Heuristic and Metaheuristic Methods for the Parallel Unrelated Machines Scheduling Problem: A Survey

Marko Đurasević[1*] and Domagoj Jakobović[1]

[1*]University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, Zagreb, 10000, Croatia.

*Corresponding author(s). E-mail(s): marko.durasevic@fer.hr;
Contributing authors: domagoj.jakobovic@fer.hr;

Abstract

Scheduling has an immense effect on various areas of human lives, be it though its application in manufacturing and production industry, transportation, workforce allocation, or others. The unrelated parallel machines scheduling problem (UPMSP), which is one of the various problem types that exist, found its application in many areas like manufacturing and distributed computing. Due to the complexity of the problem, heuristic and metaheuristic methods have dominantly been applied for solving it. Although this problem variant did not receive much attention as other models, recent years saw the increase of research dealing with the UPMSP. During that time, different problem variants, solution methods, and interesting research directions were considered. However, no study provided a systematic overview of the research in which heuristic methods are applied for solving the UPMSP. This comes as a problem since it is becoming difficult to keep track of all the relevant research directions and solution methods considered for this problem. Therefore, the goal of this study is to provide an extensive literature review on the application of heuristic and metaheuristic methods for solving the UPMSP. Each reviewed study is briefly described based on the considered problem and solution method. Additionally, studies dealing with similar problems are grouped together to outline the evolution of the research, and possible areas where further research can be carried out. All studies were systematised and classified into several categories to allow for an easy overview of different problem and solution variants. Finally, recent research trends and possible future directions are also outlined.

Keywords: Unrelated parallel machines, scheduling, dispatching rules, metaheuristics, heuristics

1 Introduction

Scheduling is defined as the process of allocating activities to scarce resources in order to optimise one or more user defined criteria (Leung, 2004). It takes many forms and appears in a multitude of real world situations. Some examples include scheduling products in manufacturing, processes in cloud and grid environments, workers in different industries (nurses, drivers, airline crews), and more (Leung, 2004; Pinedo, 2008). It is evident that scheduling directly affects the everyday life of most people. Therefore, it is of great importance to improve the efficiency of scheduling methods not only to increase the production and decrease the cost in manufacturing, but also to improve the general satisfaction of people and quality of life. A great deal of research has focused on developing new and improving existing methods to more efficiently solve scheduling problems.

Scheduling problems can be solved using various methods, which can be roughly grouped into three categories: exact, approximate, and heuristic. Exact methods provide an optimal solution to the problem by traversing the entire search space and guaranteeing that a better solution does not exist. However, such methods are computationally expensive, which makes them useful only for smaller problems and in situations where the computation time required to obtain the solutions is not restricted. Approximation methods have a smaller computational complexity than exact methods, but still provide a guarantee that the obtained solution is within a given margin from the optimal one. These methods need to be mathematically sound, which makes it difficult to develop such algorithms for different problem variants. Finally, heuristic methods use some well designed rules to construct the schedules. However, these methods do not provide any guarantee on the optimality of the obtained solutions. Because of that, they are the most flexible and easiest to design. Since heuristic and metaheuristic methods demonstrated their efficiency in solving various kinds of optimisation problems, like feature selection (Xue, Tang, Xu, Liang, & Neri, 2022), neural network development (Xue, Jiang, Neri, & Liang, 2021; Xue, Wang, Liang, & Slowik, 2021), manifold learning (Lensen, Xue, & Zhang, 2021) and others (Koza, 2010), their popularity continued to increase and they became widely accepted and applied.

Since there are many variants of scheduling problems, they have been categorised into different models. For example, timetabling deals with scheduling students and teachers to events (lectures, exams) in schools and universities, rostering deals with scheduling employees to different shifts, job-shop deals with production environments in which a product needs to go through a series of modifications until it is completed, and the parallel machine environment deals with problems where each activity can be processed by any

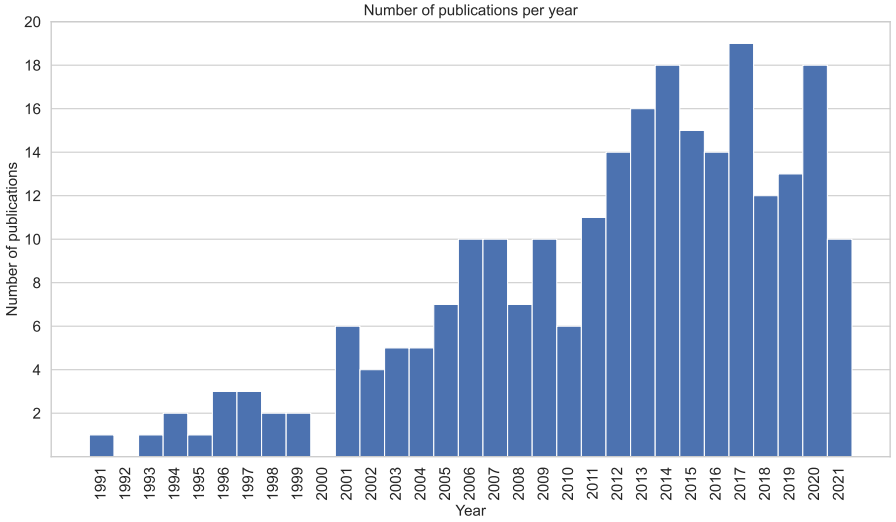

Fig. 1 Distribution of papers in the last 30 years

of the available resources. The parallel machine environment can further be divided into three categories: identical, uniform, and unrelated. Out of those, the unrelated parallel machines scheduling problem (UPMSP) is the most general, as it makes the least assumptions regarding the resources that process activities. Although this environment has applications in many areas like multiprocessor task scheduling (L. Wu & Wang, 2018), equipment scheduling (Gedik, Kalathia, Egilmez, & Kirac, 2018), and manufacturing (Yu, Shih, Pfund, Carlyle, & Fowler, 2002), it received less attention than some other environments. However, recently there have been many new studies dealing with this problem. Therefore, the number of problem variants that were considered in the literature and the solution methods applied for solving them has grown significantly. This is especially true for the solution methods, where the earlier research was more focused on exact and approximate methods. Recent years have seen the rise in application of heuristic and metaheuristic methods, as larger and more complex problems have been considered. This can best be seen from Figure 6 that shows the distribution of papers in which heuristics were used for solving the UPMSP during the last 30 years. The last 10 years show a rising trend in the number of studies which apply heuristic methods for solving the UPMSP. Such a trend is expected to continue over the following years, which makes the systematisation of existing research important to lay out a foundation for future studies and research directions.

Because of the large number of problem variants and solution methods, several survey papers provide an overview of the literature dealing with various types of scheduling problems. For example, an overview of staff scheduling and rostering was provided by Ernst, Jiang, Krishnamoorthy, and Sier (2004). Timetabling problems have been surveyed by Babaei, Karimpour,

and Hadidi (2015). Studies dealing with flexible job shop problems have been surveyed by Chaudhry and Khan (2015). An older, but detailed survey of classical scheduling problems has been provided by B. Chen, Potts, and Woeginger (1998). Other papers have reviewed problems with specific properties, like setup times (Allahverdi, 2015; Allahverdi, Gupta, & Aldowaisan, 1999; Allahverdi, Ng, Cheng, & Kovalyov, 2008) or no-wait in processing (Allahverdi, 2016). Others reviewed different solution methods used for solving scheduling problems like evolutionary algorithms (Hart, Ross, & Corne, 2005), automatically generated DRs (Branke, Nguyen, Pickardt, & Zhang, 2016; Nguyen, Mei, & Zhang, 2017), or multi-population based heuristics (Lei & Cai, 2020). However, these surveys did not focus on one single scheduling problem type, but rather on various problem types with common properties (setup times) or solution methods (evolutionary algorithms). Although some of those surveys included studies on the UPMSP, this problem was not in the focus of any of them. To the best of our knowledge, the UPMSP has only been surveyed in the papers of T. Cheng and Sin (1990), B. Chen et al. (1998), and Mokotoff (2001). However, these studies did not focus exclusively on the UPMSP, but rather on the more general parallel machine environment, which also includes the identical or uniform machine environments. Although these studies provided a good literature report of the parallel machine environments at the time of their writing, as 20 or more years have passed since their publication they are out of date and do not provide an overview of all the recent relevant studies. This is especially true for studies in which heuristic methods are applied for solving the UPMSP, since Figure 6 demonstrated that research in this area flourished after year 2000. Therefore, we can identify a huge gap in the literature during which heuristic methods for solving the UPMSP have not been surveyed at all. Because of the increase of research in this area during the last years, and since such a trend is expected to continue in the future, it is becoming more difficult to follow which approaches were investigated, and what represents the current state of the art in this field. This increases the possibility of similar research being conducted, or not considering the latest methods or results obtained by other researchers. The previous points outline the need for a systematised overview of the research considering the application of heuristic methods for the UPMSP.

In order to fill the previously outlined gap in the literature, the goal of this study is to provide a detailed survey of the application of heuristic and metaheuristic methods for solving the UPMSP. This survey considers studies which satisfy the following two criteria:

1. The UPMSP must be in the focus of the paper, although other problems can also be considered. Problems in which the UPMSP represents a subproblem of another problem, like in the flexible job shop environment, are not considered.
2. The study needs to apply one or more heuristic methods for solving the UPMSP. Other methods can also be presented besides heuristic

methods, but papers applying only exact or approximation methods are not considered.

The research surveyed in this paper is classified by two aspects, the considered problem variant and the methods applied for solving the problem. All the problems are categorised using the standard notation of scheduling problems introduced by Graham, Lawler, Lenstra, and Kan (1979) to allow an easy perusal across the existing research, and enabling the readers to find studies dealing with research areas interesting to them. Regarding the applied solution methods, they are also divided into several groups to show which methods have commonly been applied. Therefore, this survey provides an overview that allows to easily detect which methods have been applied on a considered problem variant, or all the problems for which a certain method has been applied until now. Such a systematised overview should help other researchers to obtain a thorough overview of the research performed on the UPMSP, and hopefully broaden it and lead it into new directions. The main contributions of this study can be summarised in the following points:

1. The first extensive survey of heuristic methods applied for solving the UPMSP covering the existing literature thus far.
2. Providing a short overview of research performed in each study and connecting similar or relevant studies where possible.
3. Classification of all existing studies using the standard notation for scheduling problems, and their categorisation into relevant groups.
4. Classification of existing studies based on the solution methods applied in each of them.
5. Outlining existing research gaps and potential research directions for future studies.

The remainder of the paper is organised as follows. Section 2 provides the description of the UPMSP and the notation used throughout the survey. The literature overview of heuristic methods applied for solving the UPMSP is provided in Section 3. The reviewed papers are additionally classified into several categories in Section 4. An overview of certain recent trends in the problem and possible future directions is given in Section 5. Finally, Section 6 provides the conclusion of the paper.

2 Problem description and notation

This section will shortly describe the classical scheduling problem variants. Furthermore, constraints and optimisation criteria that are considered in the reviewed studies are enumerated and shortly described. Finally, the notation used to denote the different problem variants is introduced.

Scheduling environments in classical scheduling problems can be divided into two main groups, single-stage and multi-stage environments. In multi-stage environments each job needs to be processed by a series of machines (stages) until completion. This group includes environments such as flow

shop, job shop, and open shop. The difference between those environments is whether all jobs need to be processed by all machines in the same predefined order (flow shop), whether they need to be processed on all machines but in different predefined orders (job shop), or whether the order of processing jobs is irrelevant (open shop). The previous environments can further be extended to their flexible counterparts, in which there are several machines that can process each stage of the job. On the other hand, in single stage environments each job needs to be processed by only a single machine before it is completed. This group contains the single and parallel machine environments. As the names already outlines, the single machine environment consists of only a single machine that is used to process jobs, whereas in the parallel machine environment there are several machines available to execute jobs. The parallel machines environment is further split into the identical, uniform, and unrelated machine environments. The difference between these environments lies in the processing speeds of machines. In the identical machines environment all machines are presumed to be identical, and as such it is not important to which machine the job is allocated to since its execution time will be the same on all machines. In the uniform machines environment each machine has a specified speed, thus some machines will process jobs faster than others. Based on these machine speeds, it is possible to build relations between machines to determine which machines are faster than others. Finally, in the unrelated machines environment, the time required to process a job depends both on the job and machine on which it is executed. Therefore, some jobs might execute faster on some machines, whereas others might execute faster on some other machines. As a consequence, it is not possible to build relations between machines, because there is no guarantee that a single machine will execute all jobs faster than all other machines. Thus, this machine environment is the most general variant of all the single stage environments.

Formally, The UPMSP is defined by a set of n jobs, where each job needs to be executed on one of the m available machines. The subscript i is used to refer to a certain machine, whereas the subscript j refers to a job. The most general property which needs to be specified for this environment is the execution time of job j on machine i, which is denoted as p_{ij}. As previously outlined, in the UPMSP it is assumed that this value needs to be specified for all job-machine pairs, and that no machine relations exit from which the processing times can be inferred (for example that one machine is two times faster than another, so it requires half of the time to execute the jobs). Depending on the problem variant, other properties are also defined for jobs. In certain cases jobs need to finish until a given due date d_j. Additionally, in some problems not all jobs are equally important, therefore a weight w_j is defined for each job, which is then used when calculating the objective value of the schedule. When the schedule is constructed, the completion time C_j can be determined for each job, based on which several objective functions can be calculated.

To easily describe scheduling problem, the $\alpha \mid \beta \mid \gamma$ classification scheme is often used (Graham et al., 1979). However, as the number of scheduling

problems increased, the notation also had to be expanded to encompass all the problem variants and optimisation criteria. In this scheme, α represents the machine environment, which in the case of unrelated machines is denoted as R. The second field β represents additional problem constraints and characteristics, and can contain zero or more entries. These include

- setup times (s) – when switching from one job to another on a machine, a setup operation of a certain duration needs to be performed to prepare the machine for processing the next job. The length of this operation is given with s_{ijk}, where i denotes the job which has finished executing, j the job that should be executed next, and k the machine on which the jobs are processed.
- release times (r_j) – jobs are not available immediately at the start of the system, but are released into the system over time. Each job has a release time r_j which denotes when the job becomes available, and the machine cannot start executing the job until it becomes available. This constraint is often used in dynamic scheduling environments where jobs are released over time into the system without prior knowledge when this will happen.
- machine eligibility (M_j) – for each job a subset of machines is defined that can be used for processing that job. A job cannot be scheduled nor executed on a machine that is not eligible for it.
- precedence constraints ($prec$) – jobs are not independent and cannot be executed in an arbitrary order. A job j can have several predecessors, all of which have to be finished before job j can start executing.
- batch scheduling ($batch$) – jobs are grouped into batches which are scheduled. Two variants exist, serial ($s - batch$) and parallel ($p - batch$) batch. In serial batch scheduling, all jobs are executed one after another, usually with no setup required between the jobs of the same batch and thus the execution is faster than executFng jobs of different batches one after another. In parallel batch scheduling, jobs are executed simultaneously on a machine (each machine can execute several jobs at the same time), and the execution time of the batch is equal to the longest execution time of any job in the batch. There are different variants of this constraint, in which jobs can have different sizes, machines can have different capacities, and similar.
- job sizes (J_s) – applicable only for batch scheduling problems. Denotes the size that the job takes up in the batch. If this constraint is not specified, it is presumed that each job has the same size, usually equal to 1.
- machine capacities (Q_k) – applicable only for batch scheduling. Denotes that the capacity of the batch depends on the machine on which it is executed. Thus, some machines can process larger batches of jobs than others.
- deadline ($\bar{d}$) – jobs are required to finish until a given time. Unlike the due date, the deadline is a hard constraint and a schedule in which a job does not finish before its deadline is not considered feasible.
- common due date ($d_j = D$) and common deadline ($\bar{d} = D$) – all jobs have the same common due date or deadline until which they need to finish

executing. The goal is to find the schedule and/or determine the common due date or deadline such that a certain criterion is optimised.

