Graph-Based Modeling in Shop Scheduling Problems: Review and Extensions

Jacqueline Otala, Alden Minard, Golshan Madraki * and Seyedamirabbas Mousavian

Abstract: Graphs are powerful tools to model manufacturing systems and scheduling problems. The complexity of these systems and their scheduling problems has been substantially increased by the ongoing technological development. Thus, it is essential to generate sustainable graph-based modeling approaches to deal with these excessive complexities. Graphs employ nodes and edges to represent the relationships between jobs, machines, operations, etc. Despite the significant volume of publications applying graphs to shop scheduling problems, the literature lacks a comprehensive survey study. We proposed the first comprehensive review paper which (1) systematically studies the overview and the perspective of this field, (2) highlights the gaps and potential hotspots of the literature, and (3) suggests future research directions towards sustainable graphs modeling the new intelligent/complex systems. We carefully examined 143 peer-reviewed journal papers published from 2015 to 2020. About 70% of our dataset were published in top-ranked journals which confirms the validity of our data and can imply the importance of this field. After discussing our generic data collection methodology, we proposed categorizations over the properties of the scheduling problems and their solutions. Then, we discussed our novel categorization over the variety of graphs modeling scheduling problems. Finally, as the most important contribution, we generated a creative graph-based model from scratch to represent the gaps and hotspots of the literature accompanied with statistical analysis on our dataset. Our analysis showed a significant attention towards job shop systems (56%) and Un/Directed Graphs (52%) where edges can be either directed, or undirected, or both, Whereas 14% of our dataset applied only Undirected Graphs and 11% targeted hybrid systems, e.g., mixed shop, flexible, and cellular manufacturing systems, which shows potential future research directions.

Keywords: graph theory; shop scheduling problem; modeling; manufacturing systems; a systematic review and categorization methodology

1. Introduction

Graph theory is an important and practical division in mathematics. It is defined as the study of graphs which are a visual representation and a mathematical structure showing the relations between elements and/or features of a problem ([1]). Graphs have two major components: nodes (also known as vertices and points) and edges (also known as links, lines, arrows) connecting nodes together, dictating the relationship between nodes. Edges can be directed or undirected. If a node takes precedence over another node, then this relation is represented by a directed edge. otherwise, an undirected edge should be used connecting two symmetric nodes.

Graph theory was first published in 1735 by Euler with the solution of the Königsberg bridge problem [2]. Euler turned land masses into nodes and bridges into undirected edges, finding that his solution depended on a characteristic of the graph: the degrees of each node. Rather than simplifying this problem, Euler’s method clarified the problem into only the necessary components and relationships for understanding and solving. Since 1735,
graph theory has expanded vastly in both applications and complexity. Today, graph theory includes hundreds of different types of graphs, algorithms, heuristics, and accompanying mathematical structures.

The major application of graph theory is to model a variety of problems across disciplines and industries in which networks exist including physics [3,4], computer science [5,6], social networks [7,8], political science [9], artificial intelligence [10], biology [11], electrical power systems [12–14], supply chain network [15,16], we well as global crisis research such as COVID-19 outbreak [17].

Manufacturing systems benefit greatly from graph theory because they are all, at a fundamental level, networks of assets processing materials, parts, and information [18]. With this knowledge, they can be converted into graphical models efficiently and then analyzed. Shop scheduling is a critical subsystem of control and planning process in manufacturing systems, and directly affects the system productivity. Shop scheduling problems have been profoundly studied within the past decades. Scheduling generally refers to the assignment of a set of jobs to a set of resources (usually machines) within the course of time. As the size of sets of jobs and machines gets larger, finding a feasible schedule becomes harder. The ultimate goal in a scheduling problem is to find an efficient schedule which yields optimal results for the designated system performance measure/s (objective/s) [19].

Despite the significantly increase in number of publications using graph-based models to address variety of manufacturing system problems and shop scheduling problems, the literature lacks a comprehensive review paper to systematically study the overview and the perspective of this research field.

To address this gap, we surveyed those studied utilizing graph theory to model and/or solve different variations of shop scheduling problems in the domain of manufacturing systems. We focused on articles published in peer-reviewed academic journals from 2015 to 2020 and skip conference proceedings and workshop publications.

To the best of our knowledge, this paper is the first comprehensive review paper targeting the intersectional of shop scheduling problem and graph theory research. To confirm this statement, an extended search on only review papers published in this fields has been implemented. We used the “Advance Search” tool of Google Scholar search engine and did not consider any time boundary for the search. Combination of different keywords were applied including: “Graph”, “Scheduling”, “review”, “survey”, “manufacturing”, “shop”, etc. The results of this search confirmed our statement: Table 1 summarizes and compares the closest review papers found in this specified search. As this indicates, none of these review papers are remotely comparable to our study in terms of scope, topics, contents, and the magnitude of the dataset. In this table, the first four papers are technically in the Compute Science domain.

Our major contributions are to organize our systematic literature review around the following critical points:

- We proposed a novel subjective measurement to evaluate the quality of the references in review papers. This procedure and measurement have not been observed in any previous review papers. Analyzing the distribution of publications based on the quality of journals can indicate the research trend and hotspot, as well as verify the validation of the dataset.
- We proposed a novel categorization over the most common and efficient graph-based models being used in the shop scheduling problems.
- We presented a comprehensive statistical analysis of the current dataset using both visualization tools and master tables where references can be easily compared across different criteria/categories.
- The most important breakthrough of this study is that we generated a unique modeling approach from scratch to visualize both gaps and hotspots in the literature based on our findings/analysis in the dataset. This creative model puts theory into practice and show the functionality of different types of graphs to model any systems, including a review survey, which is defined as a system here.
Table 1. Comparing the existing review papers in the literature with this study.

| Domain                                      | Review Paper | Author-Year of Publication | Number of Reviewed Papers | The Time Period of Reviewed Papers | Targeted Topics                                                                 | Additional Description                      |
|---------------------------------------------|--------------|-----------------------------|---------------------------|------------------------------------|---------------------------------------------------------------------------------|---------------------------------------------|
| Computer Science                            | [20]         | (Singh et al., 2015)        | 14                        | All before 1998                    | The processor scheduling algorithms, DAG                                           | Only DAG                                    |
|                                             | [21]         | (Sachdeva & Panwar, 2015)   | 19                        | 1979–2014                          | DAG, Genetic Algorithm                                                           | Only DAG                                    |
|                                             | [22]         | (Weise et al., 2018)        | 67                        | No time limit                      | Reviewing 9 graph-based systems, e.g., Ant colony, AND/OR, Social Networks, etc. | Investigating applications to manufacturing systems- Not scheduling problems |
|                                             | [23]         | (da Silva & Gabriel, 2020)  | 56                        | 1990–2020                          | The processor scheduling algorithms, DAG                                           | Only DAG                                    |
|                                             | [24]         | (Blazewicz & Kobler, 2002)  | 26                        | All before 1999                    | Precedence graphs, unrelated parallel machines scheduling Petri nets, shop        | Only Precedence graph                       |
|                                             | [25]         | (Tuncel & Bayhan, 2007)     | 20                        | 1989–2005                          | scheduling problem Flexible manufacturing systems, Multi-Criteria Decision Making, | Only Petri-net                              |
|                                             | [26]         | (Yadav & Jayswal, 2018)     | 70                        | No time limit                      | Petri Net, etc.                                                                 |                                             |
|                                             | [27]         | (Panwalkar & Koulamas, 2019)| 56                        | No time limit                      | Flow shop-Schematic representations                                              | Not necessarily graph                       |
|                                             | [28]         | (Sotskov, 2020)             | 88                        | No time limit                      | Mixed Graph Colorings, Shop scheduling problem                                  | Only mathematical theory of mixed graph coloring |
|                                             | Our paper    | Our paper (2021)            | 143                       | 2015–2021                          | Graph theory, shop scheduling problem                                           | All types of applied graphs                |

The rest of this paper is organized as follows. Section 2 describes our methodology to collect the dataset and provide an overview of the meta-data associated with the collected references, including the publication year, journals and publishers, and their quality analysis. Sections 3 and 4 discuss a background of variety of shop scheduling problems and their solutions that were observed in the dataset. Section 5 introduces the fundamental concepts of all graph-based modeled utilized in our dataset and their applications to shop scheduling problems. Section 6 includes the full list of the dataset sorted based on the type of the graphs they used. Also, this section analyzes the statistics of the dataset in each proposed macro and micro categories. Section 7 summarizes the results of analysis across multiple categories. Eventually, the important takeaways of our analysis are discussed in Section 8, where the future research directions are suggested afterward.

2. Materials and Methods

In this paper, a systematic search process was conducted to assess the body of literature on the application of the graph theory in shop scheduling problems. This systematic literature review was implemented through an iterative process of defining relevant and appropriate keywords, reviewing the search results in the literature, reference chasing, and categorizing the final search results. The details of our research process are as follows:
To compile our dataset, we used the most common search engines “Google” and “Google-Scholar”. We confined our search exclusively to peer-reviewed academic journal publication. In our systematic search process, we used the combination of relevant keywords and selected papers which are pertinent to the modeling of manufacturing systems and shop scheduling problems using graph-based models and the application of graph theory in this context. This combination of keywords must include at least one keyword related to manufacturing system and one related to graph-theory. For instance, “graph+shop+scheduling” was one of the most successful combinations of keywords in our trials.

Also, to have a more in-depth and broader search results, we used several well-known publishers’ databases such as Elsevier Springer, Taylor & Francis, Emerald Publishing, Wiley Online Library, INFORMS, IEEE Digital Library, etc.

This initial search resulted in finding 175 journal articles. Then, the contents of each article were evaluated by three independent researchers and further reference chasing was done. This process has been implemented iteratively until we found a satisfactory number of references. After the final round of data filtering, eventually, 143 papers were selected whose contents were completely in the scope of this study. The finally selected papers were classified and sorted based on the quality of journal and the year of publication.

Figure 1 summarizes the steps of our systematic search process to carry out the literature review in the domain of graph theory and shop scheduling problems.

![Figure 1](image-url)

**Figure 1.** The systematic search process utilized in this paper.

To give a brief context to the depth and variety of our dataset, the breakdown analysis by year, publisher, and the quality of the journal are represented for the 143 total papers.

Figure 2 shows the dataset breakdown by year (2015 to 2020) where a relatively consistent number of publications per year is observed in this dataset. This could potentially refer to a steady interest in the topic. Note that a couple of references in our dataset were officially published within January and February of 2021, counting in the 2020 category.

Figure 3 summarizes the breakdown of collected papers by their publisher, i.e., each paper was labeled with its parent publishing organization. The ‘Other’ category includes publishers that only appeared once in our dataset. For example, World Scientific, Research India Publications, Genamatics, Scimago, MIT Press, and seven other publishers. Elsevier is the most popular publisher in the overlapped domains of graph theory and shop scheduling problem. It published 41% of our dataset followed by SpringerLink publishing 18% of the collected papers.
Furthermore, a subjective categorization over the quality of the target journals is proposed in Figure 4. The prestige sorting was done by observing each journal’s Eigen-Factor and latest ABDC score [29,30]. Journals were sorted into three tiers of prestige: premier (>90th percentile in EigenFactor or ‘A*’ rating in ABDC), major (>50th percentile in EigenFactor or ‘A’ rating in ABDC), and other (journals that did not meet premier or major criteria).

According to our definition of Premier/Major/Other, the ‘Premier’ category includes the least number of journals (due to the strictest criteria) and the ‘Other’ category contains the greatest number of journals (having the loosest criteria). However, the distribution of our dataset does not comply with this order, i.e., the ‘Major’ journals are observed the most frequently in our dataset (Figure 4). An optimistic reason for this observation can be that this area of research keeps attracting a relatively great attention by more prestigious journals due to its potential applicability.

Another metric which might be able to describe the impact of the collected references is the number of citations. The number of citations for all references in the dataset was up to date as of April 2021. It should be noted that our data collection methodology had no conscious preference towards more highly cited papers, leaving this dataset unbiased with this regard. This information is distilled into three values of interest below in Table 2.
Table 2. General citation statistics for our dataset.

| Total Citations for the Entire Dataset | 2729 |
|--------------------------------------|------|
| Average ‘citations per year’ for the collected papers | 5.5 |
| \[\frac{\sum_{i=1}^{143} \text{citations per year of paper } i}{143}\] |
| Most citations for a single paper per year | 30.7 |

The total number of citations (2729) for all 143 papers can imply the validity of our dataset. Since papers published more recently have less chance to be cited, we normalized the citation data by considering the number of ‘citations per year’ for each paper as a relatively fair baseline to compare. The average of 5.5 ‘citations per year’ for papers can imply a positive perspective in this research field. The most cited paper per year has earned 92 citations since 2018 and was published in a ‘Premier’ journal [31]. An interesting result observed in the top 10% of most cited papers per year is that out of 14 papers, 5 are published in ‘Premier’ journals, six in ‘Major’, and three in ‘Other’.

3. The Theory of Shop Scheduling Problems

Shop Scheduling problems (also known as manufacturing system scheduling) are well-known optimization problems in the field of operations research [32,33]. Understanding the concepts and the characteristics of manufacturing systems is essential to efficiently model the associated scheduling problems. Thus, in this section, we will review and classify different manufacturing systems and scheduling problems.

In a manufacturing system, there might be multiple jobs that should be processed by multiple machines. Processing a job by a machine is called an operation or task. The order in which machines process jobs is known as the machines’ schedule [34–36]. Finding an appropriate machines’ schedule which results in desired values for performance measures is called as Scheduling Problem [27,34,37]. Ideally, an optimal performance measure is reached through the optimal schedule. Almost all shop scheduling problems are NP-hard. There are a few simple versions of flow shop scheduling problems that can be solved optimally in \(O(n \log n)\) where \(n\) is the number of operations.

The common terminology in the field of shop scheduling problem is listed in Table 3. Some of the terms refer to constraints or specific situations. Depends on the problem definitions/requirements, experts may incorporate some of these items into the problem.

3.1. Objectives of Scheduling Problems

Different objective functions or performance measures are used in the scheduling problems. Makespan is the most common and the popular measure one which refers to the completion time of all operations of all jobs in the systems.

Table 4 shows the list of commonly used performance measures followed by their definition.

3.2. Shop Layouts

A very basic layout of all manufacturing systems includes a finite number of jobs, machines where \(J\) and \(M\) denotes the set of jobs and the set of machines, respectively; and \(O\) is the set of operations/tasks. A route (also known as the sequence) refers to the order of machines that a job should passes through [34,38,39]. Manufacturing systems and their layouts may be distinguished by their uniformity, existence of fixed routes, the flexibility of machines, the number of machines, and existence of buffer, set up times, etc. These differences come from variations in demand and required levels of production flexibility [40].
Table 3. Common terminology in Manufacturing system and shop scheduling problems.

| Term               | Description                                                                 |
|--------------------|-----------------------------------------------------------------------------|
| Buffer             | A waiting space for jobs between machines. Buffers avoids blocking situation |
| Blocking           | This situation occurs when a machine cannot release a job due to lack of buffer. It halts further tasks until the buffer is restored |
| Start Time ($S_{ij}$) | The time when job $j$ starts being processed by machine $i$. |
| Processing Time ($P_{ij}$) | is the time it takes for machine $i$ to process job $j$ |
| Travel Time       | The time for a job to be transferred from one machine to another               |
| Setup Time        | The time required to prepare a system or machine for production               |
| Bottleneck        | The process that directly limits production capacity, i.e., the critical path. |
| Dispatching       | Assigning resources to tasks, i.e., scheduling.                              |
| Assembly          | A product of combined components.                                            |
| Subassembly       | A component that is built by combining several other components and used in a larger product. |
| Precedence        | Job $j$ can only start processing after jobs in set $s$ have been completed. |
| Machine Eligibility | Machine $i$ can only perform some subset of job $j$’s required processes. |
| Machine Breakdown | Machines have non-continuous availability.                                  |
| Routing           | Job $j$’s operations must be completed in a specific order.                  |
| Incompatible jobs | Incompatible jobs cannot be processed by the same machine                    |
| Waiting Time      | This situation occurs when the buffer between machines is too small blocking may occur, so, upstream machines cannot release jobs. |
| No-Wait           | Consecutive operations of a job must be performed without wait-time on either machine. |
| No-Idle           | Machines must continuously process jobs without overlap or pause.           |
| Deadlock          | This situation occurs when the order of operations creates a cycle, and the entire process stops under this situation. |
| Preemption        | Jobs can be interrupted while being processed on machines. They can then be resumed on the same machine or a new one. |

Table 4. Commonly used objective (performance measure) in different manufacturing systems.

| Objective                  | Description                                                                 |
|----------------------------|----------------------------------------------------------------------------|
| Makespan                   | Total completion time: the time taken to complete all jobs.                |
| Total workload of Machines | The total processing time across all machines.                             |
| Workload of most loaded machine | The machine(s) with the largest processing time(s).                      |
| Max lateness               | The largest difference between a given job’s completion time and deadline. |
| Mean flow time             | The average amount of time that a given job spends on the shop floor.      |
| Tardiness                  | The difference between a given job’s completion time and deadline.         |
| Total tardiness            | The sum of each job’s tardiness.                                           |
| Mean tardiness             | The average of all jobs’ tardiness.                                        |
| Weighted tardiness         | The difference between a given job’s completion time and deadline multiplied by a weight (corresponding to cost). |

Baker [41] was a pioneer proposing a classification of classical manufacturing systems and their associate scheduling problems into the single-stage production system and multi-stage systems. In his classification, single-machine and parallel-machine are the
single-stage layouts, and job shop, flow shop, and open shop systems are the classical multi-stage layouts.

