heuristics. In *Handbook of Heuristics*; Martí, R., Pardalos, P.M., Resende, M.G.C., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 1197–1220.
6. Vivaldini, K.C.T.; Rocha, L.F.; Becker, M.; Moreira, A.P. Comprehensive Review of the Dispatching, Scheduling and Routing of AGVs. In Proceedings of the CONTROLO'2014—Proceedings of the 11th Portuguese Conference on Automatic Control, Porto, Portugal, 21–23 July 2014 ; Moreira, A.P.; Matos, A.; Veiga, G., Eds.; Lecture Notes in Electrical Engineering; Springer: Cham, Switzerland, 2015; pp. 505–514.
7. Alatartsev, S.; Stellmacher, S.; Ortmeier, F. Robotic Task Sequencing Problem: A Survey. *J. Intell. Robot. Syst.* **2015**, *80*, 279–298. [CrossRef]
8. Blazewicz, J.; Lenstra, J.K.; Kan, A.H.G.R. Scheduling subject to resource constraints: Classification and complexity. *Discret. Appl. Math.* **1983**, *5*, 11–24. [CrossRef]
9. Pinedo, M.L. Modeling and Solving Scheduling Problems in Practice. In *Scheduling: Theory, Algorithms, and Systems*; Pinedo, M.L., Ed.; Springer International Publishing: Cham, Switzerland, 2016; pp. 431–458.
10. Leon, J.F.; Marone, P.; Peyman, M.; Li, Y.; Calvet, L.; Dehghanimohammadabadi, M.; Juan, A.A. A Tutorial on Combining Flexsim with Python for Developing Discrete-Event Simheuristics. In Proceedings of the 2022 Winter Simulation Conference (WSC), Singapore, 11–14 December 2022; pp. 1386–1400.
11. Fransen, K.; van Eekelen, J. Efficient path planning for automated guided vehicles using A*(Astar) algorithm incorporating turning costs in search heuristic. *Int. J. Prod. Res.* **2023**, *61*, 707–725. [CrossRef]
12. Pratissoli, F.; Battilani, N.; Fantuzzi, C.; Sabattini, L. Hierarchical and Flexible Traffic Management of Multi-AGV Systems Applied to Industrial Environments. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 10009–10015.
13. Vlachos, I.; Pascazzi, R.M.; Ntotis, M.; Spanaki, K.; Despoudi, S.; Repoussis, P. Smart and flexible manufacturing systems using Autonomous Guided Vehicles (AGVs) and the Internet of Things (IoT). *Int. J. Prod. Res.* **2022**, 1 –22. [CrossRef]
14. Reis, W.P.N.d.; Couto, G.E.; Junior, O.M. Automated guided vehicles position control: A systematic literature review. *J. Intell. Manuf.* **2023**, *34*, 1483–1545. [CrossRef]
15. Oyekanlu, E.A.; Smith, A.C.; Thomas, W.P.; Mulroy, G.; Hitesh, D.; Ramsey, M.; Kuhn, D.J.; Mcghinnis, J.D.; Buonavita, S.C.; Looper, N.A.; et al. A Review of Recent Advances in Automated Guided Vehicle Technologies: Integration Challenges and Research Areas for 5G-Based Smart Manufacturing Applications. *IEEE Access* **2020**, *8*, 202312–202353. [CrossRef]

16. Kim, K.H.; Bae, J.W. A Look-Ahead Dispatching Method for Automated Guided Vehicles in Automated Port Container Terminals. *Transp. Sci.* **2004**, *38*, 224–234. [CrossRef]
17. Li, G.; Li, X.; Gao, L.; Zeng, B. Tasks assigning and sequencing of multiple AGVs based on an improved harmony search algorithm. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *10*, 4533–4546. [CrossRef]
18. Zou, W.Q.; Pan, Q.K.; Meng, T.; Gao, L.; Wang, Y.L. An effective discrete artificial bee colony algorithm for multi-AGVs dispatching problem in a matrix manufacturing workshop. *Expert Syst. Appl.* **2020**, *161*, 113675. [CrossRef]
19. Hari, S.K.K.; Nayak, A.; Rathinam, S. An Approximation Algorithm for a Task Allocation, Sequencing and Scheduling Problem Involving a Human-Robot Team. *IEEE Robot. Autom. Lett.* **2020**, *5*, 2146–2153. [CrossRef]
20. Dike, S.H. Project scheduling with resource constraints. *IEEE Trans. Eng. Manag.* **1964**, *EM-11*, 155–157. [CrossRef]
21. Ding, H.; Zhuang, C.; Liu, J. Extensions of the resource-constrained project scheduling problem. *Autom. Constr.* **2023**, *153*, 104958. [CrossRef]
22. Van Eynde, R.; Vanhoucke, M. Resource-constrained multi-project scheduling: Benchmark datasets and decoupled scheduling. *J. Sched.* **2020**, *23*, 301–325. [CrossRef]
23. Hartman, S.; Briskorn, D. A survey of variants and extensions of the resource-constrained project scheduling problem. *Oper. Res. Manag. Sci.* **2011**, *51*, 67.
24. van der Beek, T.; Souravlias, D.; van Essen, J.; Pruyn, J.; Aardal, K. Hybrid differential evolution algorithm for the resource constrained project scheduling problem with a flexible project structure and consumption and production of resources. *Eur. J. Oper. Res.* **2023**, *313* , 92–111. [CrossRef]
25. Maimon, O. The robot task-sequencing planning problem. *IEEE Trans. Robot. Autom.* **1990**, *6*, 760–765. [CrossRef]
26. Suárez-Ruiz, F.; Lembono, T.S.; Pham, Q.C. Robotsp—A fast solution to the robotic task sequencing problem. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1611–1616.