- machine availability constraints ($brkdwn$) – also sometimes referred to as breakdowns. Defines that machines are not available all the time, be it due to expected or unexpected situations. Depending on the concrete problem, the periods when machines are unavailable can be known beforehand (deterministic) and the schedule be constructed considering them, or they can be unknown (stochastic) and the schedule needs to be adapted if a machine becomes unavailable during the execution of the system.
- auxiliary resources (R) – defines that additional resources are required during machine setup or job execution. Therefore, a job cannot start with its execution unless a predefined value of certain resources is available. Resources can be in the form of workers which need to perform some adaptations, or different material resources. In addition, resources can be renewable (they are restored over time) or non-renewable (once used, they cannot be restored).
- changing processing times (p_c) – processing times of jobs are adaptable and can change during time. They increase either due to job deterioration (the longer the jobs are in the system, the longer will they require to be completed), or decrease due to learning effects or by using additional resources (workers get better over time and perform the jobs or setups more efficiently)
- dedicated machines (M_{ded}) – jobs have machines that are dedicated for them, meaning that they are executed faster on those machines than on other machines. For example, this could be due to some special hardware installed at the machine which improves the execution times of those jobs.
- rework processes ($rwrk$) – after execution it is possible that jobs are faulty and have to be reworked again on one of the machines. Depending on the problem, it is possible that the job needs to be reworked once or several times. This constraint is usually stochastic, meaning that it is not known in advance which jobs will be faulty.
- machine deterioration (M_d) – machines deteriorate over time and the execution of jobs becomes slower or their capacity is lower. This can be fixed by applying maintenance activities to machines, which restore them to their original state.
- machine maintenance (M_m) – maintenance activities have to be allocated to machines either to keep the system at a desired level (preventive maintenance) or to fix broken machines (corrective maintenance). Two variants are usually investigated, one in which the maintenance periods are specified in advance and cannot be changed, rather jobs have to be scheduled around those periods. In the second, maintenance periods can be freely allocated to machines at any time at which they are not executing a job.
- machine speed (M_s) – machines can run with different speeds to process jobs faster. However, this usually requires the usage of additional resources or more energy is consumed in such operating scenarios.

- load (L) – jobs need to be loaded and transported to machines using a vehicle with a constrained capacity.

The γ field represents the criteria that are optimised. This field must have at least one entry, but can also have more in the case of multi-objective optimisation. The optimisation criteria include:

- makespan (C_{max}) – represents the latest completion time of a job $C_{max} = \max_j C_j$. Minimising this criterion is directed towards reducing the length of the schedule.
- total weighted flowtime (Fwt) – represents the total weighted time each job spent in the system $Fwt = \sum w_j F_j$, where $F_j = C_j - r_j$. A special case is the total flowtime $Ft = \sum F_j$, where all weights are equal to 1. This criterion is equivalent to the total weighted completion time $Cwt = \sum w_j C_j$. By minimising this criterion the system tries to lower the time that the jobs will be kept in the system and tries to execute them as soon as possible.
- maximum flowtime (F_{max}) – the maximum flowtime value of all jobs $F_{max} = \max_j F_j$. By optimising this criterion the maximum amount of time that a job is kept in the system is reduced.
- total weighted tardiness (Twt) – represents the sum of weighted tardiness of all jobs $Twt = \sum w_j T_j$, where $T_j = \max(C_j - d_j, 0)$. A special case is the total tardiness (Tt) when all the job weights are equal to 1. Minimising this criterion reduces the amount of time that jobs spent executing after their due dates.
- maximum tardiness (T_{max}) – maximum tardiness value of any job $T_{max} = \max_j(T_j)$. Minimising this criterion reduces the maximum amount of time that a job spent executing after its due date.
- weighted number of tardy jobs (Uwt) – $Uwt = \sum w_j U_j$, where U_j equals 1 if a job finished after its due date. A special case is the number of tardy jobs (Ut) when all weights are equal to 1. Minimising this criterion reduces the number of jobs that miss their due date.
- total energy consumption (TEC) – represents the total energy consumed during system execution. Concrete definitions of this criterion can vary. For example, in some problems machines consume energy and can be turned off or their speed can be reduced to save energy.
- total weighted earliness and tardiness ($Etwt$) – represents the sum of the weighted earliness and tardiness values for each job $Etwt = \sum w_j \max(d_j - C - j, 0) + \sum w_j \max(C_j - d_j, 0)$. The weight for the earliness and tardiness part can be equal or different. By optimising this criterion all jobs try to finish near to their due dates. The reason for this is that in certain problems products need to be stored if completed prior to their due date or have an expiry date and as such must not be completed too far away from their due date.
- machine load (ML) – considers the load balance across all machines. Usually as the difference of the total execution time between the highest loaded and lowest loaded machines. By optimising this criterion, jobs are equally

distributed across all machines so that all machines spend a similar time on executing jobs.
- Cost ($COST$) – the production cost of certain parts of the system execution. Usually it is optimised with another objective. By minimising this objective the search is directed towards the best schedule which can be achieved by the lowest cost.
- total setup time (Ts) – total time spent on setups.
- Total resources used (R_u) – total amount of resources used in the schedule. This criteria is considered only when additional resources are used, and usually the amount of using these resources needs to be minimised.
- Total late time (L) – total late time of all jobs, which is calculated in the same way as tardiness except if a job started executing after its due date, then the penalty is fixed regardless when it started.
- Number of jobs finished just in time (N_{jit}) – number of jobs that finished exactly at their due dates.

For a better overview of the different problem variants, Figure 2 shows the outline of the classification scheme used in this study.

Beside the problem variants described previously, there are several additional categories based on UPMSPs can be classified. First, depending on the reliability of the parameters, problems can be divided into:

- Deterministic scheduling – values for all parameters (job processing times, setup times, or similar) are known exactly with a satisfactory precision, regardless when they become available. This means that the values for any of the parameters do not change during the execution of the system.
- Stochastic scheduling – exact values for certain parameters might not be known beforehand (for example, the processing time of a job becomes known only when the job completes execution on a machine). However, the parameter values are not completely unknown, and are usually modelled using fuzzy sets or stochastic functions.

Secondly, based on the availability of the parameter values, all problems can be divided into:

- Offline scheduling – all parameters and their values (total number of jobs, their release times, processing times, and similar) are known before the start of the system execution.
- Online scheduling – not all parameters are known prior to the start of the scheduling system, but rather become available during the execution system. For example, jobs are released over time into the system, but it is not known when they will be released or what their properties will be.

Finally, based on the manned in which schedules are constructed, scheduling procedures can be divided into:

- Static scheduling – the schedule is constructed prior to the execution of the system. In this kind of scheduling it is required that all job parameters are

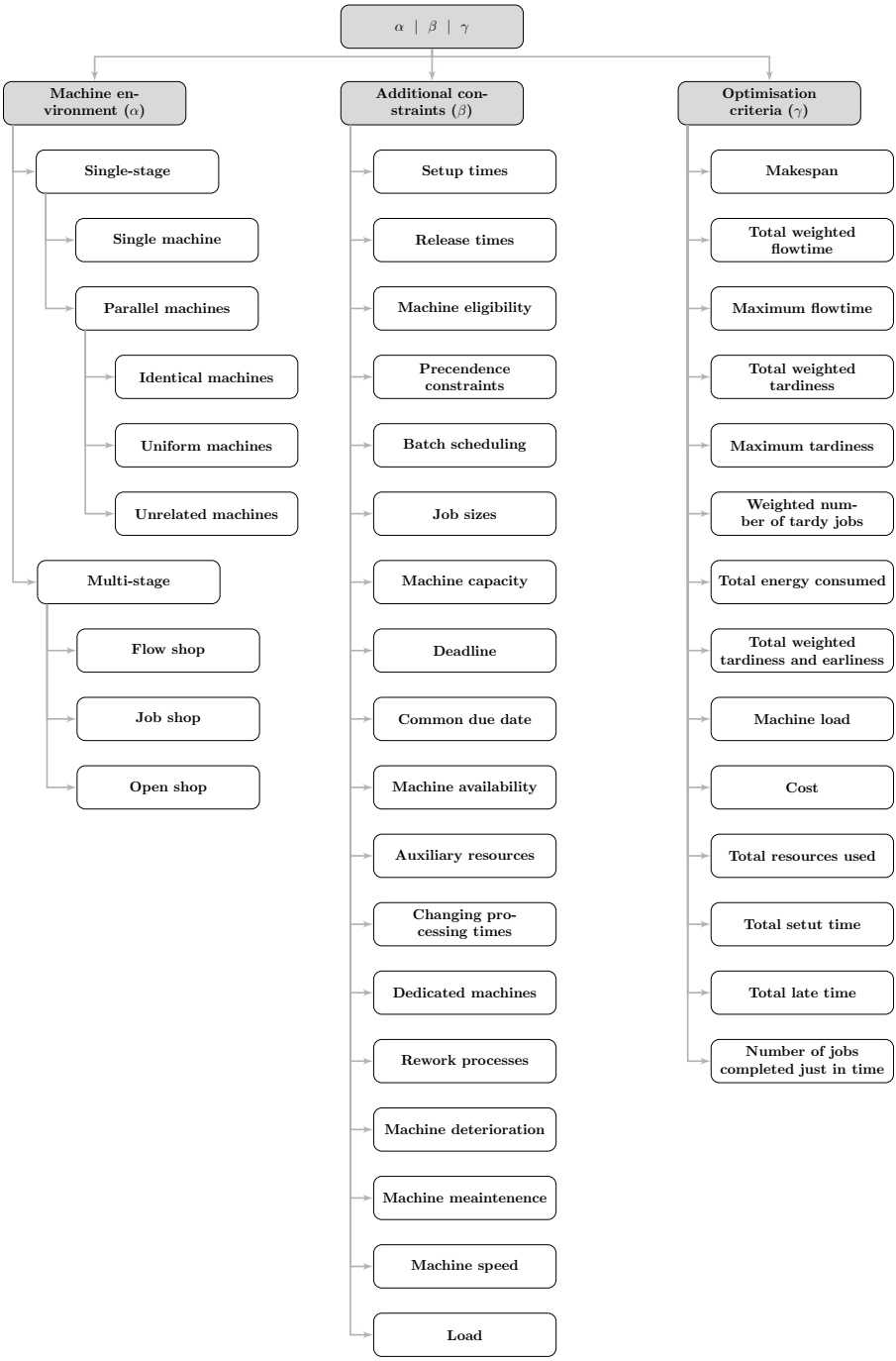

Fig. 2 Classification of problems

available up front, or otherwise the schedule could not be executed in that form as constructed (for example, if some jobs take more time to execute than expected). Most metaheuristic methods construct the solution in this way.

- Dynamic scheduling – the schedule is constructed incrementally in parallel with the execution of the system. Such methods usually only perform the immediate next decision, and do not construct the entire schedule prior than required. In most cases, DRs construct schedules in this way.

3 Heuristic methods for the unrelated machines environment

Over the years a plethora of methods for solving the UPMSP have been proposed in the literature. Figure 3 outlines the classification of solution methods and provides several examples for each group. The methods for solving the considered scheduling problem can be roughly divided into exact (Fanjul-Peyro, 2020), approximation (Lenstra, Shmoys, & Tardos, 1990), and heuristic methods. Exact methods can obtain the optimal solution for the considered problem. The simplest way to obtain such a solution would be to perform an exhaustive search of all solutions. However, other more elaborate methods are often used, like representing the problem as a mixed-integer linear programming (MILP) and solving it using an optimisation procedure, or directly applying methods that restrict the search space like branch and bound. Approximation methods like PTAS provide a guarantee on the distance of the obtained solution from the optimal solution, and can obtain it in polynomial time. A good overview of exact and approximate approaches is given by Wotzlaw (2007).

Heuristic methods provide no guarantee on the optimality of the solution and come in many variants. They can either be problem specific heuristics that were designed for solving the considered problem, or general metaheuristics that can be used for solving any optimisation problem. Problem specific heuristics are defined in different forms, with dispatching rules being the most common type for solving scheduling problems. They can either be manually designed by experts, like the apparent tardiness cost rule (Đurasevic & Jakobovic, 2018), or automatically designed using various learning methods, most notably genetic programming (Đurasevic, Jakobovic, & Knezevic, 2016). On the other hand, a plethora of metaheuristic approaches have been applied for solving the UPMSP. Although they can be classified in different ways, they are most often divided into algorithms that work only with a single solution (simulated annealing or tabu search) and those which use a population of solutions. The second group is further divided into methods that are modelled based on some evolutionary behaviour (like genetic algorithms), or those that are modelled after behaviour of swarms from nature (like ant colony optimisation).