We utilized classical classifications and modern models as well as different scholars’ literature review studies to propose our comprehensive classification of system layouts as it is represented in Figure 5. In addition to the classical layouts, we incorporated the modern complicated layouts such as mixed shop and just-in-time systems in our classification. These modern system layouts emerged during the recent decades to satisfy the needs of fast-growing manufacturing technology.

### Figure 5. Shop layout classification.

In this section, the main properties of each layout are briefly explained. Reviewing this background is particularly important to understand how different types of graphs can be applied to address scheduling problems in different manufacturing structures.

#### 3.2.1. Single-Stage System

The simplest manufacturing layout is the **single-stage system** where there is only one work-center and a set of identical jobs passing through this work-center. The work-center may contain only a single machine or multiple machines.

A scheduling problem for a single machine system may be defined when all jobs are not available at the beginning of the process but instead jobs are given release dates or alternatively, a distinct due window is defined for each job [42]. A scheduling problem for a single machine system may be defined when all jobs are not available at the beginning of the process but instead jobs are given release dates or alternatively, a distinct due window is defined for each job [42].

When there are multiple machines in the single stage system, all machines should be arranged in parallel fashion. The multiple parallel machines can be either identical machines, or similar machines with different speeds (uniform parallel machines), or unrelated parallel machines [43–45]. Many single stage scheduling problem can be solved in polynomial-time [43].

#### 3.2.2. Job Shop System

**Job shops** (JS) are well-known manufacturing systems built for high flexibility and a wide range of applications. Machines within a classic job shop are fixed, unique, meaning they only have the capability of processing one type of operation, and may only process one operation at a time. All machines are available at time zero. Each job is independent of another and can follow a unique path through the machines to fulfill all operations,
meaning routes of different jobs are not uniform. All jobs are available after release dates and operations cannot be interrupted.

The job shop scheduling problem (JSSP) is a frequent topic of research due to its NP hardness [46]. To model systems resembling realistic shop floors, the classic JSSP has been widely expanded upon with the additions of constraints such as blocking, introducing a zero-buffer capacity, [47–49], no-wait, and no-idle. Classic JSSP has also been expanded to account for distributed work centers, creating the distributed JSSP [50]. To model systems resembling realistic shop floors, the classic JSSP has been widely expanded upon with the additions of constraints such as blocking, introducing a zero buffer capacity, [47–49], and no-wait, meaning consecutive operations of a job must be performed without wait-time on either machine, [51]. Classic JSSP has also been expanded to account for distributed work centers, creating the distributed JSSP [50]. To model systems resembling realistic shop floors, the classic JSSP has been widely expanded upon with the additions of constraints such as blocking, introducing a zero buffer capacity, [47,48].

Machines in a flexible job shop (FJS) may have the capability of doing more than one type of operation. The classic JSSP can be expanded to include this new attribute, creating the flexible JSSP [52]. While increasing flexibility, the inclusion of this attribute increases the complexity as well [53]. Flexible JSSP must now include the assignment of operations to capable machines as the first step, compared to previously classic JSSP assumed a predefined route for each job and a single option of machine to complete a given operation [54]. After that assignment, then flexible JSSP works similarly to JSSP such that a feasible and ideally optimal schedule is found based on predetermined objectives [19].

Flexible JSSP may come in many names. Such that JSSP with multi-purpose machines, [56], multi-processor JSSP [57], and JSSP with unrelated parallel machines [58].

3.2.3. Flexible Manufacturing System (FMS)

A flexible manufacturing system (FMS) is a relatively new technology which refers to an automated hybrid of flexible job shop [59] which utilizes an automated material handling system to maneuver jobs between machines [60]. All machine and material handling systems are monitored and controlled by a centralized computer [61]. The FMS scheduling problem resembles JSSP with some extensions including the movements of the material handling system and its common constraints of blocking and holdup [59].

Common objectives within the FMS scheduling problem include makespan, mean flowtime, and maximum flowtime [60]. Often additional objectives will be included regarding the travel time within the material handling system [62].

3.2.4. Flow Shop System

At the opposite end of flexibility, flow shop (FS) is another well-known manufacturing system. FS features a number of machines; each machine performs a single process on jobs that flow through in a unidirectional fashion [63].

Note that FS do possess a degree of flexibility as it can process different jobs simultaneously (although machines can only perform one task on one job at a time) and jobs are not required to be processed by each machine in the sequence [63]. Additionally, jobs may be allowed to have different completion times, instead of strictly being uniform [63].

The flow shop scheduling problem (FSSP) often looks to minimize makespan, total flowtime, tardiness, idle time, and other performance measures to ideally find an optimal schedule [64]. While not NP-hard by default, FSSP can become NP hard with a sufficient number of constraints, batch size, and number of machines [65].

The simplest and the most common version of FS is when the order of machines for all jobs are predetermined and fixed, and the modifiable element is the schedule of
jobs on machines [66]. The flow line (also known as assembly line) is a subcategory of FS with less flexibility and more effectiveness and speed (due to its simplicity). Unlike FS, flow line requires all jobs to be identical [67]. Added complexity to a flow line scheduling problem is often variation in processing time of machines. Understanding the structure of the flow lines is particularly important since some scheduling problems in literature are defined as hybrid layouts featuring sections of flow lines. For example, an assembly-type FS is a hybrid production system featuring production areas arranged as a JS to produce component parts. These components or subassemblies are fed into a flow line arranged as a FS for final assembly operations [68,69].

Within FS, jobs often follow similar routes or are identical, as in flow line. When jobs are processed in the same order on every machine, meaning their routes are identical, a permutation schedule is developed [70,71]. A FS in which this type of schedule occurs is known as permutation flow shop. The processing times on a machine may vary from job to job.

The flexible flow shop (FFS) consists of multiple stages. One or more parallel machines are available in each stage, but at least one of these stages includes multiple machines [45]. This system allows jobs to “skip” stages if they are not necessary [72,73].

3.2.5. Open Shop

The layout of open shop systems is like job shop systems, i.e., a set of jobs should be processed by a set of machines. However, the order of machines that each job should pass through or the order in which the processing steps of a job take place is arbitrary and can alternate freely. This important feature allows a greater number of possible job flow permutations leading it to be significantly harder from a computational standpoint [74].

The goal of the open shop scheduling problem is to find an appropriate order of assignment of jobs to machines to optimize some performance measures (e.g., makespan). In this solution, a job cannot be assigned to two machines at the same time; and a machine cannot process two jobs at the same time [75–77].

3.2.6. Mixed Shop

A mixed shop system (in some cases known as hybrid shop [19]) can include any combination of classical shop system such as open shop and job/flow shop [78]. Within mixed shop system, some jobs have predefined routes, like job shop or flow shop, while some jobs do not, like open shop [78]. Also, an operation for a job might need a set of machines instead of a single machine. This set of machines must process simultaneously and in sequence fashion [19].

Mixed shop is well known for its capability to resemble real-world complex systems. Custom or semi-custom manufacturing systems widely use mixed shop structure since some operations are very complicated and might require multiple simultaneous processing steps (by either operators or machines) [19,79].

The complexity of the mixed shop scheduling problem depends heavily on whether the operations per job is unlimited or not [78].

3.2.7. Cellular Manufacturing System (CMS)

The cellular manufacturing systems (CMS) have been introduced as an efficient layout in the philosophy of just-in-time and lean manufacturing which concerned with optimizing processes for time and value. To meet increasing demand for mid-volume and mid-diversity product mixes, CMS is designed as a hybrid of job shops and flow lines. CMS have cells containing grouped machines (or operators, workstations, etc.) that process jobs [80,81]. Jobs are grouped in part-families and ideally do not leave their cell [80].

The scheduling of cellular manufacturing is often referred to as flow shop group scheduling [82] and sometimes distinguishes inter-cell scheduling from intra-cell scheduling [80,83]. The scheduling problem includes cell formation, assigning machines to each cell, and general task scheduling [84].
4. Classifications and Solutions of Scheduling Problems

Scheduling consists of allocating given tasks to given resources over a period of time [85]. The theory behind scheduling is to create an efficient allocation and has many applications across industries [86]. Specifically, efficient production scheduling is key to manufacturing performance and productivity [87]. Once manufacturing context has been set, scheduling solutions can be found in many ways.

4.1. Classification of Scheduling Problems

Lin, Gen [88] proposed a popular classification of scheduling problems based on their processing characteristics as represented in Figure 6.

![Figure 6. Lin, Gen [88] classification of scheduling problems based on their processing characteristics.](image)

Given the scope of this review paper, we are more interested in the wider classes of the scheduling problem including static and dynamic scheduling.

Static scheduling consists of a deterministic environment in which jobs arrive in a predetermined manner, while dynamic scheduling consists of a stochastic environment [89]. Dynamic scheduling relates better to environmental changes [88].

4.2. Methods to Solve Scheduling Problems

Researchers early on focused on simple cases and their optimal schedules developed by rules and exact methods [19]. Later scheduling problems began being classified as polynomial solvable or NP-hard, in which computation time increases exponentially with respect to problem size [19]. With the increased focus on NP-hard problems, approximate methods became more popular.

In Figure 7, we propose a classification of methods used to solve scheduling problems based on the accuracy of their solutions [90].

![Figure 7. Classification of methods used to solve scheduling problems.](image)
4.2.1. Exact Methods

Exact methods find the optimal solution once the problem has been solved [89]. Many exact methods are enumerative such that they explore so many solutions in a set number of iterations. This means that solving large-scale problems by using the exact method cannot be computationally feasible [91]. Due to the complexity of the real-world manufacturing systems, the recent papers dominantly focus applications of non-exact approaches. In our dataset, only 21% of the papers used exact method.

Some of the most applied exact methods to solve shop scheduling problem includes Linear and Dynamic Programming [92], mixed integer programming [93], and the Branch and X family (e.g., Branch and Bound algorithm, Branch and Price algorithm, Branch and Cut algorithm), etc., [94]. Integer Programming and Branch and Bound are particularly popular in the field of scheduling problems as we also observed in our dataset.

*Integer programming and linear programming* (LP) are simple and efficient optimization techniques which require the linear objective function and constraints. Linear programs can equivalently formulate dynamic programs [95]. The variables in integer programming are required to be integers and are sometimes constrained to binary.

*Branch and bound* (B&B) is also particularly popular in the field of scheduling problems. B&B (such as other Branch and X methods) decompose the problem into small disjoint subproblems to partition the solution space and generating a new lower bound for optimal cost [70]. It begins with an empty partial schedule and then unscheduled jobs are added to the top of the partial schedule, known as *forward branching* [70]. This eventually leads to multiple potential solutions. More information about the structure developed within a branch and bound algorithm can be found in Section 5.12.1.

4.2.2. Approximate Methods

*Approximate methods* do not necessarily find the optimal solution and may generate different results when solving the problem additional times [96]. Instead, approximate methods find good solutions in practical periods of time (unlike exact methods being time-expensive). Approximate methods are effective particularly in a case of NP-hard problem and large-scales problems [96].

Approximate methods include constructive heuristics, improvement heuristics, and meta-heuristics. Constructive and improvement heuristics differ in their starting search state (an empty versus an assumed pre-existing solution) while meta-heuristics utilize different high-level strategies to search [97].

Constructive heuristics form the solution from scratch and literally step by step, such that in each iteration, they choose the best available option [91]. *The Nawaz, Enscore, Ham algorithm* (NEH) is a constructive heuristic method observed multiple times in our dataset. This method sorts jobs in descending sums of processing times and then generates a job sequence by evaluating that sorted order of jobs through minimizing makespan [98].

Local search [39, 99–101] and shifting bottleneck [102,103] are among Improvement heuristics which were repeatedly applied in our dataset.

Some of the more frequently occurring meta-heuristics from our dataset include genetic algorithms (GA), ant colony optimization algorithms (ACO), simulated annealing (SA), tabu search (TS), and particle swarm optimization algorithms (PSO). Genetic algorithms (GA) imitate the evolutionary “survival of the fittest” behavior [89]. All possible solutions are represented as chromosomes, and chromosomes make up generations in which “survival of the fittest” is applied to eliminate suboptimal solutions and create a better next generation [89]. Eventually GA should reach a generation with a good or a close-to-optimal solution.

Ant colony optimization (ACO) method imitates the behavior of ants in their search for food using pheromones [97]. While randomly searching for food, artificial ants representing agents deposit pheromones along their path and subsequent ants choose paths with stronger pheromone concentrations [97], eventually leading to a good or a close-to-optimal solution.
Simulated annealing (SA) is a relatively simpler and older meta-heuristic. SA imitates the physical process of annealing, the cooling of a solid material and the subsequent arrangement of its particles, and uses the decrease of temperature in order to search for the optimal solution [97].

Tabu search (TS) extends the local search improvement method by incorporating other concepts to potentially find the close-to-optima such as best improvement (a method to move past local optima), tabu lists (limited memory to avoid cycles), and aspiration criteria (a work-around limited memory potentially locking out optimal solutions) in order to potentially find the close-to-optima [97].

Particle swarm optimization (PSO) imitates swarm intelligence by representing solutions as particles which then move around the state space toward theoretically better positions, until ideally converging to the global optima [44,51]. Particles move based on their own local neighborhood and based on the total findings within the swarm, reinforcing the idea of communication [104,105].

Algorithm 1  Solving a generic scheduling problem by an approximate method

1: Start with a random schedule or use a constructive method to generate a base schedule.
2: Calculate the performance measure
3: While a good solution (ideally, the optimal solution) has not been reached:
4: Perturb the schedule
5: Recalculate the performance measure

5. Theory and the Application of Different Types of Graphs

In this section, we discuss the theory of the most popular graph-based models utilized in different shop scheduling problems. It should be noted that general speaking there is a nested relation between the graph models and their equivalent matrix-based models. In some cases, matrix calculations are incorporated in the application of the graph-based model [106]. However, we kept the focus of this paper on the graph aspects.

Graphs model the relationships between objects rather than the objects themselves [22]. Two main elements of a graph are nodes and edges. Nodes (also known as vertices or points) represent the objects and edges (also known as links or arcs) show the relationship between the objects by connecting them. If a pair of nodes are connected by an edge, then these nodes are called as adjacent nodes. Edges can be directed or undirected, dictating the symmetry of the relationship between two nodes. Either edges or nodes in a graph can be weighted depending on the requirements of the system represented by the graph.

Let $N$ and $E$ be the set of nodes and edges in a given graph, then the graph can be denoted by $G(N,E)$. A series of connected nodes create a path and if there is a path from node $n \in N$ to $m \in N$, then $m$ is reachable from $n$. Loop or cycle refers to a cyclic path such that all nodes on this cyclic path are reachable from themselves.

Figure 8 shows a comprehensive categorization over the most common graph-based models used in shop scheduling problems. The macro categorization is based on the existence of directed and undirected edges in different graph types. The third category entitled “Un/Directed” refers to those graphs which can potentially contain either directed or undirected edges, or those graphs which contains both directed and undirected at the same time.