27. Li, D.; Wang, Q.; Zou, W.; Su, H.; Wang, X.; Xu, X. An Efficient Approach for Solving Robotic Task Sequencing Problems Considering Spatial Constraint. In Proceedings of the 2022 IEEE 18th International Conference on Automation Science and Engineering (CASE), Mexico City, Mexico, 20–24 August 2022; pp. 60–66.
28. Chen, C.H.; Chou, F.I.; Chou, J.H. Optimization of robotic task sequencing problems by crowding evolutionary algorithms. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *52*, 6870–6885. [CrossRef]
29. Li, H.; Liu, S.Y.; Huang, Y.W.; Chen, Y.Q.; Fu, Z.H. An Efficient 2-opt Operator for the Robotic Task Sequencing Problem. In Proceedings of the 2019 IEEE International Conference on Real-time Computing and Robotics (RCAR), Irkutsk, Russia, 4–9 August 2019; pp. 124–129.
30. Martin, X.A.; Hatami, S.; Calvet, L.; Peyman, M.; Juan, A.A. Dynamic Reactive Assignment of Tasks in Real-Time Automated Guided Vehicle Environments with Potential Interruptions. *Appl. Sci.* **2023**, *13*, 3708. [CrossRef]
31. Korsah, G.A.; Stentz, A.; Dias, M.B. A comprehensive taxonomy for multi-robot task allocation. *Int. J. Robot. Res.* **2013**, *32*, 1495–1512. [CrossRef]
32. Khamis, A.; Hussein, A.; Elmogy, A. Multi-robot task allocation: A review of the state-of-the-art. In *Cooperative Robots and Sensor Networks 2015*; Springer, Cham, Switzerland , 2015; pp. 31–51.
33. Alshaboti, M.; Baroudi, U. Multi-robot task allocation system: Fuzzy auction-based and adaptive multi-threshold approaches. *SN Comput. Sci.* **2021**, *2*, 1–11. [CrossRef]
34. Chakraa, H.; Guérin, F.; Leclercq, E.; Lefebvre, D. Optimization techniques for Multi-Robot Task Allocation problems: Review on the state-of-the-art. *Robot. Auton. Syst.* **2023**, p. 104492. [CrossRef]
35. Leon, J.F.; Li, Y.; Peyman, M.; Calvet, L.; Juan, A.A. A Discrete-Event Simheuristic for Solving a Realistic Storage Location Assignment Problem. *Mathematics* **2023**, *11*, 1577. [CrossRef]
36. López, J.; Zalama, E.; Gómez-García-Bermejo, J. A simulation and control framework for AGV based transport systems. *Simul. Model. Pract. Theory* **2022**, *116*, 102430. [CrossRef]
37. Inoue, K. Discrete-Event Simulation for Autonomous Guided Vehicle. *IFAC Proc. Vol.* **2001**, *34*, 87–92. [CrossRef]
38. bin Md Fauadi, M.H.F.; Li, W.L.; Murata, T.; Prabuwono, A.S. Vehicle requirement analysis of an AGV system using discrete-event simulation and data envelopment analysis. In Proceedings of the 2012 8th International Conference on Computing Technology and Information Management (NCM and ICNIT), Seoul, Republic of Korea, 24–26 April 2012; Volume 2, pp. 819–823.
39. Kühn, M.; Schmidt, T.; Völker, M. Simulation-Based Optimization Approach for Efficient Generation of Sequencing Heuristics for Solving the Stochastic Resource-Constrained Scheduling Problem. In *Proceedings of the Simulation in Produktion und Logistik*; Kassel University Press: Kassel, Germany, 2019; pp. 403–412.
40. Azimi, P.; Sholekar, S. A simulation optimization approach for the multi-objective multi-mode resource constraint project scheduling problem. *Int. J. Ind. Eng. Prod. Res* **2021**, *32*, 37–45.
41. Kesen, S.E.; Baykoç, O.F. Simulation of automated guided vehicle (AGV) systems based on just-in-time (JIT) philosophy in a job-shop environment. *Simul. Model. Pract. Theory* **2007**, *15*, 272–284. [CrossRef]
42. Chen, J.C.; Chen, T.L.; Teng, Y.C. Meta-model based simulation optimization for automated guided vehicle system under different charging mechanisms. *Simul. Model. Pract. Theory* **2021**, *106*, 102208. [CrossRef]
43. Viana, J.; Brailsford, S.C.; Harindra, V.; Harper, P.R. Combining discrete-event simulation and system dynamics in a healthcare setting: A composite model for Chlamydia infection. *Eur. J. Oper. Res.* **2014**, *237*, 196–206. [CrossRef]

44. Mensah, P.; Merkuryev, Y.; Longo, F. Using ICT in Developing a Resilient Supply Chain Strategy. *Procedia Comput. Sci.* **2015**, *43*, 101–108. [CrossRef]
45. Wales, J.; Marinov, M. Analysis of delays and delay mitigation on a metropolitan rail network using event based simulation. *Simul. Model. Pract. Theory* **2015**, *52*, 52–77. [CrossRef]
46. Nawaz, M.; Enscore, E.E., Jr.; Ham, I. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega* **1983**, *11*, 91–95. [CrossRef]
47. Hatami, S.; Calvet, L.; Fernández-Viagas, V.; Framinan, J.M.; Juan, A.A. A simheuristic algorithm to set up starting times in the stochastic parallel flowshop problem. *Simul. Model. Pract. Theory* **2018**, *86*, 55–71. [CrossRef]
48. Dominguez, O.; Juan, A.A.; Faulin, J. A biased-randomized algorithm for the two-dimensional vehicle routing problem with and without item rotations. *Int. Trans. Oper. Res.* **2014**, *21*, 375–398. [CrossRef]