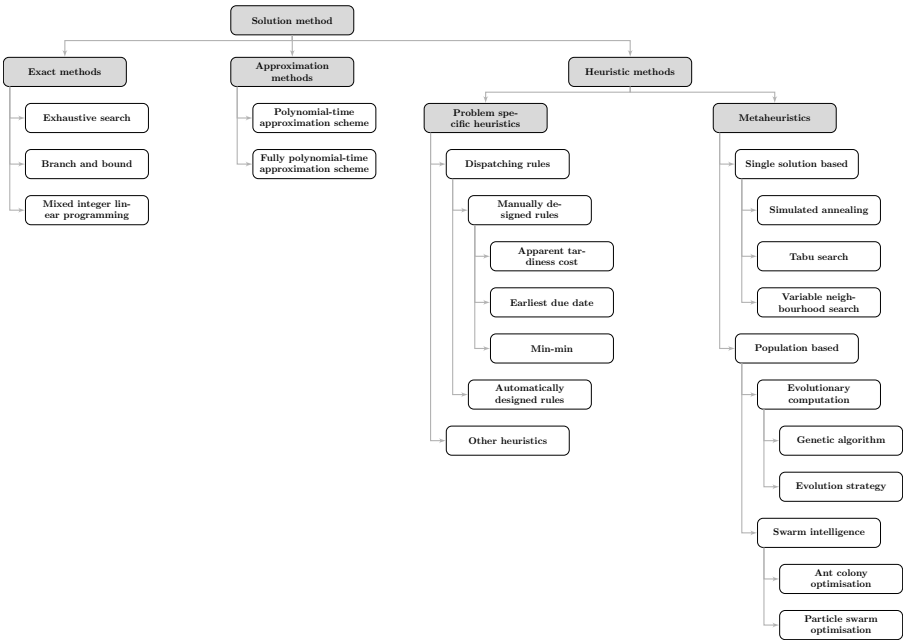

Fig. 3 Classification of solution methods

The rest of this section will provide a detailed review on heuristic methods used for solving the UPMSP. Section 3.1 provides the overview of research considering manually and automatically designed DRs. Other problem specific heuristics that cannot be classified as DRs are reviewed in Section 3.2. Finally, Section 3.3 provides the overview of metaheuristic methods that were applied for solving the UPMSP. Due to the large number of methods and terms that appear in the text, a list of abbreviations is given in Table 1.

3.1 Dispatching rules

In the context of scheduling problems, a special kind of heuristics, denoted as DRs, are often applied (Morton & Pentico, 1993; Panwalkar & Iskander, 1977). These heuristics create the schedule incrementally by assigning jobs on free machines by ranking them using a priority function. Based on their rank, the DR selects which job should be scheduled next. The advantage of such heuristics is that they are simple and fast, which makes them applicable for large or dynamic problems. However, the downside is that they usually cannot match the performance of more complex heuristics. Section 3.1.1 provides the overview of manually designed DRs, whereas Section 3.1.2 provides the outline of DRs that were generated automatically.

Table 1 Abbreviations used in the text

Term	Abbreviation
Ant colony optimisation	ACO
Apparent tardiness cost	ATC
Apparent tardiness cost with setup times	ATCS
Artificial bee colony	ABC
Automatically designed dispatching rule	ADDR
Branch and bound	B&B
Cat swarm optimisation	CSO
Clonal selection algorithm	CLONALG
Combinatorial evolutionary algorithm	CEA
Differential evolution	DE
Dispatching rule	DR
Earliest completion time	ECT
Earliest due date	EDD
Electromagnetism-like algorithm	EMA
Estimation of distribution algorithm	EDA
Evolution strategy	ES
Fixed set search	FSS
Fruit fly algorithm	FA
Greedy randomised adaptive search procedure	GRASP
Genetic algorithm	GA
Genetic programming	GP
Genetic simulated annealing	GSA
Harmony search	HS
Harris hawk optimisation	HHA
Imperialist competitive algorithm	ICA
Intelligent water drop algorithm	IWDA
Iterative descent	ID
Iterated greedy	IG
Iterative local search	ILS
Learning automata	LA
Local search	LS
Longest processing time	LPT
Manually designed dispatching rule	MDDR
Multi-agent	MA
Multi-objective	MO
Non dominated sorting genetic algorithm II	NSGA-II
Particle swarm optimisation	PSO
Path relinking	PR
Record to record travel	RRT
Salp swarm optimisation	SSA
Scatter search	SS
Shortest processing time	SPT
Simulated annealing	SA
Sine-Cosine Algorithm	SCA
Squeky wehll optimisation	SWO
strength pareto evolutionary algorithm	SPEA2
Tabu search	TS
Treshold acceptance	TA
Variable neighbourhood descent	VND
Variable neighbourhood search	VNS
Weighted shortest processing time	WSPT
Whale optimisation algorithm	WOA
Worm optimisation	WO

3.1.1 Manually designed DRs

This section provides an overview of DRs that were designed manually by researchers and experts for different UPMSP variants.

Makespan minimisation

Ibarra and Kim (1977) consider the minimisation of makespan and adapt 5 DRs based on the LPT rule, which schedules jobs with the longest processing time first. The proposed rules include the later commonly used min-min and max-min DRs, which are among the most popular and best performing DRs for makespan minimisation. As such, this study can be considered one of the pioneer studies in the field of UPMSP, as it was one of the first to examine DRs for that problem. Furthermore, the two proposed DRs have been used as a baseline in various studies in the future which demonstrates their importance and influence on the field. De and Morton (1980) expanded the previous research by proposing a new DR that first approximates the makespan by using some other DR. Then, when certain machines reach a load equal to the makespan multiplied by a certain factor, the rule does not consider these machines in further scheduling decisions. In that way, the DR distributes the load evenly across all machines. Although the proposed rule represents an interesting concept, due to the requirement that the makespan is approximated it is applicable only in static environments, which limits its usability.

Hariri and Potts (1991) propose a two phase method denoted as LP/ECT for minimising the makespan, which applies linear programming (LP) to construct a partial schedule and then the ECT rule to schedule the remaining jobs. This method achieved a better performance than using ECT by itself, but with a significantly larger execution times. Furthermore, an improvement procedure was applied to all methods to show that the results of ECT can be significantly improved and match the performance of LP/ECT. This improvement procedure can be regarded as an IG algorithm that applies simple neighbourhood operators on the solution obtained by ECT. However, the considered methods are not applicable to dynamic problems or problems which need to be solved fast, due to their increased execution time. The importance of this study is that the authors demonstrated that different methods (LP, DRs, LS) can effectively be used in synergy to achieve an even better performance.

An extensive comparison of DRs for minimising the makespan was performed by Maheswaran, Ali, Siegel, Hensgen, and Freund (1999). The authors compare 5 existing and propose 3 novel DRs, one of which is the sufferage rule that can be considered one of the best rules for makespan minimisation, and has later on been often used for benchmarking other novel DRs. The experiments were performed on 4 problem sets that differ in machine and job heterogeneity, which specify the difference in magnitudes between the processing times of jobs across the machines. The results demonstrate that the proposed sufferage rule is superior to other rules across all the tested scenarios, which is a valuable contribution as later studies demonstrated sufferage to be among the overall best rules when the makespan is considered

(Đurasevic & Jakobovic, 2018). T. Braun et al. (1999) and T.D. Braun et al. (2001) expand the previous study by considering 11 methods for makespan minimisation. The considered methods include 5 DRs, a GA, SA, GSA, TS, and A*, all of which were examined on problems with different heterogeneity properties. The results demonstrate that GA achieved the best performance, closely followed by the min-min rule. However, it also demonstrated the sheer difference in the execution times between metaheuristics and DRs, outlining that metaheuristics are significantly more computationally expensive. The previous studies are important in the sense that they perform an in depth comparison of different DRs to determine under which conditions each of them performs the best, but also provide a comparison of the more general metaheuristic methods to outline the gap between between them and the benefits of each.

M.-Y. Wu and Shu (2001) provide an analysis of the existing min-min rule, based on which they propose the relative cost (RC) rule. This rule takes into account both processing times and completion times of jobs on all machines and balances between the two measures when scheduling jobs. The proposed rule exhibits a better performance than other tested DRs but also against a GA. Such a result is surprising as in subsequent studies metaheuristics outperformed DRs, which probably denotes that the GA was not appropriately adapted for the considered problem. Du Kim and Kim (2004) propose a novel DR called minimum execution completion time (MECT). This rule represents a combination of two simple DRs and alternatively uses the execution and completion times when scheduling jobs. The rule selects jobs by their processing times if this does not increase the makespan, otherwise it uses the completion times of jobs to select the next one. As such, this rule can be considered as a simple adaptive rule that based on the conditions of the system selects the better rule, i.e. it switches between the minimum execution time (MET) and minimum completion time (MCT) rules. The rule achieved a better performance than the individual rules used to construct it, but the authors did not compare to other rules used for makespan minimisation.

Munir, Li, Shi, Zou, and Yang (2008) propose a novel DR called MaxStd. The goal of this DR is to prioritise jobs with a high standard deviation of their processing times, since these jobs could suffer the most if not scheduled on the right machine. The DR is applied for minimising the makespan and is compared to 5 existing DRs, like min-min or max-min. The experiments demonstrate that it achieves slightly better performance than other existing rules. e Santos and Madureira (2014) present a new DR called OMCT, in which jobs are sorted according to a priority calculated based on their sufferage values and standard deviation of processing times. Each job is then scheduled on the machine on which it will complete the soonest. The rule was compared only to the simple MCT rule and tested only on very small problem instances, which did not provide a good overview of its quality. In the previous four studies the experimental setup was different and all the methods were not compared with each other, which makes it impossible to compare their results based solely

on those studies. However, in a recent study (Đurasevic & Jakobovic, 2018) all these methods were compared, and the results demonstrated that RC and MECT usually achieved the best results, whereas MaxStd and OMCT achieved the worst. Both the RC and MECT rules are based on the concept that only a single property, e.g. completion or processing times of jobs, is not enough to construct good schedules, but rather that good rules need to take several job properties into account. Since MECT is the simplest among the tested DRs, this can serve as the motivation to focus on the design of simple DRs, since obviously they can perform equally well or even better than more complex ones.

A set of 20 DRs, 17 of which are novel, is analysed by Luo, Lü, and Shi (2007). All newly proposed DRs follow the same structure, since they first select the best machine for each job based on one rule, and then among all the pairs select the job that will be scheduled on its selected machine using a second rule. In addition, the integration of a task priority graph based on the consistency between jobs and machines into the rules is proposed. The authors provide an extensive analysis of all the proposed algorithms and show that taking task consistency into account improves the performance over standard DRs like min-min and max-min, which do not use such information. Therefore, this study provides motivation for trying to introduce more problem specific information into DRs to improve their performance. However, to construct the task priority graph, all the information about the system needs to be available, which limits the application of this method in dynamic environments. An alternative would be to build the graph partially, only based on available jobs, but it is not known what an effect this would have on the performance of the approach.

An analysis of 6 DRs from the literature is performed by Briceño, Siegel, Maciejewski, and Oltikar (2012). The authors propose an iterative procedure to minimise the total execution time of jobs on non-makespan machines. This is done by creating an initial schedule, removing the makespan machine and all jobs allocated to it from the problem, and trying to create a new schedule for the reduced problem with a DR. Although the idea is interesting and sounds intuitive, the experimental results showed that there is no guarantee that such a procedure results in smaller completion times on non makespan machines. Additionally, such a method would hardly be usable in dynamic situations where the characteristics of the system change constantly, and as such the schedule cannot be reconstructed. As such, this method has a limited applicability in DRs, especially as metaheuristics should also be able to easily solve such a problem.

Minimisation of flowtime and makespan related criteria

The following two studies focus on minimising flowtime related criteria. Weng, Lu, and Ren (2001) consider the minimisation of the total weighted flowtime with setup times, and propose 7 DRs based on the WSPT rule for the single machine environment. They demonstrate that the DR which orders jobs based

on the ratio between the processing time plus setup times and the job weight performs best among the tested rules. Although simple, the DR showed its efficiency in solving a wide range of problems. Strohhecker, Hamann, and Thun (2016) consider the same problem but with flowtime minimisation, and test several DRs in different scenarios (with varying processing time and setup time speeds) to determine their behaviour in different situations. The authors examine different simple rules for performing the assignment of jobs to machines and then sequencing of jobs on machines. They demonstrate that the best DR strategies obtain results close to optimal and can improve the overall production performance. Although a detailed study on different scenarios is performed, the authors unfortunately did not compare with methods that take into account both sequencing and assignment decisions simultaneously to demonstrate whether considering both decisions at the same time can help improve the results.

A novel DR called min-max is proposed by Izakian, Abraham, and Snasel (2009), which is used for optimising the makespan and total flowtime. This rule adapts the min-min heuristic in a way that it selects jobs based on the ratio of their minimum execution time and the execution time on the selected machine. In that way the rule tries to schedule jobs on machines on which they execute more quickly. The rule usually outperforms other rules for makespan, whereas for flowtime the DR came second. Rafsanjani and Bardsiri (2012) examine the same problem and propose an extended sufferage rule. This rule works in the same way as sufferage, however, it scales the priority values of jobs with a quotient between the processing time and completion times of jobs. In that way it can better judge whether a machine is appropriate for executing a job. The rule is compared to several others and achieves the best results for makespan, while for flowtime it comes second. A later study (Đurasevic & Jakobovic, 2018) demonstrated that the proposed sufferage rule is the best rule for makespan minimisation, followed closely by the min-max rule. Both of these rules demonstrate that by augmenting their decision process with additional information about the system better results can be obtained, which is a similar conclusion as with the MECT and RC rules. This suggests that to obtain better rules it is required to design more sophisticated decisions when selecting which jobs to schedule. Unfortunately, both rules were rarely or never used for comparisons in later studies, even though the results suggest that they perform better than many other rules.

Minimisation of due date based criteria

RANDHAWA and SMITH (1995) examine how different design choices in DRs affect their performance. The authors examine decisions like ranking jobs on machines (using the LPT or EDD rules), assigning jobs to machines and similar. Setup times were defined for jobs and three objectives were optimised, the flowtime, number of tardy jobs, and machine load. Based on the experiments, the authors outline the important design choices for each objective. For example, the authors outline that the system load and setup

times had the most influence on the examines objectives, whereas the relations between machine speeds had a smaller effect and thus the authors outlined that they need not be considered. The analysis provided by the authors makes it more clear as to how certain system parameters and decisions in the scheduling process affect the values obtained for the considered criteria, which can be a valuable knowledge for other practitioner.

Golconda, Dogan, and Özgüner (2004) propose a DR for optimising the number of tasks which meet their due dates, often called quality of service. The motivation for this problem comes from real-time systems in which it is required that tasks complete until their respective deadlines, otherwise the tasks are interrupted and do not execute until the end. The authors examine several DRs from the literature and analyse them with the respect to the number of tasks that complete before their deadline. The main contribution of this study is the evaluation of existing DRs for a problem modelled based on real-time systems. Unfortunately, such a problem in which jobs are dropped if they do not meed their due date were seldom researched, and later studies focused almost exclusively on problems in which jobs would continue executing and only the tardiness would be minimised. J.-F. Chen (2009) propose a DR for problems with setup times and total weighted tardiness minimisation. The authors modify the ATCS rule, which was initially designed for identical parallel machines, for the unrelated machines environment. Although the study focused dominantly on proposing a SA method, it is significant because of the adaptation of the ATC rule for the UPMSP, since ATC became one of the most effective and widely used rules for minimisation of tardiness based criteria. Tseng, Chin, and Wang (2009) compare the performance of 5 DRs when considering setup times and three objectives, the makespan, total weighted tardiness, and computing cost. The authors extend the ATCS rule with the MCT strategy to improve its performance, which results in a new rule that outperforms all other rules for the total weighted tardiness criterion. In some cases the results also demonstrate that the ATCS rule achieved a better result than other DRs for the makespan criterion. This result is somewhat strange in the sense that the ATC rule can perform equally well as other DRs designed especially for makespan minimisation. Such an observation is not consistent with the results of a subsequent study (Đurasevic & Jakobovic, 2018), but it could be due to the way in which the problem instances were constructed.