In each subsection, after we introduce the concept, we will describe the application of the graph-based model to shop scheduling problems.
5.1. Clique

Clique is one of the fundamental concepts of graph theory [28]. To define clique, first we need to introduce a few concepts. Let C be a subgraph in an undirected graph G. Subgraph C is complete if each node within C is connected to every other node. C is maximal if C is a subset of G and is not contained in any other complete graph within G. Finally, C is a clique if it is a complete maximal subgraph of G [107].

The maximum clique is the clique with the most nodes. The clique number, $\omega(G)$, dictates the number of nodes in the maximum clique within G [108].

Consider a random undirected graph in Figure 9. The maximum clique contains nodes \{3,4,5,6\} which is a 4-vertex clique. This clique is called a clique of size 4. Moreover, \{3,4,5\}, \{3,5,6\} are cliques of size 3 (3-vertex).

The application of cliques to manufacturing/production related problems is mainly for partitioning [109] and simplifying classification [110]. We are particularly interested in its applications to shop scheduling problems where the concept of clique usually is coupled with graph coloring, agreement graph, and the disjunctive graph which are explained respectively in Sections 5.2, 5.4 and 5.14.1.

5.2. Graph Coloring

Graph coloring assigns colors (or integer values) to nodes, and generally subject to the following constraint: no two adjacent nodes share the same color [28]. The colored graph is then optimized to reduce the number of colors applied [28].
This general definition of the graph coloring reveals the close connection between cliques and graph coloring, i.e., if $G$ contains a clique of size $k$, then at least $k$ colors are needed to color that clique. For instance, in Figure 9, at least four colors are needed for the clique of size 4. However, nodes 1 and 2 can have the same color of either nodes 4, 5, or 6.

Color assignment can become more complex by encoding a set of colors rather than just a single color, known as multi-coloring [111]. With this, the sets of colors of two adjacent nodes must be disjoint [111].

Graph coloring is commonly used to model incompatible jobs which cannot be processed by the same machine [112]. Given an undirected graph $G = (V, E)$, where $V$ represents the set of jobs (nodes) and $E$ is a set of edges between incompatible jobs [113]. Colors $c$ are unique time units within the scheduling horizon and are assigned to a node if that corresponding job was processed at all during that time unit $c$ [113]. The number of colors $c$ are limited so as to not color each node uniquely, or equivalently processing each job one at a time. Another important concept is equitably $c$-colorable defined for a graph whose nodes can be divided into $c$ independent sets [114].

Graph coloring can also be used for timetabling [115] and other potential constraints in shop scheduling problems. Graph coloring is conducive for exact and approximate methods to be applied [113].

5.3. Social Network (Collaborative Network)

Social networks, also known as a collaborative network, come from sociology and focus on how the interactions (edges) between actors (nodes) affect the collective, or the entire system [116]. Within this network, centrality measures are defined to focus on the most prominent actors (who are involved the most within the network, having the most incoming/outgoing edges) and their engagement in the network [116].

Recently, the social network has been utilized to model manufacturing systems as well. Social networks require a particular form of dataset as the input to be created. These datasets are summarized by the Affiliation matrix which is a symmetric Binary (Boolean) matrix where columns and rows shows the full list of all jobs, machines, and operations.

Reddy, Ratnam [116] represented a benchmark flexible job shop scheduling problem with 10 jobs and 6 machines using Affiliation atrix and collaborative network as in Figure 10. After modeling the problem, Reddy, Ratnam [116] identified key machines using centrality measures. This method creates descriptive statistics early on, helping to determine how sensitive the system may be when manipulating actors. After manipulation, metaheuristics can be re-run to conduct an accurate sensitivity analysis on key machines.

![Figure 10. Collaborative network for the dataset 10 by 6 [116].](image)

5.4. Agreement Graph and Bipartite Graph

An agreement graph is an undirect graph $G = (V, E)$ where $V$ is the set of jobs (nodes) and $E$ is a set of edges connecting a pair of agreeing jobs. Agreeing jobs are jobs that can be scheduled simultaneously on separate machines [117]. The agreement graph can be very useful when resources are non-sharable [117].
A bipartite graph, also known as a bigraph, is an undirected graph consisting of two disjoint independent sets of nodes, $A$ and $B$, where $V = A \cup B$. Every edge connects a node in $U$ to a node in $V$. Agreement graphs often are bipartite graphs since only pairs of agreeing jobs are graphed. Figure 11 shows a simple example of an agreement graph.

![Figure 11. An agreement graph, $G = (A \cup B, E)$, that is also a bipartite graph in which agreeing jobs are connected by edges.](image)

A tangible example of school exam planning can help the understanding of the concept of agreeing nodes (jobs) in the agreement graph of a shop scheduling problem [117, 118]: suppose $n$ exams need to be scheduled, equivalent to $n$ jobs. Exams are scheduled to take place in $m$ classrooms, equivalent to $m$ machines, which can accommodate any exam. The objective is to minimize the examination period. The coinciding agreement graph for this situation would be an undirected graph, $G = (V, E)$ such that $V$ includes all of the jobs, $J_i$ (exams), and $E$ includes edges between jobs $J_i$ and $J_j$ if there are no students who take both exams (corresponding to those jobs) at the same time [117].

5.5. Markov Random Fields

Markov Random Fields (MRFs) are probabilistic graphical models and categorized as undirected graphs [119]. MRFs are used to model the existing correlation between the decision variables. Simply, MRF describes the dependency relationship of the variables. Nodes represent variables and edges represent the dependency between variables. That dependency can now be conditional according to the probabilities associated with each node.

In the shop scheduling context, the corresponding MRF model is defined as follows: Let $(G, \Phi)$ be a MRF where $G = (N, A)$ is the graph structure ($N$ is set of nodes and $A$ is set of arcs) and $\Phi$ is parameters of the MRF. Nodes denote operations and an undirected arc between two nodes represent the correlation relationship between the two nodes.

Sun, Lin [119] used MRF to solve a real-world stochastic flexible job shop scheduling problem with the objective to minimize the expectation and variance of makespan. Unlike the traditional problem, in this stochastic scheduling problem the processing time of each operation is a stochastic value and not predefined. The concept of maximal clique is utilized to solve the stochastic problem which is modeled by MRF [119].

Despite the importance of stochastic nature of real-world manufacturing systems/processes, only one reference has been found (in our dataset) using MRF. We believe that shop scheduling field can clearly benefit from this graph-based approach in future. In addition to MRF, Bayesian network is another great probabilistic graph-based model that can potentially be utilized in shop scheduling problems.

5.6. Traveling Salesman Graph

The traveling salesman graph is a complete undirected graph with weighted edges where nodes represent cities and edges represent paths between cities such that weights of edge dictate distance [120]. Weights may also be assigned to nodes depending on the problem. A circuit that reaches each node once (and only once) is known as a tour and the length of a tour is the summation of weight of traversed edges. The objective of the
problem associated with the traveling salesman graph is to find the optimal tour with the shortest length [120].

Traveling salesman graph is used to model the cyclic shop scheduling problem [121] which is a less common scheduling problem. Cyclic shop scheduling problem focuses on batches of products being produced over a fixed interval of time which is known as cycle time [77,121]. Cyclic scheduling problem looks to minimize cycle time [122].

Within a manufacturing context, the cyclic shop scheduling problem may have n jobs and m machines. A corresponding travel salesman graph can be defined for kth machine in the cyclic problem by \( H_k(V,E,p,s) \) in which \( V \) represent jobs with weights corresponding to Operation Time (\( OP \)) and edges \( E \) represent the sequence of jobs with weights corresponding to Setup Times (\( ST \)). Then, the problem is formally defined, and the optimal schedule is equivalent to the optimal tour within a traveling salesman graph [121]. Figure 12 illustrates a sample of traveling salesman graph \( H_k \).

![Figure 12. A partial graph \( H_k \) with parametric weights for nodes and edges.](image)

5.7. Directed Acyclic Graph (DAG)

The Directed Acyclic graph (DAG) is the most important type of directed graph in the field of modeling the manufacturing systems. DAG can also explain some foundations for other types of directed graphs used in shop scheduling problems. Thus, we elaborated on some details of DAG and its mathematical background.

DAG is usually represented by the notation of \( G(N,E) \) where \( N \) is the set of nodes and \( E \) is the set of directed edges in \( G \). Either nodes or edges can have weights. However, following the majority of references in our dataset, we considered weights on nodes in this paper.

Each node \( n \) has a set of predecessor and successor nodes which are connected to \( n \) through incoming edges and outgoing edges, respectively. A first node does not have any predecessors and a last node does not have any successors. Note that there might be multiple first and last nodes in a DAG. The length of the path to node \( n \) is calculated by the maximum of the summation of weights of nodes on the path from all first nodes to \( n \) [123]. The maximum of lengths of all paths from a first node to a last node is the length of the longest path, or the critical path which is an important measure in DAG calculation. The length of the longest path generally implies a critical performance measure in the corresponding system modeled by a DAG. It should be noted that the longest path can be defined only in directed graphs.

To find the length of the longest path, first the DAG should be topologically sorted, then, the length of the longest path to each node should be calculated using dynamic programing [34]. In the following subsection these two steps will be explained.

5.7.1. Topologically Sorting a DAG

To arrange nodes in a DAG, topological sorting is commonly done which converts nodes into linear order such that edges all point in one direction [124]. Multiple topological sorts can be defined for a DAG.

Kahn [125] proposed the most efficient algorithm known to find the topological sort in a DAG. A pseudo code of this Kahn’s algorithm is represented in Algorithm 2. Removing nodes from the dynamic set of \( Q \) (Step 4) is arbitrary. Different orders of removal result in a new, valid topological sort.
Algorithm 2: Topological Sort Algorithm

Output $T$, the set of nodes which are topologically sorted
1. Count the number of predecessors for all nodes using a counter $\Pi$
2. Insert all the first node with no predecessor in the dynamic set $Q$
3. While $Q$ is not empty:
   4. Remove a node from $Q$
   5. Place the node in the topological set ($T$)
   6. For all successors:
      7. Decrement $\Pi$ of the successor
      8. If $\Pi = 0$
         9. Insert the successor in $Q$

5.7.2. Calculating the Longest Path in a DAG

After topologically sorting the DAG, the dynamic programming algorithm finds the length of the longest path. Dynamic programming (also, known as dynamic optimization) is a popular mathematical optimization technique. It breaks down the problems into smaller sub-problems and try to find the optimal solution for the sub-problems to eventually solve the original problem. By applying dynamic programing methods, larger and more complex problems can be solved.

In DAG, an algorithm based on dynamic programming algorithm is proposed to find the longest path, $L_{max}$, by calculating the path of all nodes in topological order (Algorithm 3). In Algorithm 3, $l_i$ denotes the length of the longest path from the first node to node $x$.

Note that $T$ in Algorithm 3 is the output of Algorithm 2, and the data structure of $T$ is predefined as a sorted set. Therefore, in step 2 of Algorithm 3, nodes are removed from $T$ based on the topological order (the order of set $T$).

Algorithm 3: Finding the length of the longest path using Dynamic Programing Algorithm

Output $L_{max}$ the length of the longest path
1. For all nodes $x$ on the topological sort ($T$):
2. Remove $x$ from $T$
3. $l_x = \max_{n \in \text{pred}(x)} \left( l_n \right) + w_x$
4. $L_{max} = \max_{i \in L} \left( l_i \right)$

5.7.3. DAGs for Manufacturing Systems

If a DAG is used to model a manufacturing system, then it may be known as a task graph or precedence graph which will be explained in detail in Section 5.8 [126]. Within a manufacturing context, a manufacturing DAG’s (MDAG) can be defined where nodes represent operations with weights showing processing time. Also, edges represent the precedence of operations.

Ashour and Parker (1971) were among the pioneers applying a DAG/precedence graph to shop scheduling problem. They defined simple rules to model a system, i.e., horizontal edges represent the predefined route of a job, while vertical edges represent the schedule of a machine. Then, the length of the critical path is equivalent to the makespan, translating a manufacturing scheduling problem into a longest path search problem [127].

Figure 13 represents a job shop system and its topologically sorted version. Furthermore, Figure 14 depicts the flow shop system. In both examples, there are 4 jobs (J1 to J4) that should be processed by four machines (M1 to M4). The difference between the routings of flow shop and job shop system can be easily observed in these two figures. This confirms the powerfulness and simplicity of DAG as an appropriate graph-based models.
5.8. Precedence Graph (Conflict Graph)

A precedence graph, also known as conflict graph, can be used to model a situation with constraints related to the concurrency of events. Two events that cannot be processed simultaneously are in conflict and therefore need a precedence relation to be defined [128]. The concept of a precedence graph has been visualized in several ways during its existence. In most references in our dataset, the precedence graphs are visualized as a DAG. 

Birgin, Ferreira [129] proposed an efficient precedence graph in the flexible job shop scheduling problem. They used a similar approach to MDAG explained in Section 5.7.3. But they added a sequencing flexibility to the model. Also, they represented the precedence constraints of a single job as in Figure 15. This representation of precedence constraints seems to work well with industrial environments including the printing industry [99,130].

Although precedence graphs are mainly categorized as directed graphs, Tellache and Boudhar [131] defined an undirected graph as a precedence graph to model a flow shop scheduling problem. They discussed an interesting logical relevancy between their precedence graph and an associated agreement graph (which explained in Section 5.4) [117]. Instead of connecting agreeing jobs (nodes), this proposed precedence graph connects conflicting jobs [131]. Therefore, solving the FS problem with conflict graph is equivalent to solving FS problem with agreement graph.
5.9. Petri Net

Petri nets are directed bipartite graphs in which nodes are either classified as places (generally represented as circles) or transitions (generally represented as rectangles). Recalled from Section 5.4, in bipartite graphs, edges should connect nodes across the two disjoint groups. Edges pointing from a place to a transition are known as input places of a transition, and edges pointing from a transition to a place are known as output places of a transition. Places may acquire any number of tokens (depicted by dark filled circles as in Figure 16). If all input places connected to a transition have at least one token, then the transition is enabled.

![Figure 16](image_url)  
**Figure 16.** A petri net representing a manufacturing system with a job and 2 machine (M1 and M2) such that the job can be processed by either M1 or M2.

Once the transition is enabled and fires, it consumes the input places' tokens and produces tokens in output places [62]. If more than one transitions are simultaneously enabled, then the transitions can fire in any order. Thus, firing of transitions and executing the petri nets are nondeterministic (unless an execution policy is defined). This means, systems with concurrent behavior, such as some manufacturing systems, can benefit from being modeled as a petri net.

Baccelli, Cohen [132] proposed an application of a particular type of petri net called as timed event graph to choice-free manufacturing systems. In timed event graphs, a node can have only one incoming and outgoing edge, while timing feature can be considered [133,134]. Moreover, timed event graph has been applied to model the job shop scheduling problem [135,136].

Figure 16 shows an example of a manufacturing system modeled by a petri net. This model includes six states denoted by places (circles), and six transitions which represent the changes between these states (rectangular). The states can be defined as follows: the process has not started, the process is being done by either machine 1 or machine 2, and the process is finished.

5.10. Alternative Graph

An alternative graph is a directed graph and denoted by \(G(N,F \cup A)\) where \(N\) is the set of nodes and \(F \cup A\) contains all edges. The edges in set \(F\) represents precedence relations and are usually visualized by the solid directed weighted edges. On the other hand, the edges in set \(A\) are alternative paths and visualized by doted/dashed edges [47,137]. Alternative edges can be weighted or not dictating whether swapping is allowed or not. In the context of shop scheduling problem, swapping refers to the ability for jobs to swap machines to avoid deadlock [48].

An alternative graph is a generalization of a disjunctive graph (described in Section 5.14.1) in which undirected edges represent the alternative path. Based on this, the disjunctive graph is categorized as a mixed graph, while the alternative graph is classified as a directed graph.

The alternative graph can be useful to model the scheduling problem with blocking situation (described in Table 3) [47,48,138,139]. In the corresponding alternative graph of a manufacturing system considering the blocking situation, nodes represent the op-
operations. Edges in $F$ showcase the precedence relation between consecutive operations, while edges in $A$ showcase an alternative processing order for operations that cannot occur concurrently [48].

Figure 17 is an example for an alternative graph representing a partial manufacturing system with blocking situation. In this example, there are two operations $i$ and $j$ that cannot be done on the same time, i.e., these operations are processed by the same machine [47]. Also, $\sigma_i$ and $\sigma_j$ denote the operations associated with the blocking operations. Solid edges with weights show that $i$ and $j$ should be processed before $\sigma_i$ and $\sigma_j$, respectively. Since $i$ and $j$ that cannot be done concurrently, dashed edges show alternative paths for them. Eventually, one of these alternatives will be selected.

![Figure 17](image)

**Figure 17.** The alternative graph depicts a partial system with two operations $i$ and $j$ that cannot be processed at the same time.

5.11. Multi-Stage Graph

A multi-stage graph is a directed graph where nodes are grouped into stages and edges connect nodes from one stage to the next stage [140]. In a multi-stage graph, the goal is to find the shortest path from the first node to the last node, so, sometimes it is called as the shortest path problem.

The multi-stage graph has been used in several multi-stage shop scheduling problems such as multi-stage flow shop systems discussed in Section 3.2.4 [141,142], and classic open shop and job shop scheduling problems [36]. Within these applications, the combinations of the shop scheduling problems and the shortest path problem are studied using multi-stage graphs [142].

This so-called combination problem can be formally defined as follows: Let $G = (V,E)$ be a directed graph with two distinguished vertices $s,t \in V$ as a start and finish points which represents a manufacturing system with $m$ machines and $p$ jobs. Also, edges $j \in E$ correspond to jobs $j_i \in J$, $(j = 1, 2, \ldots , p)$ and the weights on edges represent the processing time of jobs on the machines $(p_{1j}, p_{2j}, \ldots , p_{mj})$. Then, the problem is to find a path from node $s$ to $t$ such that the appropriate schedule of jobs on machines yields the minimum makespan (or generally, with optimum performance measure) [36]. This is a combination of shop scheduling problem and the graph shortest path problem. It should be noted that the scheduling problem and the shortest path problem individually are two special cases of this so-called combination problem.

For instance, consider Figure 18A for a system with two machines and $n$ jobs. Since in this figure, there is only one unique path from $s$ to $t$, our problem is reduced to the two-machine shop scheduling problem. However, if we assumed that processing times on the second machine is zero, then, as shown in Figure 18B, then our problem is reduced to the shortest path problem [36].

![Figure 18](image)

**Figure 18.** Special cases of the combination problems using multi-stage graph. (A) a system with two machines. (B) the reduced problem to the shortest path problem.
5.12. Tree

A traditional tree is an undirected acyclic graph. However, a directed version of tree is also defined (it is called polytree). Nodes are connected by at most one edge. Adding any edge to a tree creates a cycle and deleting any edge in a tree makes the tree disconnected (having a node without any incoming or outgoing edge).

One of the applications of tree in the context of manufacturing systems is to build the critical tree based on the associated disjunctive graph (discussed in Section 5.14.1) representing a shop scheduling problem. A critical tree only considers the longest paths within a disjunctive graph, which allows for quick exploration of a potential schedule solution [143].

The rest of this section will discuss some of the most common types of trees and their applications in the scheduling problem that we could find in our dataset.

5.12.1. Branch and Bound Algorithm/Tree

As explained in Section 4.2.1, the branch and bound algorithm utilizes forward branching which appends unscheduled jobs to the head of the partial schedule developed. Throughout this process, a B&B tree is generated [144]. Nodes represent potential solutions and are branched such that their tree level corresponds to the number of jobs scheduled in that node [144].

5.12.2. Hyper-Heuristic Tree

A hyper-heuristic tree is used to select appropriate heuristics method or generate a new heuristics method which yields better solutions in a given search problem such as shop scheduling problem [145]. Genetic programming, a technique for breeding better solutions for a given problem [146], often goes hand-in-hand with hyper-heuristics. Trees are utilized in genetic programming hyper-heuristics to represent potential solutions (solutions being heuristics, not optimal schedules) [146]. Those trees are manipulated at the subtree level with the intent of breeding better solutions [35].

The hyper-heuristic tree is also used to solve a classical job shop scheduling problem by proposing a heuristic which consists of linear orders of dispatching rules: a tree structure is used to show each dispatching rule [147].

5.12.3. AND-OR Graph

An AND-OR Graph is a directed tree which utilizes some basic logic, including conjunction (and) and disjunction (or). AND/OR graph can provide a simple and visual understanding of a system, while it helps us to avoid unnecessary enumerating every alternative route of a given job in a scheduling problem [148,149].

AND/OR trees are typically made for each job, so nodes represent operations and directed edges present the predetermined route of a given job [150]. Logic is implemented, dictating whether the subsequent paths are all required (conjunction, represented by AND node) or either path is required (disjunction, represented by OR node) [148]. There are other methods for implementing logic, such as manipulating edges, which are also satisfactory.

One of the particular applications of AND-OR tree is to those problems where the shop scheduling problem is in a dynamic environment with alternative routes, such as flexible job shop [151], or with some kind of joining final assembly operation, such as flexible assembly job shop scheduling problem [148]. In such systems, each final product may be assembled from several subassemblies, and each subassembly may be assembled from several types of parts. Each part might go through linear or nonlinear process plans with operations on machines, as shown in an example in Figure 19.
AND/OR graph of machine operations of parts with alternative process plans [148].

Figure 19. (A) An example of a flexible assembly job shop scheduling problem with three parts, two subassemblies. (B) AND/OR graph of machine operations of parts with alternative process plans [148].

5.12.4. Game Tree

Game theory and game trees are constructed based on the relationship between decision makers/players [152]. A game tree represents the decisions of players by assigning layers of a game tree to each player. Within that layer, nodes represent the different positions a player can choose from [153]. Directed or undirected edges represent moves a player can do. These edges connect nodes from a lower layer to a node in a higher layer which indicates that the choice of one player affects the choice of the next players since players make decision sequentially [153]. A game tree which includes every possible position and move is seen as complete.

The game tree is an efficient tool to model multiple objectives optimization problems such as shop scheduling problems since it can handle the competition and conflicts between the objectives [152].

In this application, objectives can be considered as players. Each objective (player) makes decision/ selects a feasible schedule based on its individual goal (e.g., minimizing makespan, minimizing total workload). Then, using dynamic game theory, the equilibrium solutions are considered as the optimal results of the multi-objective scheduling problem [152].

5.12.5. Processing Operation Tree

The integrated scheduling problem is defined when processing and assembly are performed simultaneously. This problem could be considered as a subset of the traditional job-shop scheduling problem. This problem produces a processing operation tree, also known as a tree-structured product-integrated scheduling, which is a directed tree where nodes represent operations and directed edges represent dependent relationships between the operations [154]. A processing operation tree is made for each job, like AND/OR graphs.

Figure 20 showcases a sample for a processing operation tree for a single job \( j \) that should be processed by 3 machines (M1, M2, and M3), through 15 operations (O1 to O15) [154].
5.13. Semantic Graph

A semantic graph is a practical graph database and can be either directed or undirected. In a semantic graph, concepts and their relationships are represented by nodes and edges, respectively [155]. Unlike an ordinary graph in which edges can only represent the precedence relationships with possible quantitative weights, the edges in a semantic graph can also be assigned with semantic relations.

The distinguish advantage of a semantic graph is the ability to express qualitative and conceptual relationships compared to quantitative relationships. Semantic relations are encoded through a triple which refers to three entities relating to subject-predicate-object. Figure 21 shows an example for a triple.

![Figure 21. An example for a triple: ‘He needs money’](image)

Due to the special features of the semantic graph, it can integrate the entire lifecycle data of the system, and efficiently model many conceptual relationships that do exist in manufacturing systems beyond those represented numerically [155]. Examples of these relationships include stock constraints, sales orders, and other lifecycle data [155].

5.14. A Mixed Graph

Graphs which can contain both directed and undirected edges at the same timed are known as mixed graphs and they have some applications in different manufacturing systems. Disjunctive graphs and ant colony graphs are the most common mixed graphs to represent the shop scheduling problems. In this section, we will discuss these two models.

5.14.1. Disjunctive Graph

Disjunctive graph is one of the most used graph-based models in the field of shop scheduling. It was first proposed by Roy and Sussman, 1964 [92]. Disjunctive graphs can be compared to DAG when they are used to model a manufacturing system. Unlike DAG, in a disjunctive graph, the set of edges are represented by the union of two subsets of directed and undirected edges \( G = (V, C \cup D) \) [156]. The operations are represented by nodes \( (V) \), and the route and the schedule are represented by the set of directed edges \( (C) \) and undirected edges \( (D) \), respectively. This means that the edges in \( C \) connect the sequential operations of each job. However, the undirected edges \( (D) \) between the operations show unspecified order of operations on a machine.

This ordering of all operations processed on each machine should be determined to find a good (ideally the optimal) schedule for the system. This can be done by using the concept of a clique. If all dashed edges are oriented such that each clique associated to a machine becomes acyclic then a feasible schedule is achieved. Thus, a schedule is feasible if and only if it can be represented by a directed acyclic disjunctive graph. Once the schedule is deemed feasible, the longest path from the oriented graph will compute makespan, the objective to be minimized.

Figure 22 illustrates a simple example of disjunctive graph representing a job shop system with 3 jobs (J0, J1, J2) and 3 machines (M0, M1, M2). Nodes s and t are dummy operations (nodes) showing the star and the end of the process. The rest of the nodes represent the operations, and the label of each node consists of two items: the machine which processes the operation and the operation time. By considering the dashed edges, a feasible schedule is determined where the length of the longest path (bold edges) is the makespan, i.e., \( 0 + 4 + 4 + 2 + 2 + 2 = 12 \).
5.14.2. Ant Colony Graph

Ant colony graph or ants’ moving disjunctive graph [157] utilize the concept of disjunctive graphs to implement the ant colony optimization algorithm (detailed in Section 4.2.2) in shop scheduling problems.

In addition to all the same characteristics of a disjunctive graph, Ant colony graph include ants crawling from node to node across edges, leaving pheromones which increase the likelihood of taking that path again [157–159]. The ants resemble operations such that operations look for a machine to be processed on as quickly as possible just as ants look for food as quickly as possible when foraging [50]. The incorporation of dashed edges can be seen as an alternative path for searching ants [157]. The feasible solution is the directed path which visits every node, completing every operation [160].

6. Overview of the Full Dataset

According to the systematic categorization proposed in Sections 3–5, this section presents the full sorted list of 143 papers in our dataset, in addition to a comprehensive overview of dataset to provide a baseline for further comparison. Also, to get a sense of the most frequently cited papers in our dataset, the papers making up the top 10% of citations per year were chosen for additional analysis. Most of the statistics regarding these papers follow very closely to the trends of the overall dataset. However, in two cases of layout and scheduling type, a significant difference between the top subset and the entire dataset was observed and reported in this section accordingly.

In this section, appropriate abbreviations are defined to refer to the micro categories inside tables and charts as needed. Table 5 can be used as a legend for all tables and charts in this section.

Table 5. The legend for tables and charts in this section.

| Topic          | Abbreviation | Description                          |
|----------------|--------------|--------------------------------------|
| Layout         | FJS          | Flexible Job Shop                    |
|                | FFS          | Flexible Flow Shop                   |
|                | FMS          | Flexible Manufacturing System        |
|                | Perm         | Permutation Flow Shop                |
|                | CMS          | Cellular Manufacturing System        |
|                | Classic      | Classic Job Shop                     |
|                | Multi        | Multi-stage Flow Shop                |
|                | Open         | Open Shop                            |
|                | Mixed        | Mixed Shop                           |
|                | Parallel     | Parallel Machines (Single Stage)     |
| Scheduling     | S            | Static                               |
| Type           | D            | Dynamic                              |
| Objectives     | MS           | Makespan                             |
|                | T            | Tardiness related                    |
Tables 6–8 summarize the relevant properties of the collected references in our dataset based on the usage of directed graphs, undirected graphs, and un/directed graphs, respectively. All tables are chronologically sorted, and the macro categories are used as the table headings.

Table 5. Cont.

| Topic | Abbreviation | Description |
|-------|--------------|-------------|
| Solutions | LP | Linear Programming |
| | PSO | Particle Swarm Optimization |
| | SB | Shifting Bottleneck |
| | SA | Simulated Annealing |

Table 6. Overview of references in the dataset using directed graphs.

| Ref. No. | References (Listed Chronologically) | Type of Directed Graph | Layout | Problem & Solution Characteristics | Solution Method |
|----------|--------------------------------------|------------------------|--------|----------------------------------|-----------------|
| [124]    | (González et al., 2015)              | ✓                      | FJS    | S                                 | MS              | Other          | Exact  | Approximate | Case Study |
| [141]    | (Wang et al., 2015)                 | ✓                      | Multi  | S                                 | MS              | Other          | Exact  | Approximate | Bench      |
| [129]    | (Birgin et al., 2015)               | ✓                      | FJS    | S                                 | MS              | Other          | Exact  | Approximate | Bench      |
| [142]    | (Nip et al., 2015)                  | ✓                      | Multi  | S                                 | MS              | Other          |        |             |             |
| [69]     | (Wang & Wang, 2015)                 | ✓                      | Perm   | S                                 | MS              | Other          | Exact  | Approximate | Bench      |
| [161]    | ( Başak & Albayrak, 2015)           | ✓                      | FMS    | D                                 | Other           | RC             |        |             |             |
| [162]    | ( Baruwa & Piera, 2015)             | ✓                      | FMS    | S                                 | MS              | Other          |        |             | Bench      |
| [79]     | ( Wang et al., 2016)                | ✓                      | Mixed  | D                                 | Other           | ACO            |        |             | Bench      |
| [48]     | ( Pranzo & Pacciarelli, 2016)       | ✓                      | Classic| S                                 | MS              | Other          |        |             | Bench      |
| [36]     | ( Nip et al., 2016)                 | ✓                      | Classic| Open                             | S               | MS             | Other          |        |             |             |
| [163]    | ( Marin et al., 2016)               | ✓                      | FFS    | S                                 | MS              | Other          |        |             |             |
| [164]    | ( Al-Ahmari, 2016)                  | ✓                      | CMS    | S                                 | MS              | Other          |        |             | Num        |
| [62]     | ( Baruwa & Piera, 2016)             | ✓                      | FMS    | S                                 | MS, Other       | Other          |        |             | Bench      |
| [165]    | ( Madraki & Judd, 2018)             | ✓                      | Classic| S                                 | MS              | Other          |        |             | Num        |
| [131]    | ( Tellache & Boudhar, 2018)         | ✓                      | Multi  | S                                 | MS              | NEH, Other     |        |             | Num        |
| [122]    | ( Bozejko et al., 2017)             | ✓                      | Classic| S                                 | MS              | TS, Other      |        |             | Bench      |
| [166]    | ( Samarghandi & Behrouzi, 2017)      | ✓                      | Multi  | S                                 | MS              | LP             |        | Res         | Bench      |
### Table 6. Cont.