Yang-Kuei and Chi-Wei (2013) perform an analysis of several DRs for scheduling jobs with release times and optimising the makespan, weighted flowtime, and weighted tardiness criteria. For each criterion the authors propose a DR that is further improved by additional job reassignment procedures, which improve the schedule. The authors also propose the application of the ACTR rule for the UPMSP, which takes into account all the jobs that need to be scheduled when calculating their priorities. However, the rules proposed in this study are only applicable to static scheduling problems, and the authors did not make a comparison with metaheuristic methods to demonstrate whether the proposed rules are competitive with them. Y.-K. Lin

and Hsieh (2014) consider a problem with setup times, release times, and the total weighted tardiness objective. The authors adapt the ATCSR rule for the UPMSP and denote it as ATCSR_Rm. This rule takes machines into consideration when calculating the priority values of jobs. The results demonstrate that it performs better than other other DRs for the considered problem. The contribution of the preceding two studies lies in the adaptation of the static ATC rule variants for the UPMSP, which consider all jobs in the system when calculating priorities. The benefit of these rules is in the fact that they will achieve a better performance than the standard ATC rule when applied on static problems. However, a more thorough comparison between the proposed static ATC variants and metaheuristics would have been useful to show the gap in the performance of these methods, but also the differences in their execution times.

Problems with additional constraints

Apart from the studies which considered problems without additional constraints or consider only setup and release times, several studies focused on problems with additional constraints and the development of appropriate DRs for such problems. Liu and Yang (2011) propose a new DR for problems with precedence constraints and makespan minimisation. This DR prioritises jobs with a larger deviation of their processing times, whereas the machines are prioritised by the speed by which they process the jobs on machines. Unfortunately, the proposed DR is quite sophisticated and it is not compared to any existing DRs. This would be quite useful as it is not difficult to adapt existing DRs for precedence constraints by considering only those jobs whose predecessors all completed their execution. With this it could be analysed how this more sophisticated rule proposed for the considered problem performs in comparison to rules that are just slightly adapted to it. Ramezanian and Saidi-Mehrabad (2012) consider a scheduling problem with rework processes and release times. The information about rework processes is probabilistic, meaning that only after a job is processed it becomes known whether the job has to be reprocessed or not. Therefore, it is difficult to apply methods that search the entire solution space and the authors propose five DRs for minimising the makespan. Two types of DRs are applied, those which assign a small slot for the potential rework process immediately after each job finishes executing, and those in which the rework processes would be processed after all normal job have been scheduled. The second variant achieved a better result as it allows more flexibility because rework processes are not prescheduled, but rather assigned when they appear. However, it should be outlined that such a procedure only makes sense when considering the offline construction of the schedule, since in dynamic problems, where the schedule is constructed in parallel with the execution of the system, rework processes could be considered as standard jobs and the DR could determine whether to schedule a new job or a rework process. As such, no preliminary assignments of slots for rework processes would be required.

Ruiz and Andrés-Romano (2011) examine a scheduling problem that includes setup times and additional resources that are used when performing setups. By assigning more resources to machines, the setup times can be reduced by a certain amount. The authors optimise the sum of the total flowtime and the cost of using resources for setups. Several DRs are proposed for solving this problem, which prioritise jobs with lower processing and setup times. The authors perform an analysis and denote that it is best to schedule more resources for jobs earlier in the schedule, as well as to machines in which the addition of resources leads to a larger decrease in the duration of setup times. Based on these observations the authors propose an addition to DRs that assigns the resources to machines after a solution is constructed. However, due to this second pass such a DR would not be usable in dynamic environments. Nevertheless, the performed analysis is valuable as it is one of the rare studies which considers DRs for problems with auxiliary resource constraints and outlines an effective strategy for resource assignment. I.-L. Wang, Wang, and Chen (2012) minimise a weighted combination of the makespan and number of tardy jobs in a problem that included job release times, machine eligibility constraints, and setup times. In this problem it is considered that recipes can be attached to machines, and that the setup times between jobs of the same recipe do not exist. As such, the authors define a rule which should assign a job without setup times (i.e. no recipe changeover), and secondly prefer jobs that are going to miss their due dates. The proposed DR uses different job assignment mechanisms depending on the number of waiting jobs and available machines, and is further improved by a LS with three neighbourhood operators. The basic DR (without LS improvements) does not outperform the EDD rule in all cases. In addition, the authors compare their DR (using LS improvement) with TS and show it performs better. However, it is not clear whether the TS search uses a random initial solution or not. If TS used a random initial solution, the comparison cannot be considered fair as it had a worse start than the method which combines the DR with LS.

The problem of minimising the energy and tardiness cost is considered by Z. Li, Yang, Zhang, and Liu (2015). In this variant machines have three operating modes (operation, wait, stop) with different energy consumption rates. Therefore, in addition to the standard scheduling decisions it is also required to determine when machines should be turned off or on. The main contribution of this study is that the authors propose several new DRs that are adapted for minimising the energy consumption. These rules, in addition to scheduling jobs, also determine whether the machines should be shut down until the next job arrives, or whether they should remain idle, depending on which decision would lead to a lower energy consumption. Since green scheduling problems, which often deal with reducing the consumption of energy or greenhouse gas emissions, are gaining more attention because of environmental concerns, this study is relevant as it is the only one which deals with the design of appropriate DRs. It is very likely that, as green scheduling problems will be gaining more momentum, more research will be performed in this area.

Batch scheduling

Several studies also focus on batch scheduling problems, the first group of which focused on the serial batch scheduling problem. D.-W. Kim, Na, and Frank Chen (2003) consider a problem with setup times and total weighted tardiness minimisation. The authors adapt two existing DRs, earliest weighted due date (EWDD) and shortest weighted processing time (SWPT), for ordering the batches that are then assigned to machines. The results demonstrate that although these rules can obtain good results, they are inferior to a metaheuristic approach (in this case SA). J.-P. Arnaout and Rabadi (2005) examine four DRs for the same problem but with flowtime minimisation. Since the batch can contain only identical jobs, the selection of jobs into batches does not need to be performed, but rather only the selection of the batch and its allocation to a machines. The authors adapt several existing rules and propose a novel DR based on the rule proposed by Weng et al. (2001) with an adapted strategy for ordering the batches. J.-P.M. Arnaout, Rabadi, and Mun (2006) extend the previous research by considering stochastic execution and setup times. In this case the real processing times are not known until the job starts executing on the machine. The authors use the same methods from the previous study and the conclusions are the same as when the deterministic variant of the problem was considered. The previous two studies demonstrate that existing DRs can quite easily be adapted for the serial batch scheduling problem variant. Na, Kim, Jang, and Chen (2006) consider a serial batch scheduling problem with the goal of minimising the total weighted tardiness. In this problem, a batch is not allocated to only a single machine, but rather a set of machines that can process the jobs contained in the batch simultaneously. The authors apply standard DRs (those used by D.-W. Kim et al. (2003)), and DRs adapted for batch scheduling (which first sequence the batches, and then allocate them to corresponding machines). The results demonstrate that DRs adapted for batch scheduling significantly outperform the results obtained by standard DRs. Klemmt, Weigert, Almeder, and Mönch (2009) examine a problem with incompatible job families and release times, with the objective of minimising the total weighted tardiness. They propose a variant of the ATC rule adapted for batch scheduling. This version of the ATC rule tries to fill holes in the schedule that would occur because batches are delayed due to of the release times of jobs. Unfortunately, the authors do not compare to any of the previous batch scheduling DRs proposed for the same criterion. As can be seen from the previous descriptions, most of the rules were simply adapted in a way that they first select the batch to be scheduled and then allocate it to a machine for execution. However, since the preceding studies focused mostly on different problem types, the proposed DRs were rarely compared across studies and therefore it is difficult to outline which of the previously described rules would perform the best.

The second group of studies, which were published more recently, shift the focus to the parallel batch scheduling problem. X. Li, Huang, Tan, and Chen (2013) consider a parallel batch scheduling problem with different job

sizes and makespan minimisation. They propose two kinds of DRs for solving the considered problem. The first kind schedules jobs to batches and then allocates the batches on machines. The second group of DRs first schedules jobs to machines, and then constructs batches out of the scheduled jobs. The authors perform extensive experiments and show that the heuristic which first allocates jobs to machines achieves better results. The study provides a good investigation in potential DRs for parallel batch scheduling, especially considering different variants that can be used. Arroyo and Leung (2017b) consider a parallel batch scheduling problem with the objective of minimising the makespan, which includes job release times and different job sizes. The authors propose three DRs for the considered problem, where the first two are similar to the ones proposed by X. Li et al. (2013). The third DR selects whether it is better to create a new batch, or add the current job to an already existing batch assigned to the machine. Thus, in this rule, both decisions (allocation of jobs to batches and machines) are performed simultaneously, meaning that if the job does not fit in an existing batch a new batch is created on the machine and it is immediately allocated to it. The authors show that simple DRs can be used to tackle the considered problem efficiently, and that the proposed DRs achieves the best results. Out of the two remaining rules, better results are achieved by the rule that first allocated jobs to machines and then groups them into batches. Therefore, the study demonstrated that it is more beneficial to perform both decisions (grouping jobs into batches and allocating them to machines) simultaneously, since the rule has a better overview of the problem. Arroyo and Leung (2017a) consider the same problem but with different machine capacities, and they adapt the best DR from Arroyo and Leung (2017b) for that problem. However, the rule is compared only to metaheuristic methods against which it performs quite poorly. Zarook, Yaser, Rezaeian, Javad, Mahdavi, Iraj, and Yaghini, Masoud (2021) investigate a batch scheduling problem with release times, unequal job sizes, and makespan minimisation. Several simple DRs used for creating batches and scheduling them on machines are proposed. These DRs work in two ways, the first group constructs the batches and then allocates them to the machines, whereas the second group first allocates jobs to machines and then groups them into batches, which makes them both similar to DRs proposed by X. Li et al. (2013). Neither DR variant is always superior, however, the results seem to suggest that better results are usually obtained by the rules that first allocate jobs to machines and then group them into batches, which is consistent with the previous observations of X. Li et al. (2013). The authors also propose a genetic algorithm, however, the results are odd from the point that the proposed GA achieved a lower execution time than DRs and worse results than most of DRs. As such, the comparison might not be quite fair as the GA was not given the same amount of time and it is unknown why it achieved worse results, especially as the test instances were not too large. Furthermore, the execution times of the DRs were quite high, reaching up to 1500 seconds, which could potentially provide a limitation in dynamic environments. Although not a

great deal of studies were performed on the parallel batch scheduling problem, the contributions are nevertheless significant as the studies examined several ways of defining DRs for the considered problem, and the conclusions about which is the best for this problem are consistent throughout all the studies. This represents valuable knowledge for other researchers that will tackle this problem in order to be aware which type of DR to use for solving the problem.

Performance over multiple critera

Due to the large number of DRs that were proposed during the years for various objectives, it became difficult to determine which rules are the most appropriate for which criteria and scheduling conditions. However, only two studies actually provided a comparison of different DRs across several objectives and scheduling conditions. Xhafa, Barolli, and Durresi (2007) perform a comparison between 5 DRs and evaluate their performance on four criteria: makespan, flowtime, machine utilisation, and matching proximity (defines how many jobs are scheduled on machines which execute them the fastest). The rules are tested under different job and machine heterogeneity conditions. The results show that there is no single rule that performs well over all criteria, however, the min-min DR achieved the best results for optimising the makespan and flowtime. In addition, in their tests the authors obtain that the sufferage method does not perform as well as some other methods like it was demonstrated by Maheswaran et al. (1999). Đurasevic and Jakobovic (2018) provide a more detailed analysis by considering 26 DRs (24 from the literature, and two novel ones), and evaluate their performance on a problem with job release times and for optimising 9 criteria. Four data sets with different job and machine heterogeneity properties were used for testing, as was done previously (Maheswaran et al., 1999). This study provided a detailed overview on the performance of all DRs, outlining which DRs are the most suitable for which criteria given certain scheduling conditions, as well as which rules perform the best across all the tested criteria. The conclusions obtained in it are similar in certain points to that performed by Xhafa et al. (2007), with the main difference being that in the results obtained for the sufferage rule are more in line those obtained by Maheswaran et al. (1999), which might be due to the fact that Xhafa et al. (2007) did not consider release times. The obtained results show that there is no single DR which is always dominant, but they still show that certain DRs are constantly performing poorly (and as such they do not represent a viable choice). However, for each criterion there is usually a set of rules that always achieve good performance regardless of the scheduling conditions. Although this study provided a great overview of different DRs, it still did not manage to cover all of them from the literature and also focused only on problems with release times, which still leaves room for performing similar analyses for other variants as well.

Conclusion

As can be seen from the review, designing DRs is a topic which is constantly active throughout the years, even though in recent years the research in this area has been slightly declining. Although initial studies focused mostly on the problem variant in which no constraints were considered, recent studies considered the design of DRs for more complex variants which include several constraints like batch scheduling or additional resources. However, there is still a limited number of DRs applicable for problems with multiple constraints, which is a consequence that it is difficult to design good DRs when there are several constraints that need to be considered. Therefore, there is a lot of opportunity in this area either in defining novel DRs or proposing procedures for adapting existing ones for additional constraints. The later variant would be especially interesting, since procedures for adapting existing DRs could then be used to adapt a large number of rules for different constraints. Furthermore, the performance of DRs is still limited, and as such there is still a lot of room for designing improved DRs.

3.1.2 Automatically generated DRs

Aside from manually designed DRs, the application of GP and similar methods for automatic development of DRs has become increasingly popular over the last several years (Branke et al., 2016; Nguyen et al., 2017). The main reason for this is due to the fact that it is difficult to manually design good DRs for various scheduling problems, especially for the more complex ones which consider additional constraints or multiple objectives. ADDRs have been quite a popular topic for the job shop environment, however, they have been significantly less researched in the context of the UPMSP. This section provides an overview of existing research dealing with the automated design of DRs.

Z. Zhang, Zheng, and Weng (2006) propose the application of reinforcement learning for solving scheduling problems, in which the scheduling problem is modelled as semi-Markov decision process. Five DRs are used as actions during the learning phase and the reward function is based on the minimisation of tardiness. A Q-learning algorithm is applied to solve the defined reinforcement learning problem and its performance is compared to individual DRs. In all cases Q-learning significantly outperformed all the tested DRs and demonstrated that it can select good actions at various decision points. This study is interesting from the point that it tries to model the scheduling problem as a reinforcement learning problem and actually learn an algorithm which would perform the best decisions at each point in time. As such, this work can be seen as a predecessor to subsequent studies that focus more directly on generating novel DRs. However, the main difference between this and later studies focusing on automated design of DRs is in the fact that the knowledge obtained by Q-learning is not interpretable, i.e. it is not possible to actually explain the strategy by which this method performs scheduling decisions. On the other hand, subsequent studies in this area use GP and similar methods to generate an expression for performing scheduling decisions, which can be

interpreted by human experts in order to get a notion on the inner workings of such automatically designed DRs.

All the following research on automated design of DRs was performed on problems which included release times. The reason for this was that almost all studies focused on dynamic problems in which jobs are released over time and no information about them is known before they are released. The first application of GP to generate DRs for the UPMSP was considered by Đurasevic et al. (2016). GP evolves a priority function used to rank jobs and machines when creating the schedule. The authors considered four scheduling objectives, makespan, total flowtime, total weighted tardiness, and number of tardy jobs. The evolved DRs are compared with manually designed DRs and demonstrate better performance. This study laid out the foundations for all subsequent studies in this area, since it proposed which properties should be used when constructing the expressions of the DRs as well as how these expressions should be used to construct the schedule. Subsequent studies focus on different directions either in improving the performance of the methods used to design DRs, or to design these rules for other problem variants.

Đurasević and Jakobović (2017b) considered automatically designing DRs for MO problems, where 9 scheduling criteria were considered in various combinations. The authors applied 4 MO algorithms: NSGA-II, NSGA-III, MOEA/D, HaD-MOEA. The results show that automatically developed DRs achieved much better results than manually designed DRs when applied for solving problems in which several criteria are optimised simultaneously. This result is not surprising as manual DRs were usually designed for optimising only a single criterion. However, often it is required to optimise several objectives simultaneously. Since manually designing such rules is even more difficult, designing them automatically via GP represents a viable alternative. Đurasević and Jakobović (2017a) propose the application of ensemble learning methods from machine learning for automatically designed DRs. The motivation for this research is that a single DR cannot perform well on all problem instances, but combining them into ensembles which jointly perform scheduling decisions could further improve their performance. Therefore, ensemble learning methods were adapted for this problem to create sets of DRs that perform their decisions jointly. Four ensemble learning methods were tested: simple ensemble combination (SEC), BagGP, BoostGP, and cooperative coevolution. The obtained ensembles show a better performance in comparison with a single manually or automatically designed DRs. Because the SEC method constructed the best ensembles, Đurasević and Jakobović (2019) investigate the SEC method further in different scenarios to determine how it performs with different rule sets and ensemble construction methods. The main contribution of these two studies is that they demonstrate that by using DRs in synergy their results can be further improved. This is an important finding as it shows that it can be easier to achieve better results by using several rules simultaneously than to perform complex extensions on the GP to generate better individual DRs. It would also be interesting to

see whether such a methodology could be applied to a certain degree also on MDDRs, and if their results could also be improved to a certain extent.

Đurasevic and Jakobovic (2020) examine different strategies which can be used to schedule jobs on machines in ADDRs when minimising the total weighted tardiness. This includes the analysis of allowing idle times or not, as well as whether a single priority function should be used to determine the sequence and allocation of jobs to machines, or whether this decision should be split into two priority functions. The main contribution of this study is in demonstrating why ADDRs actually perform better than MDDRs. In most MDDRs the selected job is allocated to the machine on which it would complete the soonest. However, when such a strategy is used with ADDRs, it can be noticed that the generated rules do not perform equally well as those generated when the rule performs both the scheduling and sequencing decisions. This suggests that the strategy of allocating jobs to machines might be the limiting factor of existing MDDRs, and that it might be beneficial to research alternative strategies for that decision. Previous studies on ADDRs focused on dynamic scheduling problems in which the decisions had to be performed on line during the execution of the system. However, Đurasević and Jakobović (2020) focused on problems in which the schedule could be constructed prior to system execution, and in which the total weighted tardiness criterion was optimised. Four methods for improving the performance of ADDRs when considering static scheduling conditions were proposed. The authors show that in these cases ADDRs can match the performance of a GA, or achieve a better solution in a smaller amount of time. This shows that given all the information about the problem, it is possible to develop DRs that can even match the results of more sophisticated metaheuristic algorithms. Jaklinovic, Đurasevic, and Jakobovic (2021) considered the minimisation of the same criterion with additional constraints like release times, setup times, machine eligibility, precedence constraints, and machine availability periods in various combinations. The ATC rule and a GA were adapted for solving all the combinations of these constraints. The results demonstrate that ADDRs achieved a better performance than MDDRs for most constraint combinations, however, they could not match the performance of the GA. This study demonstrated that it is possible to automatically design efficient DRs for more complex scheduling problems that include many additional constraints, which is important if such rules would be applied for real world problems with several constraints that need to be considered. The previous study can be extended further by considering other more complicated constraints like auxiliary resources or batch scheduling.

The next several studies are more focused on the algorithms used in the automated design of DRs, rather than the considered scheduling problem. Planinić, Backović, Đurasević, and Jakobović (2022) compare different GP representations for the automated design of DRs. The authors considered four scheduling objectives, makespan, total flowtime, total weighted tardiness, and number of tardy jobs. They show that all tested methods achieve a similar

performance, with GP and GEP usually generating the best DRs. However, they also demonstrate that different methods often create DRs that are quite complicated for interpretation, which makes it difficult to understand how the DRs actually perform their decisions. To alleviate this problem, Planinić, Đurasević, and Jakobović (2021) apply simplification methods to reduce the sizes of the evolved DRs and make them more understandable. This study demonstrated that the size of the rules can be significantly improved by these reduction methods without an impact on the performance of the DRs. This implies that such simplification methods have potential to partially fix the problem of GP in designing overly complicated DRs, and can be used to reduce the rules to only the most important elements to easily extract knowledge from them. Planinić, Đurasević, and Jakobović (2021) investigate different parent selection mechanism for designing DRs, including the lexicase selection. However, this study is focused more on the algorithmic perspective, rather than on the UPMSP. Đurasević and Jakobović (2022) outline that a significant problem with ADDRs, but also MDDRs, is selecting the appropriate rule for the problem instance under consideration. The most appropriate DR depends on many different system properties, as shown in previous studies, and it is not possible to know up front which rule is the most suitable for solving the considered problem instance. The goal of the study is to use different machine learning methods to learn which DRs perform well on instances with known or estimated characteristics, and which can then be used to suggest an appropriate DR for the considered instance. The results demonstrate that such a method has potential to improve the performance over randomly selecting a single DR, since during the execution of the system the method tries to select the most appropriate rule given the current system state. However, the study considered only a few system parameters, mostly those which describe the entire problem rather than the current status of the system. This research direction was already widely investigated for other machine environments (Priore, Gómez, Pino, & Rosillo, 2014), however, not for the UPMSP. As such, this study can potentially open a new research direction in the automatic selection of DRs most appropriate for the considered problem instance.

From the previous overview, it can be seen that the majority of research in this area has been performed in the last several years. Therefore, this area is still new and a lot of directions are still open. For example, the ADDRs are not as interpretable as MDDRs, which makes it difficult to understand the logic behind their choices. Since ADDRs can achieve better performance than MDDRs, this motivates further research. As most studies were focused on a problem variant which included only release times, there is still a need to research the applicability of this approach for more complex problem variants, especially those appearing in real world scenarios.

3.2 General heuristics for UPMSP

This section provides an overview of general heuristics for the UPMSP. These heuristics are more complicated than DRs and they usually use more

information or more complex strategies for constructing the solution, or even construct it in several iterations. Quite often these heuristics even represent combinations with other methods like mathematical programming. However, since these heuristics include more domain knowledge and more sophisticated procedures, they usually perform better than DRs and in some cases match the performance of metaheuristics.

Suresh and Chaudhuri (1994) propose a heuristic for minimising the maximum tardiness. The authors apply an algorithm to resequence all the jobs on the machines in the increasing order of their due dates. Additionally, improvement procedures (exchanging jobs) on the final solution are also applied to improve its performance, which leads to better results. Unfortunately, the authors do not provide any comparisons with other metaheuristics or DRs to asses the performance of the proposed method. SURESH and GHAUDHURI (1996) consider a scheduling problem with machine availability periods that can either be deterministic or stochastic. A heuristic method is proposed, which takes into account the possible occurrence of machine availability periods during scheduling. If the heuristic is applied in a probabilistic scenario it performs rescheduling procedures whenever it detects that a machine will not be available. This study is important as it tries to tackle an important constraint in scheduling, namely that machines will not always be available or can break down. However, the authors did simplify the model in a way that machine availability periods can occur only after a job has finished executing. Nevertheless, with little modifications the proposed method should also be applicable in the general case.

Herrmann, Proth, and Sauer (1997) examine a scheduling problem with precedence constraints and makespan minimisation. The authors propose several heuristic methods which prioritise jobs that could delay future tasks. The procedures first assign jobs to machines and then sequence them to satisfy the considered constraints. In addition, SA is coupled with these heuristics to further improve the results. The obtained results show that the proposed heuristics obtain good, even optimal results in some cases. However, the authors examined problems in which only a small number of precedence constraints existed (less than the number of jobs), and performed no comparison with alternative methods. As such, it is not known how this method would perform on more complicated problems or how it performs in comparison with metaheuristic or similar methods.

Randhawa and Kuo (1997) devise a heuristic method to optimise three criteria (flowtime, total tardiness, and number of tardy jobs) considering several constraints like machine eligibility, setup times, and product types. The proposed heuristic consist of several steps, where jobs are first grouped into tasks by product types to decrease setup times. Those tasks are then assigned to machines and individually sequenced on them. In the sequencing step of jobs several simple DRs rules are used, and the best for each optimised criterion is determined. Finally, the authors provide a detailed analysis on the effects of different system parameters on the results. This study is the only one in which

a heuristic for a MO problem is proposed, in a way that they are not linearly combined, however, the heuristic still provides a single solution and not a set of solutions as MO metaheuristic methods. Dhaenens-Flipo (2001) also consider a MO problem with setup times, however, they linearly combine the makespan and cost incurred by setups and job operations. The authors define a heuristic procedure to solve the given problem, two methods for constructing initial solutions, and introduce an improvement phase to further enhance solutions. However, the authors provide only a quite brief experimental analysis of the proposed method on one larger problem instance, with no comparisons to other methods. Therefore, it is difficult to judge on the general performance of the proposed heuristic.

Bank and Werner (2001) consider the minimisation of the total weighted tardiness and earliness penalties with job releases and a common due date. The authors propose several constructive heuristics and iterative algorithms (SA, TA, iterative improvement, and multi-start heuristic) for solving the considered problem. The experiments conclude that the performance of constructive heuristics depends heavily on the characteristics of the problem, whereas among the metaheuristics, TA usually achieved the best results. Based on these results it seems that metaheuristics might be a better choice for the considered scheduling problem, rather then developing a new heuristic method. Al-Salem and Armacost (2002) proposed a heuristic for minimising the makespan with machine eligibility constraints. The heuristic first constructs the solution in two steps. In the first step it assigns jobs to eligible machines, and then in the second step it sequences them based on their processing times. After the construction of the schedule, the improvement heuristics from Hariri and Potts (1991) is used and adapted for considering machine eligibility constraints. In the experimental study the authors show that the proposed method achieves solutions that are slightly worse than the optimal ones, however, no comparison is performed with other methods to demonstrate how difficult the problem instances really are and if other algorithms could also achieve a similar performance.

Several studies also proposed heuristics for the serial batch scheduling problem. However, all of them focused on solving different scheduling variants, and as such the methods and results are not comparable between the different studies. D.-W. Kim et al. (2003) consider a problem with setup times and total weighted tardiness minimisation. The authors adapt two existing DRs, EWDD and SWPT, for this problem, and also propose a SA and a problem specific heuristic. The heuristic method first orders the batches using EWDD, and then allocates the batches to machines and fills them with jobs to minimise the weighted tardiness. The experimental results demonstrate that the DR methods achieved the worst results, and that SA significantly outperformed all other methods, which shows that metaheuristics perform better than heuristics specifically designed for the considered problem. This observation is important from the point that it denotes it might be more beneficial to apply metaheuristics than design novel problem specific heuristics for the

problem. Silva and Magalhaes (2006) examine a problem from the textile industry and model it as an UPMSP in which the number of tardy jobs criterion needs to be optimised. The characteristics of this problems are that jobs cannot be processed on all machines and setup times occur when two lots are interchanged. The authors define a specialised heuristic which first selects the job, then the batch into which it will be included, and finally the position in the batch. The authors perform an interesting case study of the proposed algorithm by applying it on real problem instances and comparing it to schedules obtained by planners that were executed in production. The experimental analysis shows that the proposed heuristic constructed better schedules than those that were actually used. As such, this provides motivation to further investigate real world problems and apply such methods on them as they can lead to significant improvement of the production process.

Dolgui, Eremeev, Kovalyov, and Kuznetsov (2009) examine the problem of scheduling lots with setup times and machine eligibility. In this case, lots represent a serial batch of jobs between which no setup times are invoked. The authors propose a heuristic, based on the nearest neighbour strategy from the travelling salesman problem, which iteratively schedules the jobs to machines. The method achieved good results compared to a GA and an exact method for random instances, but on real world instances the method was significantly inferior to both methods. Therefore, it is questionable how general the proposed heuristic is, and if it only works well for a certain problem type. X. Xu, Ma, Zhou, and Zhao (2015) consider a batch scheduling problem in which jobs can be split across different machines to improve the performance of the system. The jobs that are being processed require an additional quantity of certain resources that are available. The authors propose three heuristic algorithms based on several optimality properties, which reduce the original problem to several sub problems that are iteratively solved. The authors provide a detailed analysis of the proposed heuristics on different scenarios. Although the authors compare the obtained results to a lower bound, they do not provide any comparison with other heuristic or metaheuristic methods to demonstrate the relative performance of the proposed heuristics to other methods.

J.-F. Chen (2004) consider a scheduling problem with setup times and additional resources with the objective of minimising the makespan. The auxiliary resources are limited and a job cannot be scheduled if not enough resources are available. Resources can freely be attached to machines, but this process requires a certain setup time. The authors propose a heuristic method which is based on assigning jobs to machines with the smallest processing times, scheduling jobs that require the same resource one after another, and keeping the load balanced across all machines. The authors compare the proposed heuristic with the SA algorithm proposed by Tamaki, Hasegawa, Kozasa, and Araki (1993), through which it was demonstrated that the proposed heuristic achieved better results and required less computational effort. This demonstrates that a well designed but complex heuristic, can

outperform a general metaheuristic method. However, this also raises an important question in balancing between complex and very specific methods, which might not perform well for other problem types, and more general methods that are less complex but might not be able to perform equally well. This issue will also become more apparent when considering metaheuristics.