References Using DIRECTED GRAPHS (1/2)

| Ref. No. | References (Listed Chronologically) | Problem & Solution Characteristics | Type of Directed Graph | Layout | Scheduling Type | Objective | Solution Method | Case Study |
|----------|-------------------------------------|------------------------------------|------------------------|--------|-----------------|-----------|-----------------|-----------|
| [71]     | (Qian et al., 2017)                 |                                    |                        | Perm   | S               | MS        | LS, Other       | Num       |
| [167]    | (Channanlor et al., 2017)          |                                    |                        | FFS    | S               | MS        | LS, GA, ACO     | RC        |
| [168]    | (Yu et al., 2017)                  |                                    |                        | FJS    | S               | MS        | LP, LS, Other   | Num       |
| [169]    | (Mejia et al., 2017)               |                                    |                        | Open   | S               | MS        | Other           | Num       |
| [77]     | (Pempera & Sznitnicki, 2018)       |                                    |                        | Open   | S               | MS        | TS              | Bench     |
| [170]    | (Bozek & Werner, 2018)             |                                    |                        | FJS    | S               | MS        | LP, TS          | Bench     |
| [171]    | (Shokouhi, 2018)                   |                                    |                        | FJS    | S               | MS, Other | GA              | Num       |
| [139]    | (Gholami & Tornquist Krassmann, 2018) |                                |                        | Classic | D       | T        | LP              | Bench     |
| [173]    | (Madrazi & Judd, 2019)             |                                    |                        | Classic | S       | MS        | Other           | Num       |
| [140]    | (Corominas et al., 2019)           |                                    |                        | FJS,   | S               | T, Other  | LP              | Other     |
| [58]     | (Cao et al., 2019)                 |                                    |                        | Multi  | S               | MS, Other | Other           | Num       |
| [174]    | (Liao & Fu, 2019)                  |                                    |                        | Perm   | S               | MS, T     | GA              | Num       |
| [175]    | (Billaut et al., 2019)             |                                    |                        | Classic | S       | MS        | TS              | Bench     |
| [176]    | (Kress et al., 2019)               |                                    |                        | FJS    | S               | MS, T     | Other           | Bench     |
| [177]    | (Tian et al., 2019)                |                                    |                        | FJS    | D               | MS        | Other           | Bench     |
| [43]     | (Guan et al., 2019)                |                                    |                        | Parallel | S     | MS        | ACO             | Bench     |
| [178]    | (Wu et al., 2019)                  |                                    |                        | FJS    | D               | MS        | ACO             | Bench     |
| [179]    | (Hu et al., 2020)                  |                                    |                        | FMS    | D               | MS        | Other           | Num       |
| [180]    | (Vital-Soto et al., 2020)          |                                    |                        | FJS    | S               | T        | LP, LS, SA      | Other     |
| [130]    | (Lunardi, Birgin, Laborie, et al., 2020) |                        |                        | FJS    | S               | MS        | LP              | Num       |
| [149]    | (Rossi & Lanzetta, 2020)           |                                    |                        | FJS    | S               | MS        | ACO             | Bench     |
| [181]    | (Zhu & Zhou, 2020)                 |                                    |                        | FJS    | S               | MS, Other | LP              | Other     |
| [128]    | (Chen et al., 2020)                |                                    |                        | Open   | S               | MS        | Other           | Bench     |
| [99]     | (Lunardi, Birgin, Roncori, et al., 2020) |                        |                        | FJS    | S               | MS        | LS, GA, TS      | Bench     |
| [182]    | (Xie et al., 2020)                 |                                    |                        | Mixed  | S               | MS        | Other           | Num       |
| [183]    | (Wang & Combolay, 2020)            |                                    |                        | FMS    | D               | MS        | Other           | Bench     |
| [184]    | (Meija & Lelebvre, 2020)           |                                    |                        | FMS    | S               | MS        | Other           | Num       |
| [185]    | (Pan et al., 2020)                 |                                    |                        | FMS    | S               | MS        | Other           | Num       |
| [123]    | (Madrazi & Judd, 2021)             |                                    |                        | Classic | S       | MS        | SA, Other       | Num       |
### Table 7. Overview of references in the dataset using undirected graphs.

| Ref. No. | References (Listed Chronologically) | Type of Undirected Graph | Layout | Scheduling Type | Objective | Solution Method | Case Study |
|----------|-------------------------------------|--------------------------|--------|----------------|-----------|-----------------|------------|
| [118]    | (Bendraouche et al., 2015)          | ✓                        | Other  | S              | MS        |                 |            |
| [121]    | (Bożejko et al., 2015)              | ✓                        | FFS    | S              | MS        | TS              | Num        |
| [162]    | (Baruwa & Piera, 2015)              | ✓                        | FMS    | S              | MS        | Other           | Bench      |
| [186]    | (Tellache & Boudhar, 2017b)         | ✓                        | Multi  | S              | MS        | LP              | BB Num     |
| [117]    | (Bendraouche & Boudhar, 2016)       | ✓                        | Other  | S              | MS        |                 |            |
| [44]     | (Behnamian, 2016)                   | ✓                        | Parallel| S              | MS        | PSO             | Bench      |
| [187]    | (Tellache & Boudhar, 2016)          | ✓                        | Multi  | S              | MS        | LP              | Bench      |
| [62]     | (Baruwa & Piera, 2016)              | ✓                        | FMS    | S              | MS, Other | Other           | Bench      |
| [188]    | (Kouider et al., 2017)              | ✓                        | Classic| S              | MS        | LP              | TS Bench   |
| [131]    | (Tellache & Boudhar, 2018)          | ✓                        | Multi  | S              | MS        | NEH, Other      | Num        |
| [76]     | (Tellache & Boudhar, 2017a)         | ✓                        | Open   | S              | MS        | Other           | Num        |
| [189]    | (Ilani et al., 2017)                | ✓                        | ✓       | Open           | S         | MS              | Other      |
| [116]    | (Reddy et al., 2017)                | ✓                        | FJS    | S              | MS        | Other           | Bench      |
| [190]    | (Grinshpoun et al., 2017)           | ✓                        | Open   | S              | MS, T     | Other           | Bench      |
| [113]    | (Thevenin et al., 2018)             | ✓                        | FJS    | S              | MS, Other | LP              | NEH, LS    | Num        |
| [191]    | (Cheng et al., 2018)                | ✓                        | Other  | S              | MS        | Other           | RC         |
| [75]     | (Tellache et al., 2019)             | ✓                        | ✓       | Open           | S         | MS              | Other      |
| [119]    | (Sun et al., 2019)                  | ✓                        | FJS    | S              | MS, T     | Other           | Num        |
| [109]    | (Mohabedding & Boudhar, 2019)       | ✓                        | ✓       | Other          | S         | MS              | Other      |
| [192]    | (Ilani et al., 2019)                | ✓                        | Open   | S              | MS        | Num             |            |
| [193]    | (X. Shi et al., 2020)               | ✓                        | FJS    | S              | MS        | GA              | Bench      |
| [28]     | (Sotskov, 2020)                     | ✓                        | Multi  | Classic        | S         | MS, T           |            |
Table 8. Overview of references in the dataset using Un/Directed graphs.

| References Using UN/DIRECTED GRAPHS (1/2) | Problem & Solution Characteristics |
|------------------------------------------|------------------------------------|
| **Type of Un/Directed Graph** | **Layout** | **Scheduling Type** | **Objective** | **Exact** | **Approximate** | **Case Study** |
| Semantic | Ant Colony | Disjunctive | Tree | Flow Shop | Job Shop | Other | |
| **Case Study** | **Ref. No.** | **Type** | **S** | **MS** | **LS** | **GA** | **Num** | **LP** | **NEH** | **RC** | **PSO** | **Other** | **Bench** | | | | | |
| **[81]** | (Zeng et al., 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[194]** | (Rossi et al., 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[105]** | (Nasiri, 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[101]** | (Mencia et al., 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[156]** | (Grimes & Hebrard, 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[159]** | (Ciro et al., 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[196]** | (Cruz-Chávez, 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[92]** | (Kemmer S MS, 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[197]** | (Amirghasemi & Zamani, 2015) | ✓ | | | | | | | | | | | | | | | | | |
| **[200]** | (Nouri et al., 2016a) | ✓ | | | | | | | | | | | | | | | | | |
| **[51]** | (AitZai et al., 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[199]** | (Kuhpfahl & Bierwirth, 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[201]** | (Shahzad & Mebarki, 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[100]** | (Afsar et al., 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[39]** | (El-Desoky et al., 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[103]** | (Choo et al., 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[203]** | (Huang et al., 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[83]** | (Y. Yang et al., 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[33]** | (Sobeyko & Monch, 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[204]** | (Ge et al., 2016) | ✓ | | | | | | | | | | | | | | | | | |
| **[205]** | (Q. Yang et al., 2016) | ✓ | | | | | | | | | | | | | | | | | |
| Ref. No. | References (Listed Chronologically) | Problem & Solution Characteristics |
|---------|--------------------------------------|------------------------------------|
|         |                                      | Semantic | Ant Colony | Disjunctive | Tree | Flow Shop | Job Shop | Other | Scheduling Type | Objective | Exact | Approximate | Case Study |
| [206]   | (Elmi & Topaloglu, 2016)             | ✓        | FFS        | CMS         | S    | Other     |          |       |                |           |       | ACO         | Num        |
| [207]   | (Jin et al., 2016)                  | ✓        | FJS        | S           | MS   |           |          |       |                | LP        |       |             | Bench      |
| [147]   | (Hart & Sim, 2016)                  | ✓        | Classic    | S           | MS, T|           |          |       |                | Other     |       |             | Bench      |
| [188]   | (Kouider et al., 2017)              | ✓        | Classic    | S           | MS   |           |          |       |                | LP        | TS   | Bench       |           |
| [208]   | (Knopp et al., 2017)                | ✓        | FJS        | S           | MS   | Other     |          |       |                | LS, SA    |       | ACO         | Bench      |
| [52]    | (El Khoukhi et al., 2017)           | ✓        | FJS        | S           | MS   |           |          |       |                | LP        | ACO  |             | Bench      |
| [54]    | (Wu et al., 2017)                   | ✓        | FJS        | S           | MS   | Other     |          |       |                | ACO       | RC   |             |           |
| [209]   | (Sotskov & Gholami, 2017)           | ✓        | FJS        | S           | MS   |           |          |       |                | Other     |       |             | Bench      |
| [151]   | (Zhang & Wong, 2017)                | ✓        | FJS        | D           | MS   |           |          |       |                | ACO       |       |             | Bench      |
| [210]   | (Bürgy, 2017)                      | ✓        | Classic    | S           | MS   | T          |          |       |                | TS        |       |             | Bench      |
| [211]   | (Hao et al., 2017)                  | ✓        | Classic    | S           | MS   | T          |          |       |                | Other     |       |             | Bench      |
| [157]   | (Wang et al., 2017)                 | ✓        | FJS        | S           | MS   | Other     |          |       |                | ACO       |       |             | Bench      |
| [168]   | (Yu et al., 2017)                   | ✓        | FJS        | S           | MS   |           |          |       |                | LP        | LS, Other |             | Num        |
| [212]   | (Zuo et al., 2017)                  | ✓        | FJS        | S           | MS   |           |          |       |                | LS, GA    |       |             | Bench      |
| [213]   | (Xiong et al., 2017)                | ✓        | Classic    | D           | T    |           |          |       |                | Other     | Num  |             | Bench      |
| [143]   | (Bierwirth & Kuhpfahl, 2017)        | ✓        | Classic    | S           | T    |           |          |       |                | LS, Other |       |             | Bench      |
| [38]    | (Mei et al., 2017)                  | ✓        | Classic    | D           | MS, T|           |          |       |                | GA, Other | Num  |             |           |
| [152]   | (Zhang et al., 2017)                | ✓        | FJS        | D           | MS, Other |           |          |       |                | Other     | RC   |             |           |
| [214]   | (Tamssaouet et al., 2018)           | ✓        | Classic    | S           | MS   |           |          |       |                | TS, SA    |       | Num         |           |
| [31]    | (Shen et al., 2018)                 | ✓        | FJS        | S           | MS   |           |          |       |                | TS        |       |             | Bench      |
| [215]   | (Yu & Lee, 2018)                    | ✓        | FJS        | S           | T    |           |          |       |                | LP        |       |             | Bench      |
| [49]    | (Lange & Werner, 2018)              | ✓        | Classic    | S           | T    |           |          |       |                | LP        |       |             | Num        |
| [58]    | (Zhao et al., 2018)                 | ✓        | FJS        | S           | MS, T, Other |           |          |       |                | ACO       |       |             | Num        |
| [216]   | (Xiong & Fu, 2018)                  | ✓        | FJS        | S           | MS   |           |          |       |                | Other     |       |             | Bench      |
| [104]   | (Abdel-Kader, 2018)                 | ✓        | Classic    | S           | MS   |           |          |       |                | PSO       |       |             | Bench      |
| [217]   | (Nagata & Ono, 2018)                | ✓        | Classic    | D           | MS   |           |          |       |                | LS, TS    |       |             | Bench      |
| Ref. No. | References (Listed Chronologically) | Type of Un/Directed Graph | Layout | Scheduling Type | Objective | Solution Method | Case Study |
|---------|------------------------------------|---------------------------|--------|----------------|-----------|----------------|-----------|
| [160]   | (Riahi & Kazemi, 2018)             | ✓                         | Multi  | S              | MS        | ACO, SA        | Bench     |
| [218]   | (Zhang & Wong, 2018)              | ✓                         | Classic | S              | Other     | ACO            | Bench     |
| [219]   | (Meolic & Brezočnik, 2018)        | ✓                         | FJS     | S              | MS, Other | Other          | Bench     |
| [148]   | (Zhang & Wang, 2018)              | ✓                         | FJS     | D              | MS        | LP             | Bench     |
| [146]   | (Zhou et al., 2018)               | ✓                         | FJS     | D              | MS, T     | GA             | Bench     |
| [154]   | (Lei et al., 2018)                | ✓                         | FJS     | S              | MS        | GA             | Bench     |
| [32]    | (Lamorgese & Mannino, 2019)       | ✓                         | Classic | S              | Other     | LP             | RC        |
| [50]    | (Chauouch et al., 2019)           | ✓                         | Classic | S              | MS        | ACO            | Bench     |
| [220]   | (Liu et al., 2019)                | ✓                         | FJS     | S              | T, Other  | LP             | GA        | Num       |
| [221]   | (Burdett et al., 2019)            | ✓                         | FJS     | S              | MS, Other | SA             | Num       |
| [222]   | (Cruz-Chávez et al., 2019)        | ✓                         | Classic | S              | MS        | GA             | Bench     |
| [155]   | (Zhu et al., 2019)                | ✓                         | FJS     | S              | MS        | ACO            | Num       |
| [158]   | (Qin et al., 2019)                | ✓                         | Mixed   | S              | MS        | ACO            | Bench     |
| [223]   | (Zhou et al., 2019)               | ✓                         | FJS     | D              | T, Other  | GA             | RC        |
| [224]   | (Jun et al., 2019)                | ✓                         | FJS     | S              | T         | LP             | GA        | Num       |
| [144]   | (Kim & Lee, 2019)                 | ✓                         | Multi   | S              | MS        | BB             | Num       |
| [102]   | (F. Shi et al., 2020)             | ✓                         | FJS     | S              | MS        | SB GA          | Bench     |
| [225]   | (Abedi et al., 2020)              | ✓                         | Classic | S              | T, Other  | LS, GA         | Bench     |
| [28]    | (Sotskov, 2020)                   | ✓                         | Multi   | Classic        | S          | MS             | Num       |
| [226]   | (Fan et al., 2019)                | ✓                         | FJS     | S              | MS        | LS, SA         | Bench     |
| [227]   | (Vela et al., 2020)               | ✓                         | Classic | S              | T         | GA, TS         | Bench     |
| [185]   | (Pan et al., 2020)                | ✓                         | FMS     | S              | MS        | Other          | Num       |
| [149]   | (Rossi & Lanzetta, 2020)          | ✓                         | FJS     | S              | MS        | ACO            | Bench     |
| [150]   | (Liu et al., 2020)                | ✓                         | FJS     | S              | MS        | GA             | Bench, RC |
| [70]    | (Gmys et al., 2020)               | ✓                         | Perm    | S              | MS        | BB             | Bench     |
| [145]   | (Zhang et al., 2020)              | ✓                         | FJS     | D              | MS, Other | GA             | Num       |
| [228]   | (Ahmadian et al., 2021)           | ✓                         | Classic | S              | T         | LS             | Bench     |
To have a better understanding of where the dataset stands within each category, this section presents simple visualizations to compare the frequency of subcategories in each macro category.

Figure 23 compares the percentage of different graphs in the dataset. It is observed that undirected graphs have been utilized less than the other types, while some of these undirected graphs could have great potential applications in the field of shop scheduling. For instance, Markov Random Field (MRF) model and Bayesian Network (BN) are among a few probabilistic graph-based models which can potentially consider the uncertainty of the real-world shop environments. However, they barely appeared in the dataset, (i.e., one usage of MRF and no usage of BN was observed). On the other hand, the significant percentage of references using the disjunctive graphs stands out in this chart.

![Figure 23. Percentage of different types of graph-based models in the dataset.](image)

Figure 24 shows that the majority of references reviewed in this study falls under the job shop macro category. This reveals an eye-opening gap in this literature, i.e., while job shop is not the most complicated and novel layout among shop manufacturing layouts, job shop and its relevant problems are still attracting so much attention. Whereas the intelligent systems including mixed systems have been practical in today’s dynamic production environment, they have been relatively neglected (based on our dataset). This is also true about flexible manufacturing systems (FMS) and cellular manufacturing systems (CMS) which often incorporate newer technology such as IoT and robots. An interesting observation is that 57% of the top 10% most cited papers focus on flexible job shop (FJS) in comparison to the entire dataset where 28.9% of papers focus on FJS.