Y. Lin, Pfund, and Fowler (2011) propose several heuristics for the individual optimisation of three objectives, makespan, total weighted flowtime, and total weighted tardiness. The first heuristic uses LP to construct the schedule and improves it with several neighbourhood procedures. The other heuristics use a DR to construct the initial solution, and improve the solution quality with LS. All the proposed heuristics are compared with a simple GA without any additional local search operators or additional information about the problem. The results demonstrate that the GA outperformed all of the other methods for the three tested criteria, which suggests that metaheuristics provide a better alternative than designing specific heuristics for certain problems.

Polyakovskiy and M'Hallah (2014) developed a multi-agent system for the optimisation of the total weighted earliness and tardiness criterion. The proposed system consists of three agent types with different fitness functions and roles that they need to achieve. Each agent uses different procedures (approximate and LS methods) to solve its own problem. The authors perform an extensive examination of the method and compare it to a state of the art heuristic method which, however, was developed for the identical parallel machine problem. As such, there might be a possibility that the method to which the authors compare under performs for the UPMSP. Nevertheless, the idea that the authors present is novel, and shows that the scheduling algorithm can be successfully tacked with methods from different areas. A problem from Polyvinyl Chloride pipes is formulated as a scheduling problem by C.-H. Lee, Liao, and Chao (2014) and includes dedicated machines, setup times, and a common deadline for all machines. The objective is to minimise the total completion time of all jobs. The authors propose 3 heuristic procedures for assigning jobs to machines that take into account which machines are dedicated for processing which job. The results demonstrate that the proposed heuristics outperforms a baseline method by a significant margin, however, neither of the proposed heuristics was dominant across all problem variants. This paper presented an interesting study of a real world problem in which the authors show that the production can be improved by considering more specialised heuristic methods.

Fanjul-Peyro, Perea, and Ruiz (2017) study a scheduling problem with renewable resources that are required for processing jobs and need to be assigned to machines. The authors propose the application of matheuristics for minimising the makespan, which represent a combination of mathematical programming and heuristics. In those heuristics a mathematical model is solved, with the addition of some decisions being performed heuristically, like reducing the jobs or machines that are considered in the problem which is being

solved. The proposed methods perform better than regular exact methods for problems until a certain size, however, for larger instances the method struggles with obtaining good results. The method is somewhat limited as it requires a MIP definition of the problem, which can be quite difficult for the more complicated problem variants. Finally, the execution times of the method are quite large, which makes the entire method quite time consuming. Villa, Vallada, and Fanjul-Peyro (2018) propose several heuristic methods for solving the same problem. The first group of heuristics execute in three phases. In the first phase they order the jobs (based on different strategies), then construct the entire solution, and finally improve it by using various LS operators. The second group of heuristics does not consider resources at all, but rather constructs the mapping of jobs to machines. This is likely to produce infeasible schedules, and therefore a correction procedure is applied to fix the solution. The results demonstrate that the proposed heuristics improve the results of the methods from Fanjul-Peyro et al. (2017), and that the second group of heuristics achieve better results. This study nicely fills the gap left by the study of Fanjul-Peyro et al. (2017), since it shows that heuristic methods are still a better choice for solving the considered problem since they always provide a better efficiency, and are more effective for large and medium sized instances.

Che, Zhang, and Wu (2017) examine the energy conscious unrelated machines environment in which the goal is to minimise the total electricity consumption. In this problem each job is additionally characterised by the electrical energy that is consumed when it is being executed on certain machines. The electricity prices are considered to change during the day, because of which the time horizon is divided into several periods, each with its own electricity price. The authors propose a two step heuristic in which the jobs are assigned to machines to minimise the total cost (with the property of being preemptive) and then in the second stage the jobs are scheduled without preemption using an insertion heuristic. The proposed heuristic achieved a good performance in comparison with an exact model, but was not compared to any other heuristic or metaheuristic methods to demonstrate its effectiveness. Nevertheless, this study introduces an interesting model for the green scheduling problem, which is different from the models in other studies.

Perez-Gonzalez, Fernandez-Viagas, Zamora García, and Framinan (2019) consider a problem with setup times, machine eligibility constraints and total tardiness minimisation, for which they adapt several heuristic methods. These heuristics are based on ordering jobs by certain properties and iteratively inserting them in the schedule at the position which leads to the smallest value of the optimised criterion. The authors test their heuristic against those proposed by Al-Salem and Armacost (2002) and Rabadi, Moraga, and Al-Salem (2006), as well as against CLONALG proposed by R.O. Diana, Filho, Souza, and Silva (2013). The results show that the proposed heuristics perform better than existing ones, but they cannot match the performance of CLONALG which achieved the overall best results. However, the heuristics construct schedules much faster and provide trade-off between performance

and execution time. In the end this study again proves that metaheuristics can outperform problem specific scheduling heuristics.

Based on the previous overview of the research we can see that that such general heuristics did not receive a wide attention as DRs or metaheuristic methods. The reason for this is that usually much more effort is required in designing problem specific heuristics, than simply applying general metaheuristic methods of modifying them for the UPMSP. However, research performed in this area is nevertheless quite valuable as most of the procedures and strategies can easily be used in combination with metaheuristic methods to improve performance. Although certain heuristic methods will still be proposed, it is quite probable that these methods will not be gaining any wider attention, and that most research will be directed towards applying metaheuristic algorithms.

3.3 Metaheuristics

This section will provide an overview of metaheuristic methods applied for solving the UPMSP, which will show that over the years metaheuristic methods have become the most popular methods for solving the considered problem. The research will be grouped into several major groups depending on the considered problem or criteria. For example, studies dealing with makespan minimisation and setup times will be grouped in one section, whereas other sections will deal with studies that examine problems that consider additional resources or batch scheduling. The categorisation was done completely provisionally with the aim to group as much interrelated research into the same section as possible. It is possible that a certain study would belong to two or more categories, however, it will only be mentioned in one which is deemed more relevant for the study. Furthermore, the research in each section will not be described in a completely chronological order, rather similar studies will be grouped together to outline similarities and differences between them where possible.

3.3.1 Problems with makespan minimisation and no constraints

The problem of minimising makespan without any additional constraints is one of the first and most commonly investigated variants of the UPMSP. Regardless of that it does not include any additional constraints, this problem is still NP-hard, and as such received wide attention. Due to its simplicity, it was considered mostly in the initial studies dealing with the UPMSP.

Hariri and Potts (1991) use an ID algorithm which applies two simple neighbourhood operators on initial solutions constructed with DRs. The neighbourhood operators they use are insertion (inserting a job from one place to another) and interchange (exchanging two selected jobs), which are applied on the starting solution until convergence. Essentially, this makes the entire procedure quite greedy and susceptible to getting stuck in local optima because

the authors do not apply any operators to introduce random changes in the solution. Nevertheless, this study can be considered pivotal as it is the first to apply some kind of a metaheuristic, although basic, for the considered problem. Glass, Potts, and Shade (1994) outlined the possibility of getting stuck in a local optimum when using the method proposed by Hariri and Potts (1991), and in addition to ID, they use several metaheuristic methods like SA, TS, and GA, which introduce randomness in the search process. The authors apply the same neighbourhood operators and demonstrate that the algorithms which include stochastic behaviour achieve better results. This study was one of the first in which metaheuristics were applied for the UPMSP, and as such it is relevant due to the fact that it demonstrated the applicability of such methods for the considered problem. However, Piersma and van Dijk (1996) criticised the use of such methods as they were quite general and did not exploit the structure of the underlying problem. They proposed a new ID algorithm which performs better than standard metaheuristics, but also combine it with TS to further improve the results. As such, this paper can be considered as one of the first in the direction of introducing more domain knowledge and special search structures into general metaheuristic methods. This research direction will become even more popular in later studies where different metaheuristic algorithms will be coupled with various LS and problem specific methods to improve their performance. GUO, LIM, RODRIGUES, and YANG (2007) adapt the neighbourhood structure proposed by Piersma and van Dijk (1996) and use it in combination with SA, TS, and SWO. The results obtained by the authors further show the importance of combining metaheuristics with LS methods to obtain better performance. Although the SWO metaheuristic achieved the best results, it is a rather exotic algorithm that was not used in any further studies when dealing with UPMSPs.

Srivastava (1998) proposes a modified TS which uses hashing to keep track of the visited solutions and improve the performance of the algorithm. The results demonstrate that the proposed TS algorithm performs better than the canonical variant. However, the improvement of the algorithm is rather general and not tied directly to the UPMSP. T. Braun et al. (1999) and T.D. Braun et al. (2001) apply the GA, TS, SA, and GSA metaheuristics and compare their results with those of simple DRs. The authors also consider constructing initial solutions for metaheuristics by using DRs. However, they only apply this for GA by introducing a single solution generated by DRs in the population, whereas the others are randomly generated. The results demonstrate that GA achieved the best results among all the metaheuristics, and that other metaheuristics sometimes even performed worse than DRs. However, the comparison can be considered slightly unfair because the GA used a better initial solution that ensured that it would never perform worse than the best DR, which was not the case with the other considered metaheuristics. Furthermore, the other metaheuristics used quite simple neighbourhood operators, which also limited their performance. Therefore, with a better

design of metaheuristics their results could have been improved, which is the case in other studies since metaheuristics almost always outperform DRs.

Ritchie and Levine (2003) use a simple LS method on solutions obtained by DRs, and show that such a procedure performs better than a standard GA. These results are consistent with previous studies (Glass et al., 1994), which demonstrate that simple methods based on LS operators can perform better than more complicated metaheuristics. Such results are interesting as they show the limitation of standard metaheuristic methods as well as that simple LS based methods with a good initial solution can achieve competitive or even better results. Gao (2005) applied a standard GA parallelised using MPI. However, the contribution of this study is minor in the UPMSP as it mostly focuses on algorithm parallelisation, and not that much on the problem or adaptation of the algorithm to the problem. Charalambous, Fleszar, and Hindi (2010) use two additional auxiliary objectives (related to the makespan and processing of jobs) to improve the performance when optimising the makespan. They apply a VND method coupled with a MILP model in the large neighbourhood search to improve its performance. From the results it is difficult to asses whether the formulation with auxiliary objectives had a positive effect and in which extent the MILP model in the VND improved the performance, since a deeper analysis was not provided. Since defining a MILP model for the problem is not always trivial, this represents a limiting factor on the proposed approach. Briceño et al. (2012) apply a GA to solve the problem where in the addition of optimising the makespan, the goal is also to reduce the makespan on all non makespan machines to distribute the load across all machines. For that purpose the authors use the GA in an iterative way, in which they first solve the full problem, and after finding the solution they remove the makespan machine with all jobs allocated to it, and solve the reduced problem again. However, they also introduce the best solution from the previous iteration in the initial population of the next iteration to ensure that the GA always achieves at least as good results as in previous iterations. In the experiments the GA has shown to be a better choice for such a problem in comparison with DRs, which could not guarantee that the makespan will be decreased for the reduced problem.

Balin (2011) proposes a novel matrix encoding and uses it in combination with a GA. However, the algorithm is evaluated on a very simple problem instance and compared only to the LPT DR, which does not provide a good insight of the performance of the proposed solution representation. However, a different study (Vlašic, Đurasevic, & Jakobovic, 2019) showed that although this representation does perform well, it cannot match the performance of other permutation based encodings. Fanjul-Peyro and Ruiz (2010) examine a VND method with several novel neighbourhood operators and structures, which obtained better results than a commercial solver and other state of the art methods. The authors show that with a simple algorithm and well defined neighbourhood structures it is possible to match the performance of more complex and computationally expensive solvers. This result is quite important

as it demonstrates the capability of simple LS based methods to match or improve the performance of much more powerful and complex methods, which is something that will be demonstrated in studies dealing with other problem types. Fanjul-Peyro and Ruiz (2011) extend the previous method by combining it with a commercial solver to obtain initial solutions for VND. The commercial solver is used to solve a reduced problem in order to improve its execution time. The results outline that using a good initial solution is vital for the performance of metaheuristics. However, it is not investigated to which extent the initial solution has an influence on the performance of VND, since in the original study only a simple solution initialisation procedure was used. Therefore, it would have been interesting to see how the performance of VND would have been affected by using other methods, like DRs, to generate initial solutions. Fanjul-Peyro and Ruiz (2012) examine two problem variants that are called not all jobs (NAJ) and not all machines (NAM), in which it is not required to schedule all jobs or to use all the machines. The previously described VND method was used with a preliminary phase in which the jobs and machines which should be considered for scheduling are selected based on a ranking that denotes their importance in the minimisation of the makespan. Both examined problem variants are interesting, especially the NAJ which resembles the order acceptance and scheduling problem (Slotnick, 2011). Unfortunately, neither of the two variants were further examined in later studies, although there would be many research directions that could be further investigated for those two problem variants, like a MO problem variant. Sels, Coelho, Manuel Dias, and Vanhoucke (2015) define a hybrid method that combines TS and GA with a LS and a truncated B&B method to achieve near optimal results. The proposed methods performs better than the VND of Fanjul-Peyro and Ruiz (2010), however, it does introduce additional complexity and was not compared to the method proposed by Fanjul-Peyro and Ruiz (2011) to asses whether it can also match the results obtained by the improved VND method.

After the previous studies the problem was not considered for a while until it was revisited by Lei, Yuan, Cai, and Bai (2020). The authors consider a distributed UPMSP, in which machines are grouped in factories, and therefore a job first needs to be assigned to a factory and then to a machine in the corresponding factory. However, as no additional constraints are introduced, the problem is completely equivalent to the problem in which all machines would be located in the same factory. The authors apply a modified ICA with an additional memory structure. A major drawback of this study is that no comparison to any of the previously outlined methods is made, and the authors use the PSO method adapted from Salehi Mir and Rezaeian (2016) for comparison. Although the proposed method achieves a better performance than the PSO algorithm used for the comparison, it is not known how it would compare to the other methods more specifically designed for the considered problem. However, the authors also provide an interesting real world example of scheduling jobs in the Polyvinyl Chloride pipe production, which they use showcase their method. Orts, Ortega, Puertas, García, and Garzón

(2020) apply a GA for tackling a real world problem in which they use the algorithm to distribute the set of tasks over heterogeneous processing units for computing the active microrheology simulation. The authors demonstrated that their proposed algorithm achieved a good load balance of tasks over the cluster. Although they perform no comparison with previous algorithms, the motivation and application for a complex real world problem represents by itself an interesting contribution.