![Figure 24. Percentage of different types of layouts in the dataset.](image)
We also observed that the proportion of the static (S) scheduling problems addressed in the dataset dramatically dominates the dynamic (D) ones, i.e., 87.4% of references addressed static scheduling comparing to 12.6% for dynamic scheduling problems. This dramatic difference can be noticed in Tables 6–8. Furthermore, an interesting point was observed by doing an additional analysis over the papers making up the top 10% of citations per year: of the top 10% of papers, 29% address dynamic scheduling which is notable in comparison to the entire dataset where only 12.6% address dynamic scheduling.

Minimizing the makespan is by far the most frequently objective used in 67% of the references in our dataset. The second most popular objectives are performance measures related to tardiness including total tardiness, mean tardiness, and weighted tardiness. Finally, about 17% of collected references used other objectives such as workload, lateness, flow time, etc. Figure 25 simply shows these proportions.

![Figure 25. Percentage of different objectives (performance measures) in the dataset.](image1)

Considering the efficiency and practicality of approximate methods to solve the scheduling problems on one hand, and the time expensiveness of exact method on the other, a greater application of approximate solutions was expected. Our dataset has confirmed this expectation in Figure 26. To be more precise, overall, 201 different methods are found in 143 references in our dataset; out of which only 16% are exact methods. Moreover, about 94% of all papers in our dataset used at least one type of Approximate method.

![Figure 26. Percentage of different types of solution methods in the dataset.](image2)

Linear programming (LP) and genetic algorithm (GA) are the most applied exact and approximate methods with 12.2% and 14.8% frequency among the data.

Finally, Figure 27 shows the lack of usage of real case study among the reviewed papers in the dataset. Those 10% and 29% of references which used ‘Real Case’ and ‘Numerical’ (Num) case study, usually also applied their solution to a benchmark problem.

![Figure 27. Percentage of different types of case studies in the dataset.](image3)
7. Analysis and Discussion

Beyond proposing a systematically data collection and categorization methodology, this paper seeks to provide meaningful insights while extracting and analyzing potential correlation among multiple categories and subcategories.

Performing analysis on the dataset might have obstacles that could lead to misleading results. A challenge in analyzing our dataset is related to the size of some subcategories: Comparing to other survey studies in literature (reviewing other manufacturing/scheduling related topics), the size of our dataset is adequately large. However, since the mainstreaks of our research (shop scheduling and graph theory) require detailed breakdown, some of the described micro categories have a few entries which can be problematic in terms of statistical analysis and comparisons.

To resolve the insufficient data issue in some subcategories, the majority of our analysis has been done by grouping these subcategories under more general labels. For example, Permutation Flow Shop, Single Stage Flow Shop, Multi-Stage Flow Shop, and Flexible Flow Shop are all combined into a blanket Flow Shop.

Also, it should be noted that a large proportion of our dataset features more than one of the several categorizations in each area. For example, one paper may discuss both 'Job Shop' and 'Flow Shop' while utilizing both 'Disjunctive' and 'Bipartite Graph' to model their problems. Our dataset shows that on average, more than one layout type was discussed in each paper. This means that when we look at the presence of a specific feature as a percent and compare it to other features in the same macro category, the sum of percentages might be greater than 100%. Despite this detracting factor, percentage presence is still a very useful tool in our analysis as it simplifies making comparisons especially between differently sized categories.

Our analysis starts with interpreting the impact of the system layout on the choice of type of graphs to model the system (the overlap between macro categories of graph types and layouts) shown in Figure 28. Overall, undirected graphs are the least popular across all different type of systems. An observation in this figure is that un/directed graphs are used more in job shop systems than others, particularly flow shop systems. A possible reason for this observation is that job shop systems are generally more flexible than most flow shop systems; also, most of un/directed graphs can potentially feature more flexibility to model the system. Therefore, un/directed graphs might be a better modeling tool for job shop systems. This justification can become more valid when we look at the highest percentage of the application of directed graph in flow shop problems comparing to other systems: using obligatory oriented edges in a directed graph can make the directed graph a more precise tool to model flow shop systems with a predefined routing.

![Figure 28. Overlap between Graph types and Shop types.](image)

After understanding the system and formally defining the problem using potential graph-based models, an appropriate method to solve the scheduling problem should be selected. Figure 29 summarizes the relationship between the usages of solution methods and types of graphs. Despite the dominant usage of approximate methods within our dataset, almost the same proportions of different types of graphs are observed in the subsets
of papers applying exact and approximate methods. This means that, excluding those solution methods requiring a particular graph structure (e.g., Ant colony graph which require disjunctive graph-Un/Directed), a variety of graph types can be used in problems which are solved by either exact or approximate methods.

![Graph overlap chart](image)

**Figure 29.** Overlap between Graph types and Method types.

Finally, the selected model and solution method should be verified with a case study. The choice of case study (numerical example, benchmark problem, or a real case) is seemingly independent of the type of graph used, so an in-depth analysis is not necessary. The usage of benchmark problems dominates our dataset while real cases were rarely utilized. This can most likely be explained due to the ease of access, minimal required resources, and the widely accepted verification aspect that come with using a benchmark problem when showcasing the validity of a new solution. Since this field of research does incorporate modeling of real manufacturing systems, it could be helpful for researchers to focus further on real cases to promote the implementation of these solutions.

We closed our analysis by proposing an absolutely novel modeling approach to visualize both gaps and hotspots in the literature based on our findings in the dataset (Figure 30). It should be noted that we generated this graph-based model from scratch, while incorporating a few basic concepts from AND/OR graph such as “Join”, and “OR” nodes. Also, based on the structure of this modeling approach it can be classified as DAGs. Figure 30 represents all steps required to address a shop scheduling problem in research. We took advantage of our proposed categorizations to defines nodes in each step of the problems. Nodes (micro-categories) are numbered to be called later. Edges show the precedence of these steps in scheduling problems. The weights of nodes denote the percent of frequency of associated step in the dataset. Each path from the ‘Start’ node to the ‘End’ node can potentially be the definition of a scheduling problem. Therefore, many permutations of potential scheduling problems can be defined. Paths with minimum weights can possibly show the least frequently addressed problem in the literature based on this dataset.

The last step in Figure 30 refers to the integration of scheduling problems and process planning procedure. Integrating these two procedures is essential in today’s industry since it can boost the sustainable development in high-tech facilitated manufacturing systems. Interestingly, this progressive topic has been observed multiple times in the subset of our references which are selected here as representatives of gaps and hotspots in the literature. Given the importance of the integration problem for future research directions, this step is added to Figure 30.

Then, Table 9 contains a sample of the least common graph types use in a scheduling problem which were modeled by a path. Each path (problem) can be decoded using Figure 30. For instance, the first problem in this table uses game tree and addresses a ‘multi-objective’ (=2), ‘dynamic’ (=2), ‘flexible job shop’ (=3) scheduling problem is solved by an ‘other’ (=11) type of approximate method and showcases the results by a real case study ‘not considering integration’ (=2).
Figure 30. Modeling and coding different properties of shop scheduling problems using micro-categories proposed in this paper.

Table 9. Samples of gaps in the literature: The properties of problems using the least common graph types in our dataset.

| Ref. No. | Reference            | Graph Type                        | System | Objective | Scheduling | Solution | Case Study | Integration |
|----------|----------------------|-----------------------------------|--------|-----------|------------|----------|------------|-------------|
| [152]    | (Zhang et al., 2017) | Game tree                         | 3      | 2         | 2          | 11       | 3          | 2           |
| [144]    | (Kim & Lee, 2019)   | B&B tree                          | 5      | 1         | 1          | 1        | 2          | 2           |
| [70]     | (Gmys et al., 2020) | B&B tree                          | 4      | 1         | 1          | 1        | 1          | 2           |
| [154]    | (Lei et al., 2018)  | Processing operation tree         | 1      | 1         | 1          | 6        | 1          | 1           |
| [149]    | (Rossi & Lanzetta, 2020) | Multiple graphs: And-Or & precedence graphs | 3      | 1         | 1          | 7        | 1          | 2           |
| [218]    | (Zhang & Wong, 2018) | AND-OR graph & disjunctive graph  | 2      | 2         | 1          | 7        | 1          | 1           |
| [155]    | (Zhu et al., 2019)  | Semantic graph                    | 3      | 1         | 2          | 7        | 2          | 2           |
| [185]    | (Pan et al., 2020)  | Multiple graphs: Semantic graph & Petri Net | 10     | 1         | 2          | 11       | 2          | 2           |
| [193]    | (X. Shi et al., 2020) | Social network                    | 3      | 1         | 1          | 6        | 1          | 1           |
| [119]    | (Sun et al., 2019)  | Markov Random Fields (MRFs)       | 3      | 1         | 1          | 11       | 2          | 2           |
| [121]    | (Bozejko et al., 2015) | Traveling salesman graph          | 6      | 1         | 1          | 8        | 2          | 2           |

The combination of Table 9 and Figure 30 can show some gaps in the field of graph applications in scheduling problems. This combination can also identify new potential problems (by tracking paths with minimum total weight).

Furthermore, Table 10 presents the top 10% most cited scheduling problems and the associated graph types used in these references. The path representing each problem is denoted in each row of the table. This table can potentially highlight the hotspots and trends in this field. Couple overlaps between Tables 9 and 10 may imply a bright future
for the research area since less common graph-based models have been getting more exposure (citations).

Table 10. Properties of the problems in the top 10% most cited papers.

| Reference | Graph Type | System | Objective | Scheduling | Solution | Case Study | Integration |
|-----------|------------|--------|-----------|------------|----------|------------|-------------|
| [31]      | (Shen et al., 2018) | Disjunctive graph | 3 | 1 | 1 | 8 | 1 | 2 |
| [179]     | (Hu et al., 2020) | Petri-net | 10 | 1 | 2 | 11 | 2 | 2 |
| [218]     | (Zhang & Wong, 2018) | Multiple graphs: AND-OR graph & disjunctive graph | 2 | 2 | 1 | 7 | 1 | 1 |
| [69]      | (Wang & Wang, 2015) | DAG | 4 | 1 | 1 | 5,6 | 1 | 2 |
| [213]     | (Xiong et al., 2017) | Disjunctive graph | 2 | 1 | 2 | 11 | 1,2 | 2 |
| [199]     | (Kuhpfahl & Bierwirth, 2016) | Disjunctive graph | 2 | 1 | 1 | 5 | 1 | 2 |
| [152]     | (Zhang et al., 2017) | Game tree | 3 | 2 | 2 | 11 | 3 | 2 |
| [180]     | (Vital-Soto et al., 2020) | Precedence graph | 3 | 1 | 1 | 2,5,9,11 | 1,3 | 2 |
| [70]      | (Gmys et al., 2020) | B&B tree | 4 | 1 | 1 | 1 | 1 | 2 |
| [145]     | (Zhang et al., 2020) | Tree | 3 | 2 | 2 | 6 | 2 | 2 |
| [200]     | (Nouri et al., 2016b) | Disjunctive graph | 3 | 1 | 1 | 6,8 | 1 | 2 |
| [216]     | (Xiong & Fu, 2018) | Disjunctive graph | 3 | 1 | 1 | 11 | 1 | 2 |
| [52]      | (El Khoukhi et al., 2017) | Ant colony graph | 3 | 1 | 1 | 2,7 | 1 | 2 |
| [124]     | (González et al., 2015) | DAG | 3 | 1 | 1 | 5,8 | 1 | 2 |

8. Summary of Our Results

In this section, a summary of our results will be presented and the interpretation of these results and future research directions are discussed in the next section, Conclusion.

Our dataset contains 143 academic peer-reviewed journal papers published from 2015 to 2020 targeting the intersection of shop scheduling and graph theory research. Given the size and scope of the literature, focusing on relevant publications within the recent years could offer interesting insights to coincide with the current trends of the field.

Next, our paper offered a brief analysis over the metadata (e.g., journals, publishers, years, etc.) of the collected references (Section 2). We proposed a novel subjective measurement to evaluate the quality of the journals in our dataset. The three subjective categories over the quality of the target journals are Premier, Major, and Others. This procedure and categorization have not been observed in any previous review papers. About 70% of our dataset had been published by either ‘Premier’ or ‘Major’ journals which confirm the validity of the references used in this survey (Figure 4).

Then, comprehensive categorizations over system layouts, scheduling problem, scheduling solutions methods, different type of graphs were proposed in Sections 3–5. The classification of graph-based models applied in shop scheduling problems in Figure 8 is novel and created from scratch in this study. The macro categories include Directed, Undirected, and Un/Directed graphs categories. The latest category refers to those graphs with both directed and undirected edges at the same time, or those graphs that can have either all directed or undirected edges. The basic concepts of all these graph types and their applications to the shop scheduling problems were described in Section 5.

The statistical analyses over individual categories and across multiple categories were performed in Sections 6 and 7. The overview of the full dataset (in Section 6) shows that more than half of the graphs (54%) used in our dataset are in Un/Directed category; while
only 14% are undirected graphs which reveals a gap of literature to applied graphs such as Social Network, MRFs, etc.

In our dataset, 56% of the total problems addressed a type of job shop systems. Also, the most cited paper within our dataset is to be a job shop scheduling problem (precisely, FJS) that utilizes an un/directed graph (precisely, disjunctive graph). This also confirms the significant attention paid towards job shop systems and Un/Directed graphs. Less popular systems within our dataset include CMS, FMS, and the cyclic shop. These shops may be less desirable to model as they are more complex and are relatively newer shop layouts in comparison to job shop or flow shop, which should be explored profoundly in future studies.

About 85% of the collected references used a single objective, and the most popular objective function used in the dataset is makespan. Considering other objective functions beside makespan and tackling multi-objectives scheduling problems can be an interesting future research if such problems can utilize less common but effective graph types such as game tree to model the problem. Note that game tree was only observed once in our dataset.

Moreover, in terms of solving the problems, approximate methods dominate the solution techniques in our dataset, comparing to the exact methods, i.e., 94% of all collected papers applied at least one type of approximate method. This was a quite expected observation given the NP-harness of most of shop scheduling problems.

Another immediate observation in this regard is the usages of more than one solution method per paper in several cases (a total of 201 methods in 143 papers). Sometimes the major contribution of these papers is in the details of their proposed approximate methods, while addressing a classic problem and using slightly better approximate methods.

Finally, only 10% of the dataset applied a real case study to demonstrate their results and, the rest of papers used either numerical studies (29%) or benchmark problems (61%). The dominant usage of benchmark problems in this research area is beneficial since these problems enables the comparison among the previous solutions/results and the new/improved solutions. However, to show the practicality of the solutions, it is important to implement the proposed solution/results on the real case, in addition to a benchmark or numerical study.

9. Conclusions and Future Research Directions

Graph theory and graph-based models have been powerful tools to understand the functionality of different systems. Graphs particularly can simplify the analysis of complex systems. Graphs have major applications in a variety of fields including many sciences and engineering research areas. Manufacturing systems and their associated problems including shop scheduling problems extensively benefits from graph theory concepts since all elements of the systems including resources, materials, parts, processes, and their connections can be represented and analyzed by the connected network of nodes and edges called as graphs.

Despite the tremendous number of publications applying graph concepts to shop scheduling problems, the literature lacks a comprehensive survey study in this area. To the best of our knowledge, our paper is the first review study in the literature to propose such a comprehensive review paper which systematically studies the overview and the perspective of this research field.

First, we proposed our methodology (Figure 1) to collect and review relevant references utilizing graph theory to model and/or solve different variations of shop scheduling problems in the domain of manufacturing systems. This methodology can easily be generalized and extended to be used in other survey studies.

Purely by the size of our collected data, we can conclude that this field of research is popular and of importance, even though neither shop scheduling nor graph theory are new concepts. This can be explained by the ever-evolving nature of manufacturing. As new technology is introduced, such as robots or IoT, the system can be altered and ideally
optimized further, allowing researchers to revisit this space. Even in a shop floor without “newer technology”, these systems are incredibly complex with many constraints and attributes that are not addressed in the classic versions of their scheduling problems, so generating sustainable modeling approaches and solutions addressing the complexity of systems with all constraints can help apply said solutions to a real manufacturing system.

After finalizing out collected data, we used a novel procedure/measurement to analyze the distribution of the quality of journals publishing our collected references. This analysis is one of our critical contributions since it can confirm the validation of our dataset, and more importantly it can indicate the research trend and hotspot in the literature.

Thereafter, considering the variety of problems addressed in our collected data, we proposed our own modified version of categorizations over system layouts, scheduling problem, and scheduling solutions methods. These categorizations are comprehensive and detailed enough to cover all aspects of problems’ properties without over-subdividing or exhausting micro categories.

Almost all papers in our dataset used at least one type of approximate method as scheduling solutions and some papers applied multiple methods, e.g., a slightly improved heuristics/meta-heuristics, to address classical scheduling problems. Considering the rich background of scheduling approximate solutions, insisting on improving the quality of solutions by proposing/improving an efficient method for well-studied shop scheduling problems seems unnecessary. Instead, future studies can focus on new problems and precisely embrace the complexity of the post-modern manufacturing environment and industry 4.0 by proposing sustainable and innovative modeling approaches incorporating different powerful graph-based techniques and concepts.

Moreover, it was observed in our dataset that the integration of shop scheduling and process planning procedures has been relatively neglected. These two procedures should not be considered independent. It was shown that integrating the process planning and shop scheduling results into the strategic and sustainable development required in intelligent manufacturing systems. Thus, studying the integration problems is another essential future research area.

Next, all types of graph-based models detected in the collected papers were analyzed using our novel proposed categorizes. Based on our record, the literature lacks an application of some particular graphs such as Social Network, MRFs, etc. Moreover, our data shows that using more than one type of graph for modeling the system has not been popular. Taking advantage of more than one graph type (generating hybrid graphs) could alleviate some of the additional encoding necessary to consider certain constraints or multiple objectives.

Finally, the lack of usage of real case study among the reviewed papers (only 10%) shows that more serious efforts are required to bring industry and academia together by providing incorporative open-access database to share real data and communicate recent development in theory and practice.

Probably the most important breakthrough in this study is to put theory into practice by proposing a unique and an absolutely novel modeling approach to visualize both gaps and hotspots in the literature based on our findings in the dataset (Figure 30). Please notice that we generated this graph-based model from scratch to show the functionality of different types of graphs to model any systems including a review survey. That is the whole point of using graphs.

Given the extent of the recent literature and the surge of researchers and top-ranked journals paying attention to the topic of the application of graph-based model in shop scheduling problems, we encourage researchers to extend the period of data collection and conduct a large-scale survey which could review the full literature of the past decades in this field.

**Author Contributions:** Conceptualization, J.O., A.M., G.M., and S.M.; methodology, J.O., A.M., G.M., and S.M.; validation, J.O., A.M., G.M., and S.M.; formal analysis, J.O., A.M., and G.M.; investigation, J.O., A.M., and G.M.; data curation, J.O., A.M., and G.M.; writing—original draft preparation, J.O.,
A.M., and G.M.; writing—review and editing, J.O., A.M., and G.M.; visualization, J.O., A.M., G.M., and S.M.; supervision, G.M. and S.M.; project administration, G.M. and S.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** Not applicable.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** The authors would like to kindly acknowledge assistance from Haley Clark for contributions to the paper including help with data collection and graph visualization.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Trudeau, R.J. *Introduction to Graph Theory*, Courier Corporation: North Chelmsford, MA, USA, 2013.
2. Carlson, S.C. Königsberg Bridge Problem, in Encyclopedia Britannica. 2010. Available online: https://www.britannica.com/science/Konigsberg-bridge-problem (accessed on 16 May 2021).
3. Mehrani, R; Sharma, S. Behavior of water confined between hydrophobic surfaces with grafted segments. *Colloid Interface Sci. Commun.* 2021, 40, 100355. [CrossRef]
4. Madraki, Y; Oakley, A; Le, A.N.; Colín, A.; Ovarlez, G.; Hormozi, S. Shear thickening in dense non-Brownian suspensions: Viscous to inertial transition. *J. Rheol.* 2020, 64, 227–238. [CrossRef]
5. Riaz, F; Ali, K.M. Applications of Graph Theory in Computer Science. In Proceedings of the 2011 Third International Conference on Computational Intelligence, Communication Systems and Networks, Bali, Indonesia, 26–28 July 2011; pp. 142–145.
6. Madraki, G.; Grasso, I; Otala, J.; Liu, Y.; Matthews, J. Characterizing and Comparing COVID-19 Misinformation Across Languages, Countries and Platforms. *arXiv 2020*, arXiv:2010.06455.
7. Amato, F; Castiglione, A; De Santo, A; Moscato, V; Picariello, A; Persia, F; Sperli, G. Recognizing human behaviours in online social networks. *Comput. Secur.* 2018, 74, 355–370. [CrossRef]
8. Amato, F; Moscato, V; Picariello, A; Sperli, G. Multimedia Social Network Modeling: A Proposal. In Proceedings of the 2016 IEEE Tenth International Conference on Semantic Computing (ICSC), Laguna Hills, CA, USA, 4–6 February 2016; pp. 448–453.
9. Otala, J.M.; Kurtic, G.; Grasso, I.; Liu, Y.; Matthews, J.; Madraki, G. 2021. Political Polarization and Platform Migration: A Study of Parler and Twitter Usage by United States of America Congress Members. In Companion Proceedings of the Web Conference 2021 (WWW ’21 Companion), April 19–23, 2021, Ljubljana, Slovenia; ACM: New York, NY, USA, 2021; p. 10. [CrossRef]
10. La Gatta, V; Moscato, V; Postiglione, M; Sperli, G. CASTLE: Cluster-aided space transformation for local explanations. *Expert Syst. Appl.* 2021, 179, 115045. [CrossRef]
11. Djakbarova, U; Madraki, Y; Chan, E.T; Kural, C. Dynamic interplay between cell membrane tension and clathrin-mediated endocytosis. *Biol. Cell 2021*. [CrossRef]
12. Mehrdad, S; Mousavian, S; Madraki, G; Dvorkin, Y. Cyber-Physical Resilience of Electrical Power Systems Against Malicious Attacks: A Review. *Curr. Sustain. Energy Rep.* 2018, 5, 14–22. [CrossRef]
13. Mousavian, S; Conejo, A.J; Sioshansi, R. Equilibria in investment and spot electricity markets: A conjectural-variations approach. *Eur. J. Oper. Res.* 2020, 281, 129–140. [CrossRef]
14. Mousavian, S; Erol-Kantarci, M; Mouftah, H.T. Cyber-Security and Resiliency of Transportation and Power Systems in Smart Cities. In *Transportation and Power Grid in Smart Cities*; Wiley: Hoboken, NJ, USA, 2018; pp. 507–527.
15. Hosseini, S; Ivanov, D. Bayesian networks for supply chain risk, resilience and ripple effect analysis: A literature review. *Expert Syst. Appl.* 2020, 161, 113649. [CrossRef] [PubMed]
16. Salmani, Y; Partovi, F.Y; Banerjee, A. Customer-driven investment decisions in existing multiple sales channels: A downstream supply chain analysis. *Int. J. Prod. Econ.* 2018, 204, 44–58. [CrossRef]
17. La Gatta, V; Moscato, V; Postiglione, M; Sperli, G. An Epidemiological Neural Network Exploiting Dynamic Graph Structured Data Applied to the COVID-19 Outbreak. *IEEE Trans. Big Data* 2021, 7, 45–55. [CrossRef]
18. Caggiano, A. Manufacturing System. In *CIRP Encyclopedia of Production Engineering*; Laperrière, L; Reinhart, G, Eds.; Springer: Berlin/Heidelberg, Germany, 2014; pp. 830–836.
19. Fan, K; Zhai, Y; Li, X; Wang, M. Review and classification of hybrid shop scheduling. *Prod. Eng.* 2018, 12, 597–609. [CrossRef]
20. Singh, G; Kaur, G; Singh, G. Review of Graph Based Scheduling Algorithms. *Int. Res. J. Eng. Technol. (IRJET)* 2015, 2.
21. Sachdeva, S; Panwar, P. A review of multiprocessor directed acyclic graph (DAG) scheduling algorithms. *Int. J. Comput. Sci. Commun.* 2015, 66, 67–72.
22. Weise, J; Benkhardt, S; Mostaghim, S. A Survey on Graph-based Systems in Manufacturing Processes. In Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence (SSCI), Bangalore, India, 18–21 November 2018; Institute of Electrical and Electronics Engineers (IEEE): Piscataway, NJ, USA, 2018; pp. 112–119.
85. Pinedo, M.; Hadavi, K. Scheduling: Theory, Algorithms and Systems Development. In Proceedings of the Operations Research Proceedings 1991; Springer Science and Business Media LLC: Berlin/Heidelberg, Germany, 1992; pp. 35–42.

86. Chen, B.; Potts, C.N.; Woeginger, G.J. A Review of Machine Scheduling: Complexity, Algorithms and Approximability. Handb. Comb. Optim. 1998, 1493–1641. [CrossRef]

87. Steep, P.P.; Wiers, V.C. The complexity of scheduling in practice. Int. J. Oper. Prod. Manag. 1996, 16, 37–53. [CrossRef]

88. Lin, L.; Gen, M.; Liang, Y.; Ohno, K. A Hybrid EA for Reactive Flexible Job-shop Scheduling. Procedia Comput. Sci. 2012, 12, 110–115. [CrossRef]

89. Amjad, M.K.; Butt, S.I.; Kousar, R.; Ahmad, R.; Agha, M.H.; Faping, Z.; Anjum, N.; Asgher, U. Recent Research Trends in Genetic Algorithm Based Flexible Job Shop Scheduling Problems. Math. Probl. Eng. 2018, 2018, 1–32. [CrossRef]

90. Mokotoff, E. Parallel machine scheduling problems: A survey. Asia Pacif. J. Oper. Res. 2001, 18, 193.

91. Marti, R.; Reinelt, G. The Linear Ordering Problem: Exact and Heuristic Methods in Combinatorial Optimization; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011; Volume 175.

92. Kemmoe, S.; Lamy, D.; Tchernev, N. A Job-shop with an Energy Threshold Issue Considering Operations with Consumption Peaks. IFAC PapersOnLine 2015, 48, 788–793. [CrossRef]

93. Sawik, T. Mixed integer programming for scheduling flexible flow lines with limited intermediate buffers. Math. Comput. Model. 2000, 31, 39–52. [CrossRef]

94. Jourdan, L.; Basseur, M.; Talbi, E.-G. Hybridizing exact methods and metaheuristics: A taxonomy. Eur. J. Oper. Res. 2009, 199, 620–629. [CrossRef]

95. büyüktahtakin, I.E. Dynamic Programming Via Linear Programming. Wiley Encyclop. Oper. Res. Manag. Sci. 2010. [CrossRef]

96. Williamson, D.P.; Shmoys, D.B. The Design of Approximation Algorithms. Design Approxim. Algorith. 2009.

97. Bianchi, L.; Dorigo, M.; Gambardella, L.M.; Gutjahr, W.J. A survey on metaheuristics for stochastic combinatorial optimization. Nat. Comput. 2009, 8, 239–287. [CrossRef]

98. Framinan, J.M.; Leisten, R.; Ruiz-Usano, R. Efficient heuristics for flowshop sequencing with the objectives of makespan and flowtime minimisation. Eur. J. Oper. Res. 2002, 141, 559–569. [CrossRef]

99. Lunardi, W.T.; Birgin, E.G.; Ronconi, D.P.; Voos, H. Metaheuristics for the online printing shop scheduling problem. Eur. J. Oper. Res. 2020. [CrossRef]

100. Afsar, H.; Lacomme, P.; Ren, L.; Prodhon, C.; Vigo, D. Resolution of a Job-Shop problem with transportation constraints: A master/slave approach. IFAC PapersOnLine 2016, 49, 898–903. [CrossRef]

101. Mencía, R.; Sierra, M.R.; Mencía, C.; Varela, R. Memetic algorithms for the job shop scheduling problem with operators. Appl. Soft Comput. 2015, 34, 94–105. [CrossRef]

102. Shi, F.; Zhao, S.; Meng, Y. Hybrid algorithm based on improved extended shifting bottleneck procedure and GA for assembly job shop scheduling problem. Int. J. Prod. Res. 2020, 58, 2604–2625. [CrossRef]

103. Cho, W.M.; Wong, L.-P.; Khader, A.T. A modified bee colony optimization with local search approach for job shop scheduling problems relevant to bottleneck machines. Int. J. Adv. Soft Comput. Appl. 2016, 8, 52–78.

104. Abdel-Kader, R.F. An improved PSO algorithm with genetic and neighborhood-based diversity operators for the job shop scheduling problem. Appl. Artif. Intell. 2018, 32, 433–462. [CrossRef]

105. Nasić, M.M. A modified ABC algorithm for the stage shop scheduling problem. Appl. Soft Comput. 2015, 28, 81–89. [CrossRef]

106. Bahalkeh, E.; Madraki, G.; Judd, R. Efficient system matrix calculation for manufacturing systems. In Proceedings of the IIE Annual Conference, Nashville, TN, USA, 30 May–2 June 2015; Institute of Industrial and Systems Engineers (IISE): Norcross, GA, USA, 2015; pp. 35–42.

107. Moon, J.W.; Moser, I. On cliques in graphs. Isr. J. Math. 1965, 3, 23–28. [CrossRef]

108. Prosser, P. Exact Algorithms for Maximum Clique: A Computational Study. Algorithms 2012, 5, 545–587. [CrossRef]

109. Mohabeddine, A.; Boudhar, M. New results in two identical machines scheduling with agreement graphs. Theor. Comput. Sci. 2019, 779, 37–46. [CrossRef]

110. Even, G.; Halldörsson, M.M.; Kaplan, L.; Ron, D. Scheduling with conflicts: Online and offline algorithms. J. Sched. 2008, 12, 199–224. [CrossRef]

111. Halldórsson, M.M.; Kortsarz, G. Multicoloring: Problems and Techniques. In Proceedings of the International Symposium on Mathematical Foundations of Computer Science, Prague, Czech Republic, 22–27 August 2004; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3153, pp. 25–41.

112. Furmańczyk, H.; Kubale, M. Scheduling of unit-length jobs with cubic incompatibility graphs on three uniform machines. Discret. Appl. Math. 2018, 234, 210–217. [CrossRef]

113. Thevenin, S.; Zufferey, N.; Potvin, J.-Y. Graph multi-coloring for a job scheduling application. Discret. Appl. Math. 2018, 234, 218–235. [CrossRef]

114. Furmańczyk, H.; Kubale, M. The complexity of equitable vertex coloring of graphs. J. Appl. Comput. Sci. 2005, 13, 95–106.

115. Ganguli, R.; Roy, S. A study on course timetable scheduling using graph coloring approach. Int. J. Comput. Appl. Math. 2017, 12, 469–485.