From the overview it can be seen that the problem of minimising the makespan without additional constraints was mostly tackled at the starting years when metaheuristics were just starting to be applied for the UPMSP, although it was recently revisited by considering certain real world problems. However, the number of studies for this problem declined in the last years, with most research moving to an extended problem which additionally includes setup times. Nevertheless, the results obtained in these studies were pivotal for the entire UPMSP, as they provided motivation for a larger adaptation of metaheuristic methods for solving various types of UPMSPs.

3.3.2 Minimisation of other objectives without additional constraints

Aside from the makespan criterion, several other criteria were optimised without considering any additional constraints.

Optimisation of flowtime related objectives

The minimisation of the total weighted flowtime was studied in the following studies. Vredeveld and Hurkens (2002) consider different approximation methods, however, they also apply TS and an IG method. The authors show that best results are obtained when using the approximation method to obtain initial solutions and then improve them with metaheuristics. This again demonstrates that metaheuristic methods can greatly benefit from using a good starting solution, rather than invoking them with a random solution, which is similar to what other studies also demonstrated (Fanjul-Peyro & Ruiz, 2011). However, the exact methods are quite time consuming, especially as the instances become larger, which is a limiting factor of the proposed method. Cruz-Chavez, Juarez-Perez, Avila-Melgar, and Martinez-Oropeza (2009) apply SA that uses simple LS operators and compare it to the results obtained by an exact method. The authors show that the method can achieve optimal performance, however, the investigation was performed only for a few instances of smaller sizes that could be solved exactly. As such the general performance of the proposed method cannot be assessed. Rodriguez, García-Martínez, Blum, and Lozano (2012) apply ABC enhanced with a LS mechanism and IG method that destructs and reconstructs the solution. They compare their approach with previous methods proposed by Vredeveld and Hurkens (2002) and Y. Lin et al. (2011), and show that the proposed method significantly improves the results.

Rodriguez, Blum, García-Martínez, and Lozano (2012) apply GRASP with standard LS operators that insert or swap jobs between machines, and combine it with PR. A special step called evolutionary PR, which performs the PR procedure on all solutions in the elite set, is also introduced into the algorithm. The authors compare their proposed method with the same methods as in their previous study to show that the proposed GRASP metaheuristics performs best. Unfortunately, the authors do not use their previously developed ABC algorithm for comparison and since the problem instances used in both papers are not the same, the results cannot be compared between the two methods to asses which is better. Rodriguez, Lozano, Blum, and García-Martínez (2013) use IG with 8 construction schemes, two strategies for solution destruction, and also propose three improvement strategies. They evaluate different choices and compare the algorithm to the state of the art methods (Rodriguez, Blum, et al., 2012) to demonstrate its superiority. This study outlines two important conclusions, which are that randomness in the acceptance criterion is useful (which might suggest why SA usually performed well), and that the strategy for destructing solutions impacts the effectiveness of the algorithm.

Y.-C. Chang, Li, and Chiang (2014) use ACO to solve a combined problem of production and distribution, in which the production part is represented as an UPMSP. The algorithm first constructs the schedule, which is done by traversing the graph obtained by enumerating all mapping possibilities between jobs and machines. After that, the algorithm solves the distribution part of the problem. The contribution of this paper is that it focuses on a difficult problem which combines two sub-problems that are often considered separately. The algorithm is shown to achieve near optimal solutions, however, the authors did not use any algorithms that consider both problems separately to demonstrate the gap between the two approaches. H. Wang and Alidaee (2019) use an enhanced version of TS and adapt the k-opt strategy commonly used for the travelling salesman and vehicle routing problems. Additionally, the authors implement an IG algorithm of Rodriguez, García-Martínez, et al. (2012) and test several options for each of the algorithm phases. The authors generate large datasets (up to several thousands of jobs), and demonstrate that the proposed TS algorithm outperformed the IG algorithm. The contributions of the paper are twofold, first in showing the scalability of metaheuristics by applying them for large problems, and secondly demonstrating that adapting solution methods from other problems can lead to improved results. Siepak and Józefczyk (2014) optimise the total flowtime objective using SS. To justify the use of SS the authors define a simple ILS method which uses the LS operators from SS on its own. The results show that better solutions are achieved by SS, however, the ILS method achieves solutions in a much smaller time. This study is interesting as the authors consider uncertain processing times, meaning that only a rough interval of possible processing times is known for each job. The results demonstrate that even under such uncertainties metaheuristics can obtain solutions of good quality.

Optimisation of tardiness related objectives

The total weighted tardiness criterion is considered in the following studies. H. Zhou, Li, and Wu (2007) apply an ACO algorithm with two pheromone trails, one for selecting the machine on which to schedule the next job, and another for sequencing jobs. In addition, a LS operator is also integrated into the algorithm. Unfortunately, the authors compare their algorithm only with a simple heuristic algorithm, which unfortunately does not give a good overview of the performance of the proposed algorithm. C.-W. Lin, Lin, and Hsieh (2013) also use ACO with two pheromone trails (one for machine selection and the other for job sequencing) and an additional LS method, similar as was done in the previous study. However, they also introduce two additional improvements, initialisation with a heuristic procedure and a machine reselection step. The authors also provide an improved ATC DR, which uses pairwise job interchanges to improve the final solution. The authors performed a thorough analysis on how each element in the improved ACO algorithm affects its performance, and it was shown that the proposed machine reselection step had the largest influence on the quality of the results, whereas the second most important element was the inclusion of the LS operator. Therefore, it seems that the final step can be important for algorithms which construct each part of the solution partially, as it tries to reiterate the solution to improve it and correct certain poor decisions.

Jolai, Amalnick, Alinaghian, Shakhsi-Niaei, and Omrani (2009) consider the objective of maximising the weighted number of jobs that complete just in time. They propose a GA and a combination of a GA with LS approaches for inserting and swapping jobs, and heuristics for creating initial solutions for the GA. The authors provide an extensive analysis of the proposed methods, and demonstrate that the GA with LS methods achieves the best performance. However, this is the only study which dealt with such an objective and follow up studies were never performed. The reason for this is probably that the considered problem variant is too strict, since other researchers mostly focused on criteria that forced the algorithm to find solutions where jobs complete near to their due dates, but did not put a requirement that they need to complete exactly at the time of their due date.

Optimisation of several objectives

Peng and Liu (2004) consider a problem with fuzzy processing times in which they independently minimise makspan, maximum tardiness, and maximum idleness. Three fuzzy scheduling models are defined and a hybrid GA is proposed for solving them. The authors demonstrate the effectiveness on quite small problem instances, and provide several examples and comment on the obtained results of the GA, which provides an interesting analysis from the standpoint of the problem. Still, using fuzzy sets to represent processing times of jobs was rarely considered in other studies, which leaves a lot of possibilities to extend such research. Y. Lin et al. (2011) propose a GA for optimising different single objective criteria, including makespan, total

weighted completion time, and total weighted tardiness. The GA uses only standard operators and is not combined with any additional LS operators or DRs. For all three criteria the algorithm achieved a better performance than the other heuristic methods which the authors considered. This study shows that a well designed algorithm can be efficiently applied to different considered criteria, without any adaptation. It also shows that even standard GAs applied for this problem can perform well. However, as no extensions with LS methods are considered, the study does not show the extent of improvement that could be achieved by including them in the algorithm.

3.3.3 Makespan minimisation with setup times only

The problem type that is most commonly found in the literature is the problem which includes setup times and the makespan is optimised. Especially the last several years saw a rise in the number of research being performed on this problem variant. The research in this section will be divided into three groups depending on which problem instances were dominantly used in those studies to evaluate the proposed methods.

Research considering instances of Rabadi et al. (2006)

Rabadi et al. (2006) proposed a novel metaheuristic called Meta-RaPS, which combines a constructive strategy for generating solutions that are further improved using various LS operators (insertion and interchange). The importance of this work is twofold. First of all, the authors generate problem instances that would later on be used by many other researchers, and which have become the de facto benchmark for the considered problem. Secondly, the Meta-RaPS algorithm was widely used as a baseline to which different researchers compared their results. As such, this work can be seen as a pioneering study in the UPMSP with setup times and makespan minimisation, as it set certain standards used by other researchers until today. Helal, Rabadi, and Al-Salem (2006) propose a TS method in which the initial solution is generated using the SPT rule, and define several perturbation operators for examining the neighbourhood of the current solution. The TS method is compared to a partitioning heuristic against which it performs better. Although the method was not directly compared to Meta-RaPS in this study, later studies which considered both methods demonstrated that the proposed TS algorithm could not match the performance of Meta-RaPS (J.-P. Arnaout, Musa, & Rabadi, 2012).

The problem was further investigated in an initial study of J.-P. Arnaout, Musa, and Rabadi (2008). The authors propose the application of a two-stage ACO, which in the first stage resolves the assignment of jobs to machines and in the second stage determines the sequence in which the jobs should be executed on each machine. The method achieved better results than the TS and PH applied previously for the same problem (Helal et al., 2006). J.-P. Arnaout, Rabadi, and Musa (2009) extend ACO by introducing LS and thoroughly optimising the parameters to improve its performance. The authors extended

the experimental analysis and compared their method to other state of the art methods like Meta-RaPS and TS, showing that the improved ACO algorithm achieves the best performance. J.-P. Arnaout et al. (2012) further extend the ACO algorithm by proposing a novel pheromone update strategy. With this update, the method dominated any of the state of the art methods for different problem scenarios. This line of research is interesting as it demonstrated that ACO can be effectively applied for the UPMSP, regardless of the fact that the solution to this problem is not easily represented in ACO. However, the authors developed an efficient encoding mechanism which resolved this incompatibility and allowed ACO to outperform other state of the art methods. P.-C. Chang and Chen (2011) develop dominance properties when exchanging jobs on the same machine or between different machines, which represent necessary conditions to obtain optimal solutions. These properties are used to generate initial solutions for GA and SA, which improve their performance over the original versions. Unfortunately, only the PH is used for comparison, but not other state of the art methods. However, by cross comparing the results with other studies it seems that the results of the proposed method are worse than those obtained of Meta-RaPS or ACO.

Ying, Lee, and Lin (2010) use a restricted SA which restricts the choices considered during LS to eliminate moves that are ineffective. The authors compare the method to other methods from the literature (ACO (J.-P. Arnaout et al., 2009), Meta-RaPS) and show that it achieves similar results for smaller problems, whereas for larger it significantly outperforms them. Therefore, this study outlines that introducing certain domain knowledge that restricts the search space can be more beneficial than to define more complicated methods. This observation is interesting in a sense that it motives the development of methods that could detect promising parts of the search space and guide the methods to exploit them. Fleszar, Charalambous, and Hindi (2011) propose a multi-start VND, where in each run an initial solution is generated upon which small and large neighbourhood searches are performed. The results show that this method can outperform other state of the art methods, such as Meta-RaPS or ACO. This leads to the conclusion that LS based methods can obtain quite good results, which can match some more complicated algorithms. However, it needs to be stressed out that the VND includes a MIP subproblem that increases the complexity of the method in comparison to standard metaheuristic methods. Haddad et al. (2012) propose a novel metaheuristic named GARP, which is a combination of GA, VND, and PR. The goal of this algorithm is to apply crossover, mutation, LS (using simple job exchange operators) and the PR during the evolution with a certain probability. The method is compared to other state of the art methods and achieves better results. However, the method does consist of many different components, including a LS in which a MILP needs to be solved, which makes it more complex than other metaheuristic methods.

R.O. Diana et al. (2013) use a hybrid metaheuristic which combines CLONALG with GRASP to initialise the starting population, and VND as a

hypermutation operator. The research is further extended by R.O.M. Diana, de França Filho, de Souza, and de Almeida Vitor (2015) with additional experiments. The authors conduct an extensive experimental analysis on two problem sets (those of Rabadi et al. (2006) and (Vallada & Ruiz, 2011)) and compare the results with several methods from the literature, against which the proposed method achieved better results. Although the method achieved significant improvements over existing methods, its complexity also increased as it combines 3 metaheuristics to improve performance, which is a trend that will continue to grow in later studies. R.O.M. Diana, de Souza, Wanner, and Filho (2017) continue with a similar research direction in which they combine immune network optimisation (INO) with VNS, because they indicate that algorithms based on immune networks may have problems with exploration that lead to a premature convergence. The method performs better than other state of the art methods (including CLONALG from the previous study), and again demonstrates the benefit of combining metaheuristic methods with specialised LS methods and operators, thus outlining that a general metaheuristic method cannot solve such problems efficiently on their own.

S.-W. Lin and Ying (2014) apply an ABC which uses a representation with partitioning symbols and a neighbourhood procedure. The procedure first destructs the solution by removing a job from the machine with the highest makespan and then reinserting them in all positions until a complete neighbourhood is constructed. The algorithm demonstrates it efficiency in comparison with other state of the art methods like SA, ACO, TS, and Meta-RaPS. An additional advantage of the proposed method is that it uses a very simple solution encoding that allows the definition of simple LS operators. Thus, in the end the operators used in the algorithm are not complex, but can still match the performance of the state of the art methods. Eroglu, Ozmutlu, and Ozmutlu (2014) apply a GA with real key encoding, in which the solutions are represented as an array of real numbers, for which specialised LS operators are defined. Although this encoding may not be considered natural for such scheduling problems, in a series of experiments the authors demonstrate it can match or improve the results of other methods (ACO (J.-P. Arnaout et al., 2009) and GA (P.-C. Chang & Chen, 2011)). These results are consistent with previous studies in which it was also shown that solution representations that might not feel natural for the considered problem can be used to obtain good solutions (when ACO was applied). Such research provides motivation for investigation of alternative solution representations for scheduling problems.

Cota, Guimarães, de Oliveira, and Freitas Souza (2017) use an adaptive local neighbourhood search that is coupled with LA for learning the probabilities of applying different LS operators that are used in ALNS. The authors examine a wide variety of insertion and local search procedures in the method, and analyse the probabilities of the considered operators obtained by ALNS. The proposed method achieves a better performance than other methods like ACO or AIRP. This paper represents an interesting research direction as it considers an adaptive strategy to apply LS operators,

which could relieve the designer of the the tedious process of selecting the right combination of LS operators for the considered problem. C.-Y. Cheng, Pourhejazy, Ying, and Lin (2021) apply an unsupervised learning based ABC algorithm in which k-means clustering is used to group jobs so that the setup times between the consecutively executed jobs is decreased. Such an idea is quite novel as nothing similar has been done previously. The authors use the instances of Rabadi et al. (2006), but extend them with additional job features that affect setup times. The results demonstrate that the ABC algorithm enhanced with such clustering performs better than without it. However, the method was not compared to any other metaheuristic methods and further examination would be needed to determine its effectiveness.