116. Reddy, M.S.; Ratnam, C.; Agrawal, R.; Varela, M.; Sharma, I.; Manupati, V. Investigation of reconfiguration effect on makespan with social network method for flexible job shop scheduling problem. Comput. Ind. Eng. 2017, 110, 231–241. [CrossRef]

117. Bendaouche, M.; Boudhar, M. Scheduling with agreements: New results. Int. J. Prod. Res. 2015, 54, 1–15. [CrossRef]
118. Bendraouche, M.; Boudhar, M.; Oulamara, A. Scheduling: Agreement graph vs resource constraints. *Eur. J. Oper. Res.* 2015, 240, 355–360. [CrossRef]

119. Sun, L.; Lin, L.; Li, H.; Gen, M. Cooperative Co-Evolution Algorithm with an MRF-Based Decomposition Strategy for Stochastic Flexible Job Shop Scheduling. *Mathematics* 2019, 7, 318. [CrossRef]

120. Rosenkrantz, D.J.; Stearns, R.E.; Lewis, P.M. Approximate algorithms for the traveling salesperson problem. *15th Ann. Sympos. Switch. Automata Theory*. 1974. [CrossRef]

121. Bożejko, W.; Uchorowski, M.; Wodecki, M. Block approach to the cyclic flow shop scheduling. *Comput. Ind. Eng.* 2015, 81, 158–166. [CrossRef]

122. Bożejko, W.; Gnataowski, A.; Pempera, J.; Wodecki, M. Parallel tabu search for the cyclic job shop scheduling problem. *Comput. Ind. Eng.* 2017, 113, 512–524. [CrossRef]

123. Madraki, G.; Judd, R. Accelerating the calculation of makespan used in scheduling improvement heuristics. *Comput. Oper. Res.* 2021, 130, 105233. [CrossRef]

124. González, M.A.; Vela, M.D.C.R.; Varela, R. Scatter search with path relinking for the flexible job shop scheduling problem. *Eur. J. Oper. Res.* 2015, 245, 35–45. [CrossRef]

125. Kahn, A.B. Topological sorting of large networks. *Commun. ACM* 1962, 5, 558–562. [CrossRef]

126. Selvakumar, S.; Murthy, C.S.R. Scheduling precedence constrained task graphs with non-negligible intertask communication onto multiprocessors. *IEEE Trans. Parallel Distrib. Syst.* 1994, 5, 328–336. [CrossRef]

127. Katriel, I.; Michel, L.; Van Hentenryck, P. Maintaining Longest Paths Incrementally. *Constraints* 2005, 10, 159–183. [CrossRef]

128. Chen, Y.; Goebel, R.; Lin, G.; Su, B.; Zhang, A. Open-shop scheduling for unit jobs under precedence constraints. *Theor. Comput. Sci.* 2020, 803, 144–151. [CrossRef]

129. Birgin, E.; Ferreira, J.; Ronconi, D. List scheduling and beam search methods for the flexible job shop scheduling problem with sequencing flexibility. *Eur. J. Oper. Res.* 2015, 247, 421–440. [CrossRef]

130. Lunardi, W.T.; Birgin, E.G.; Laborie, P.; Ronconi, D.P.; Voos, H. Mixed Integer linear programming and constraint programming models for the online printing shop scheduling problem. *Comput. Oper. Res.* 2020, 123, 105020. [CrossRef]

131. Tellache, N.E.H.; Boudhar, M. Flow shop scheduling problem with conflict graphs. *Ann. Oper. Res.* 2017, 261, 339–363. [CrossRef]

132. Leake, C.; Baccelli, F.; Cohen, G.; Olsder, G.J.; Quadrat, J.-P. Synchronization and Linearity: An Algebra for Discrete Event Systems. *J. Oper. Res. Soc.* 1994, 45, 118. [CrossRef]

133. Lee, T.-E.; Park, S.-H. An Extended Event Graph With Negative Places and Tokens for Time Window Constraints. *IEEE Trans. Autom. Sci. Eng.* 2005, 2, 319–332. [CrossRef]

134. Cohen, G.; Gaubert, S.; Quadrat, J.-P. Max-plus algebra and system theory: Where we are and where to go now. *Ann. Rev. Control.* 1999, 23, 207–219. [CrossRef]

135. Zhou, M.; Venkatesh, K. *Modeling, Simulation, and Control of Flexible Manufacturing Systems*; World Scientific Pub Co Pte Lt: Singapore, 1999.

136. Zhou, M.; Dicesare, F; DesRochers, A. A hybrid methodology for synthesis of Petri net models for manufacturing systems. *IEEE Trans. Automat. Control.* 1992, 8, 350–361. [CrossRef]

137. Lusby, R.M.; Larsen, J.; Ehrgot, M.; Ryan, D. Railway track allocation: Models and methods. *OR Spectr.* 2011, 33, 843–883. [CrossRef]

138. Dabah, A.; Bendjoudi, A.; AitZai, A.; El-Baz, D.; Taboujdatem, N.N. Hybrid multi-core CPU and GPU-based B&B approaches for the blocking job shop scheduling problem. *J. Parallel Distrib. Comput.* 2018, 117, 73–86. [CrossRef]

139. Gholami, O.; Krasemann, J.T. A Heuristic Approach to Solving the Train Traffic Re-Scheduling Problem in Real Time. *Algorithms* 2018, 11, 55. [CrossRef]

140. Corominas, A.; Garcia-Villoria, A.; Gonzalez, N.-A.; Pastor, R. A multistage graph-based procedure for solving a just-in-time flexible job-shop scheduling problem with machine and time-dependent processing costs. *J. Oper. Res. Soc.* 2018, 70, 620–633. [CrossRef]

141. Wang, S.; Liu, M.; Chu, C. A branch-and-bound algorithm for two-stage no-wait hybrid flow-shop scheduling. *Int. J. Prod. Res.* 2014, 53, 1143–1167. [CrossRef]

142. Nip, K.; Wang, Z.; Nobibon, F.T.; Leus, R. A combination of flow shop scheduling and the shortest path problem. *J. Combin. Optim.* 2015, 29, 36–52. [CrossRef]

143. Bierwirth, C.; Kuhpfahl, J. Extended GRASP for the job shop scheduling problem with total weighted tardiness objective. *Eur. J. Oper. Res.* 2017, 261, 835–848. [CrossRef]

144. Kim, H.-J.; Lee, J.-H. Three-machine flow shop scheduling with overlapping waiting time constraints. *Comput. Oper. Res.* 2019, 101, 93–102. [CrossRef]

145. Zhang, F.; Mei, Y.; Nguyen, S.; Zhang, M. Evolving Scheduling Heuristics via Genetic Programming With Feature Selection in Dynamic Flexible Job-Shop Scheduling. *IEEE Trans. Cybern.* 2021, 51, 1797–1811. [CrossRef] [PubMed]

146. Zhou, Y.; Yang, J.-J.; Zheng, L.-Y. Hyper-Heuristic Coevolution of Machine Assignment and Job Sequencing Rules for Multi-Objective Dynamic Flexible Job Shop Scheduling. *IEEE Access* 2018, 7, 68–88. [CrossRef]

147. Hart, E.; Sim, K. A hyper-heuristic ensemble method for static job-shop scheduling. *Evolut. Comput.* 2016, 24, 609–635. [CrossRef]
177. Tian, S.; Wang, T.; Zhang, L.; Wu, X. Real-time shop floor scheduling method based on virtual queue adaptive control: Algorithm and experimental results. Measurement 2019, 147, 106689. [CrossRef]

178. Wu, X.; Tian, S.; Zhang, L. The Internet of Things Enabled Shop Floor Scheduling and Process Control Method Based on Petri Nets. IEEE Access 2019, 7, 27432–27442. [CrossRef]

179. Hu, L.; Liu, Z.; Hu, W.; Wang, Y.; Tan, J.; Wu, F. Petri-net-based dynamic scheduling of flexible manufacturing system via deep reinforcement learning with graph convolutional network. J. Manuf. Syst. 2020, 55, 1–14. [CrossRef]

180. Vital-Soto, A.; Azab, A.; Baki, M.F. Mathematical modeling and a hybridized bacterial foraging optimization algorithm for the flexible job-shop scheduling problem with sequencing flexibility. J. Manuf. Syst. 2020, 54, 74–93. [CrossRef]

181. Zhu, Z.; Zhou, X. An efficient evolutionary grey wolf optimizer for multi-objective flexible job shop scheduling problem with hierarchical job precedence constraints. Comput. Ind. Eng. 2020, 140, 106280. [CrossRef]

182. Xie, Z.; Zhang, X.; Xia, Y.; Xin, Y. A Multi-Rule Algorithm for Multi-Shop Integrated Scheduling Problem. Design Eng. 2020, 1, 968–979.

183. Wang, Z.; Gombolay, M. Learning Scheduling Policies for Multi-Robot Coordination With Graph Attention Networks. IEEE Robot. Autom. Lett. 2020, 5, 4509–4516. [CrossRef]

184. Mejia, G.; Lefebvre, D. Robust scheduling of flexible manufacturing systems with unreliable operations and resources. Int. J. Prod. Res. 2020, 58, 6474–6492. [CrossRef]

185. Pan, L.; Yang, B.; Jiang, J.; Zhou, M. A Time Petri Net With Relaxed Mixed Semantics for Schedulability Analysis of Flexible Manufacturing Systems. IEEE Access 2020, 8, 46480–46492. [CrossRef]

186. Tellache, N.E.H.; Boudhar, M. Two-machine flow shop problem with unit-time operations and conflict graph. Int. J. Prod. Res. 2014, 55, 1–16. [CrossRef]

187. Tellache, N.E.H.; Boudhar, M. The two-machine flow shop problem with conflict graphs. IFAC PapersOnLine 2016, 49, 1026–1031. [CrossRef]

188. Khouider, A.; Haddadène, H.A.; Ouarri, S.; Oulamara, A. Mixed graph colouring for unit-time scheduling. Int. J. Prod. Res. 2014, 55, 1–10. [CrossRef]

189. Ilani, H.; Shufan, E.; Grinshpoun, T. Partially concurrent open shop scheduling with integral preemptions. Ann. Oper. Res. 2017, 259, 157–171. [CrossRef]

190. Grinshpoun, T.; Ilani, H.; Shufan, E. The representation of partially-concurrent open shop problems. Ann. Oper. Res. 2015, 252, 455–469. [CrossRef]

191. Cheng, Y.; Bi, L.; Tao, F.; Ji, P. Hypernetwork-based manufacturing service scheduling for distributed and collaborative manufacturing operations towards smart manufacturing. J. Intell. Manuf. 2018, 31, 1707–1720. [CrossRef]

192. Ilani, H.; Grinshpoun, T.; Shufan, E. Bounded colouring motivated by the limited resource partially concurrent open shop problem. Ann. Oper. Res. 2019, 1–16. [CrossRef]

193. Shi, X.; Long, W.; Li, Y.; Deng, D.-S. Multi-population genetic algorithm with ER network for solving flexible job shop scheduling problems. PLoS ONE 2020, 15, e0233759. [CrossRef]

194. Rossi, A.; Soldani, S.; Lanzetta, M. Hybrid stage shop scheduling. Expert Syst. Appl. 2015, 42, 4105–4119. [CrossRef]

195. Ciro, G.C.; Dugardin, F.; Yalaoui, F.; Kelly, R. A fuzzy ant colony optimization to solve an open shop scheduling problem with multi-skills resource constraints. IFAC PapersOnLine 2015, 48, 715–720. [CrossRef]

196. Cruz-Chávez, M.A. Neighbourhood generation mechanism applied in simulated annealing to job shop scheduling problems. Int. J. Syst. Sci. 2014, 46, 1–13. [CrossRef]

197. Amirghasemi, M.; Zaman, R. An effective asexual genetic algorithm for solving the job shop scheduling problem. Comput. Ind. Eng. 2015, 83, 123–138. [CrossRef]

198. Nouri, H.E.; Driss, O.B.; Ghedira, K. Hybrid metaheuristics for scheduling of machines and transport robots in job shop environment. Appl. Intell. 2016, 45, 808–828. [CrossRef]

199. Kuhpfahl, J.; Bierwirth, C. A study on local search neighborhoods for the job shop scheduling problem with total weighted tardiness objective. Comput. Oper. Res. 2016, 66, 44–57. [CrossRef]

200. Nouri, H.E.; Driss, O.B.; Ghedira, K. Simultaneous scheduling of machines and transport robots in flexible job shop environment using hybrid metaheuristics based on clustered holonic multiagent model. Comput. Ind. Eng. 2016, 102, 488–501. [CrossRef]

201. Zhao, F.; Shao, Z.; Wang, J.; Zhang, C. A hybrid differential evolution and estimation of distribution algorithm based on neighborhood search for job shop scheduling problems. Int. J. Prod. Res. 2015, 54, 1039–1060. [CrossRef]

202. Shahzad, A.; Mebarki, N. Learning Dispatching Rules for Scheduling: A Synergistic View Comprising Decision Trees, Tabu Search and Simulation. Computers 2016, 5, 3. [CrossRef]

203. Huang, S.; Tian, N.; Ji, Z. Particle swarm optimization with variable neighborhood search for multiobjective flexible job shop scheduling problem. Int. J. Model. Simul. Sci. Comput. 2016, 7, 1650024. [CrossRef]

204. Ge, H.; Sun, L.; Chen, X.; Liang, Y. An Efficient Artificial Fish Swarm Model with Estimation of Distribution for Flexible Job Shop Scheduling. Int. J. Comput. Intell. Syst. 2016, 9, 917–931. [CrossRef]

205. Yang, Q.; Liu, J.; Huang, Y.; Wang, Y.; Wang, T. The dynamic 4S auto maintenance shop scheduling in a multi-constraint machine environment based on the theory of constraints. Int. J. Adv. Manuf. Technol. 2015, 83, 1773–1785. [CrossRef]

206. Elmi, A.; Topaloglu, S. Multi-degree cyclic flow shop robotic cell scheduling problem: Ant colony optimization. Comput. Oper. Res. 2016, 73, 67–83. [CrossRef]
207. Jin, L.; Tang, Q.; Zhang, C.; Shao, X.; Tian, G. More MILP models for integrated process planning and scheduling. *Int. J. Prod. Res.* 2016, 54, 4387–4402. [CrossRef]

208. Knopp, S.; Dauzère-Pérès, S.; Yugma, C. A batch-oblivious approach for Complex Job-Shop scheduling problems. *Eur. J. Oper. Res.* 2017, 263, 50–61. [CrossRef]

209. Sotskov, Y.N.; Gholami, O. Mixed graph model and algorithms for parallel-machine job-shop scheduling problems. *Int. J. Prod. Res.* 2014, 55, 1–16. [CrossRef]

210. Bürgy, R. A neighborhood for complex job shop scheduling problems with regular objectives. *J. Sched.* 2017, 20, 391–422. [CrossRef]

211. Hao, X.; Gen, M.; Lin, L.; Suer, G.A. Effective multiobjective EDA for bi-criteria stochastic job-shop scheduling problem. *J. Intell. Manuf.* 2017, 28, 833–845. [CrossRef]

212. Zuo, Y.; Gong, M.; Jiao, L. Adaptive multi-meme algorithm for flexible job shop scheduling problem. *Nat. Comput.* 2016, 16, 677–698. [CrossRef]

213. Xiong, H.; Fan, H.; Jiang, G.; Li, G. A simulation-based study of dispatching rules in a dynamic job shop scheduling problem with batch release and extended technical precedence constraints. *Eur. J. Oper. Res.* 2017, 257, 13–24. [CrossRef]

214. Tamssaaouet, K.; Dauzère-Pérès, S.; Yugma, C. Metaheuristics for the job-shop scheduling problem with machine availability constraints. *Comput. Ind. Eng.* 2018, 125, 1–8. [CrossRef]

215. Yu, J.-M.; Lee, D.-H. Scheduling algorithms for job-shop-type remanufacturing systems with component matching requirement. *Comput. Ind. Eng.* 2018, 120, 266–278. [CrossRef]

216. Xiong, W.; Fu, D. A new immune multi-agent system for the flexible job shop scheduling problem. *J. Intell. Manuf.* 2015, 29, 857–873. [CrossRef]

217. Nagata, Y.; Ono, I. A guided local search with iterative ejections of bottleneck operations for the job shop scheduling problem. *Comput. Oper. Res.* 2019, 120, 15–36. [CrossRef]

218. Liu, N.; Zhang, Y.F.; Lu, W.F. Improving Energy Efficiency in Discrete Parts Manufacturing System Using an Ultra-Flexible Job Shop Scheduling Algorithm. *Int. J. Precis. Eng. Manuf. Technol.* 2019, 6, 349–365. [CrossRef]

219. Burdett, R.L.; Corry, P.; Yarlagadda, P.K.; Eustace, C.; Smith, S. A flexible job shop scheduling approach with operators for coal export terminals. *Comput. Oper. Res.* 2019, 104, 15–36. [CrossRef]

220. Cruz-Chavez, M.A.; Afsar, S.; Palacios, J.J.; González-Rodríguez, I.; Puente, J. Evolutionary tabu search for flexible due-date satisfaction in fuzzy job shop scheduling. *Comput. Oper. Res.* 2020, 119, 104931. [CrossRef]

221. Ahmadian, M.M.; Salehipour, A.; Cheng, T. A meta-heuristic to solve the just-in-time job-shop scheduling problem. *Eur. J. Oper. Res.* 2021, 288, 14–29. [CrossRef]