The most recent studies of this problem set the focus on applying more recent metaheuristics and proposing hybrid methods. For example, Ezugwu, Adeleke, and Viriri (2018) use SOS improved with LS and a heuristic for assigning jobs to machines so that only the sequence of jobs is encoded in the solution representation. This research is extended by Ezugwu (2019) by incorporating LS strategies in their algorithm, improving individual operators and combining the enhanced SOS algorithm with SA. Ezugwu and Akutsah (2018) propose an improved FA in which they use a solution transformation procedure to apply the FA algorithm that is usually used for continuous optimisation. Jouhari et al. (2019) combine SCA with SA in a way that in each iteration SA is used to update the solution prior to application of SCA operators, which is repeated for the entire population until the termination criterion is not achieved. de Abreu and de Athayde Prata (2020) propose a new metaheuristic called GIVP, which is based on a combination of GA, ILS, VND, and PR. J.-P. Arnaout (2019) apply WO in which the solution is constructed similarly as in ACO, by assigning jobs to machines in the first phases, and sequencing those jobs in the second. The WO algorithm achieved better performance than ACO, which shows that the applied two phase solution construction procedure works well regardless of the algorithm. Jouhari et al. (2020) propose a modified HHO, in which each solution is randomly updated either via HHO or SSO operators based on a probability calculated from the fitness of the individual. A similar thing is done in by Ewees, Al-qaness, and Abd Elaziz (2021), with a combination of SSO and FA, where again based on a certain probability either the operators from SSO or FA are used. Al-qaness, Ewees, and Elaziz (2021) perform the same thing between the WOA and FA algorithms. However, it might be difficult to call the former three methods as a real hybrid, since basically the two methods are not really combined, rather just the operators of each are invoked on the solution, but completely independently of the other. Jovanovic and Voß (2021) apply FSS, which is similar to GRASP, however, certain elements that appear in high quality solutions are fixed to put more focus on other elements and obtain better solutions.

Unfortunately, the research performed by the studies denoted in the previous paragraph comes with certain drawbacks. Although the proposed

methods lead to a further improvement in the obtained results, it also resulted in a plethora of new metaheuristic and hybrid metaheuristic algorithms being proposed and applied for the considered problem. This seems to move the research more in the direction of finding the right algorithms or combinations of algorithms which perform the best for the considered problem, which leads to the development of metaheuristics that are overly specialised for solving the considered problem. This can be problematic as it does not result in better understanding of the problem, since it is difficult to understand why a certain metaheuristic works best for the problem. However, all these studies have performed a good experimental examination (performed the testing on instances proposed by Rabadi et al. (2006)) and used a lot of state of the art methods for comparison which allows the readers to select the best method.

Research considering instances of Vallada and Ruiz (2011)

Apart from the previous research, another group of studies used instances generated by Vallada and Ruiz (2011). These problem sets became quite commonly used in subsequent studies, although not as much as those of Rabadi et al. (2006). Vallada and Ruiz (2011) apply a GA coupled with crossover and mutation operators enhanced by LS, test different versions of the proposed GA (with and without the LS enhancement), and compare the results to Meta-RaPS. The experiments demonstrated the superiority of the LS enhanced GA over the standard GA which uses no LS operators and the Meta-RaPS methods. However, the authors did not compare to the ACO method of J.-P. Arnaout et al. (2009), and thus it is not possible to state which of these methods would be superior. Cota, Haddad, Freitas Souza, and Coelho (2014) propose a hybrid metaheuristic called AIRP which is based on combining properties of ILS, VND, and PR. The proposed metaheuristic is quite specialised, however, it achieved a better performance than the GA proposed by Vallada and Ruiz (2011). This suggests that to achieve the best possible results it is required to apply methods adapted for the considered problem.

Avalos-Rosales, Angel-Bello, and Alvarez (2014) use a multi-start procedure, which consists of a solution construction and improvement phase. The procedure creates a new solution in each iteration on which VND is applied with several neighbourhood structures. The proposed algorithm achieved a better performance than the best known solutions obtained by Vallada and Ruiz (2011). Nohra Haddad, Perdigão Cota, Jamilson Freitas Souza, and Maculan (2014) propose a novel metaheuristic method denoted as AIV, which combines ILS and VND. The procedure uses simple LS operators and is compared to GAs from previous studies to demonstrate its effectiveness. The previous two studies focused on simple LS based methods which can outperform the more general metaheuristic methods. L. Wang, Wang, and Zheng (2016) apply EDA embedded with an IG procedure. The authors derive certain properties about neighbourhood operators and use that information to build a probabilistic model in EDA to direct the search to promising areas.

Unfortunately, the algorithm is compared only to the GA of Vallada and Ruiz (2011), and not to any of the previous three new metaheuristic methods. Although all previously proposed methods outperform the GA proposed by Vallada and Ruiz (2011), it is not known which of the proposed methods in the end achieves the best performance, as no comparison was performed between them.

Santos, Toffolo, Silva, and Berghe (2016) perform a comparison of 5 stochastic LS methods with 6 neighbourhood structures. They examined SA, ILS, and 2 hill climbing variants, out of which SA achieves the best results. The results of this study are interesting and surprising as they show that a SA method with a simple random neighbour selection strategy can outperform more sophisticated algorithms like a GA (Vallada & Ruiz, 2011) and AIRP (Cota et al., 2014). Such a result suggests that increasing the complexity of methods too much can also worsen their performance and in some cases it might be better to use simpler metaheuristic methods. Terzi, Arbaoui, Yalaoui, and Benatchba (2020) apply a hill climbing method which generates a new solution in each iteration and then the VND method with five neighbourhood operators is applied. The proposed method was compared to the immune network optimisation of R.O.M. Diana et al. (2017), and demonstrated that it has a better convergence, but given enough time, neither method was superior. As can be seen, the number of studies using the instanced of Vallada and Ruiz (2011) is rather limited, especially those published in recent years.

Research considering other problem instances

Anagnostopoulos and Rabadi (2002) propose a SA method with five neighbourhood operators for interchanging and inserting jobs on a single machine and between two machines. The proposed algorithm obtained optimal solutions on all smaller problems sizes. However, the authors consider only quite small problem instances, and thus the scalability of the method is not clearly demonstrated. Niu, Zhou, and Zhou (2011) apply CLONALG coupled with a LS operator. However, the authors generate a new problem instance set and compare the results only to a simple GA and SA methods against which the proposed method achieves better results. Because of the lack of comparison with other state of the art methods, it is not possible to state how this method would perform in general. Keskinturk, Yildirim, and Barut (2012) propose an ACO algorithm in which each node consists of subnodes that represent machines on which the job can be scheduled. The ants visit all nodes, but not each subnode, and out of this tour a feasible schedule is constructed. In addition, the authors also propose a GA variant coupled with LS operators to serve for comparison with ACO. However, it is difficult to asses the quality of this ACO variant as the authors use their own instances. Regardless of that, the paper outlines the importance of using LS in metaheuristics to significantly improve their performance, since for both ACO and GA the results were significantly improved when using LS. This suggests that the general operators

used in metaheuristics are not powerful enough for efficiently tackling this problem.

A GA is proposed by C.-J. Liao, Lee, and Tsai (2016) with a simple constructive heuristic based on the adjacent index value that considers the flexibility of jobs. The GA uses the constructive heuristic for initialising the population and a LS is applied after the mutation operator. The authors consider a slightly different problem variant, where the setup times depend on certain job attributes, and thus the authors create their own problem sets. The algorithm is evaluated against other methods from the literature and demonstrated its efficiency for the considered problem variant. Tozzo, Cotrim, Galdamez, and Leal (2018) apply a GA which uses a simple constructive heuristic to generate the initial population and three LS operators that are applied after the genetic operators. They also use a VNS with the same initial solution construction method as GA and the same three LS operators to examine the neighbourhood. The results show that the VNS method achieves better results than the GA, which suggests that simple LS based methods might be more suitable than metaheuristic methods for the considered problem. Unfortunately, no comparison with other methods was performed to gain a better overview of the performance of these methods, nor did the authors use existing problem instances to validate their methods.

Conclusion

As the overview demonstrated that this problem has been gaining more and more attention over the years. The reason for this is mostly due to the fact that the initial studies made their problem instances available, which allowed other researchers to reuse them and compare their own methods against previous results. Therefore, this problem became a kind of a benchmark that was used by different researchers when proposing their novel hybrid algorithms. This can especially be seen from the research performed in recent years, where a lot of studies use novel metaheuristic or hybrid algorithms. Naturally, there are two sides of such research. On the positive side, such research inevitably leads to an improvement of existing results, and methods that can better solve the problem at hand. However, as already outlined, such research rarely led to new findings about the problem, but was solely focused on hybridising or adapting different methods. As such, one could argue that such metaheuristics become less and less general, and more similar to problem specific heuristics. As such, it would be a good direction that such new algorithms are tested on a variety of problem types to see whether the improvements in the algorithm lead to a better performance in general, or if the algorithms are overfitted on the problem instances that are used for testing.

3.3.4 Problems with setup times for other criteria

Problems which deal with setup times but different optimisation criteria have been studied in a significantly smaller amount. Regardless of this, most of the

studies performed for this problem variant are connected since they use the same instances or compare with algorithms from other studies.

C.-L. Chen and Chen (2008) propose hybrid metaheuristic consisting out of a combination between TS and VND for the minimisation of the weighted number of tardy jobs. The experimental results demonstrate that the proposed hybrid method achieved better results than the basic TS algorithm, which shows the importance of integrating LS operators into metaheuristics to improve their search capabilities. J.-F. Chen (2009) apply SA for a similar problem in which the total tardiness needs to be optimised, with the addition that some jobs have deadlines that must not be be broken. In the first step, ATCS is applied to generate an initial solution, which is further refined using two additional procedures. SA uses several neighbourhood operators to generate new solutions, some of which perform changes on single jobs, whereas others modify an entire chain of jobs. The proposed procedure was able to improve the initial solution obtained by ATCS. Unfortunately, it was only compared to ATCS and a simple random descent heuristic that applied the same neighbourhood operators as SA but did not accept any worse solutions. However, this comparison demonstrated that accepting worse solutions is mandatory to escape local optima, otherwise the algorithm quickly gets stuck in them.

S.-W. Lin, Lu, and Ying (2010) consider the same problem and propose an IG procedure that first destructs the solution by randomly removing jobs from machines, and then in the next phase it inserts them to construct a complete solution. After that, a LS procedure is performed by exchanging the jobs on machines to further improve the solution. The method achieved a better performance than other methods applied for different problems (J.-F. Chen, 2009) and achieved a slightly better performance. Ying and Lin (2012) provide a follow up on the previous study and propose the application of ABC. The method is enhanced with new neighbourhood solution generation methods and new LS based operators. In comparison to the methods proposed in the previous two studies, the proposed ABC algorithm achieves a better performance. A TS method, which uses 8 LS operators, is examined by J.-H. Lee, Yu, and Lee (2013) for minimising the total tardiness. In addition to standard operators that exchange a single job, the authors define several operators which insert or swap groups of jobs between machines. The authors show that the proposed method performs better than SA (J.-F. Chen, 2009) and IG (S.-W. Lin et al., 2010). Unfortunately, all three algorithms use different LS operators, and as such it is not possible to claim whether the better performance of TS stems from the LS operators or the TS itself. Therefore, based on the previous studies, it is still impossible to say whether the specifics of the algorithm or LS operators used have more influence on its performance, which certainly represents an open research question that could be further examined.

The total total weighted earliness and tardiness criterion is optimised in the following studies. Raja, Arumugam, and Selladurai (2008) combine a

GA with a fuzzy logic approach since standard procedures had difficulties in handling such an objective. Therefore, the GA was applied for generating schedules when different weight combinations of the earliness and tardiness criteria were considered, whereas the fuzzy logic approach was used to select the best combination of weights. The proposed approach is compared to several standard GAs, and demonstrated its superiority. However, when the method is compared to an existing TS only for total weighted tardiness minimisation, the algorithm achieves the same performance. Although the combined GA and fuzzy method is interesting from the perspective that the weights for the objective are adapted automatically, it is questionable which benefits such an approach could have compared to more standard MO problems that could optimise both objectives simultaneously. Zeidi and MohammadHosseini (2015) use SA as a LS procedure that tries to improve a solution from the last generation of a GA, and show that such a hybrid method achieves better results than other methods from the literature (J.-F. Chen, 2009). de C. M. Nogueira, Arroyo, Villadiego, and Gonçalves (2014) propose a hybrid GRASP that uses a greedy solution construction procedure that balances between greedily assigning jobs to the machine with the best objective value and randomly distributing jobs across machines. Additionally, the algorithm also uses PR and an ILS as an improvement phase. The authors only analyse the different GRASP versions, showing that the most sophisticated one achieves the best results, however, they do not compare to any methods previously used for the considered problem. In addition, the authors also do not demonstrate whether the inclusion of PR or LS is more important for the performance of the algorithm, which would be an interesting insight on which method is more important to include.

As can be seen from the previous overview, this problem has been rarely considered in comparison to the makespan criterion. Especially recent years did not see any new studies dealing only with setup times without any other major constraints included.

3.3.5 Batch scheduling problems

Regarding problems which consider batch scheduling, the research can be roughly split into studies which deal with serial and parallel batch scheduling problems. Batch scheduling problems add an additional complexity in the decision process, since now in addition to determining the sequence of executing jobs and their allocation to machines, it is also required to determine their grouping into batches that are being executed.

Serial batch scheduling problem

One of the first studies dealing with serial batch scheduling was done by D.-W. Kim, Kim, Jang, and Frank Chen (2002), who consider a problem with lots and setup times for minimising the total tardiness objective. In this problem, jobs represents lots that consist of several items that need to be processed. All items in the job belong to the same lot, which means that no setup time is

incurred between them and that they all have the same processing time. The goal is to group items belonging to the same lot to reduce setup times. A SA method is proposed, in which the neighbourhoods are generated considering both lots and items. The results show that using neighbourhood structures which work on jobs, rather than individual items, greatly improves the results. The method is further validated by D.-W. Kim et al. (2003), where the authors compare the SA algorithm with DRs and a novel heuristic, and demonstrate that the SA method achieves the overall best results among all the methods. This serves to further outline that metaheuristics can obtain superior results in comparison to other solution methods.

One of the most commonly considered problem variants for serial batch scheduling was introduced by LOGENDRAN and SUBUR (2004), in which the authors consider a problem with job and machine release times as well as machine eligibility constraints. In this study the authors consider a problem variant in which batches are already known, however, they can be split into smaller batches to achieve a better performance. The authors minimise the total weighted tardiness using several TS algorithm variants, which they investigate in depth. They show that smaller and medium sized instances the algorithm that focuses on exploitation performs better, whereas on the larger instances the TS algorithm which focuses on diversification achieves better results. Bozorgirad and Logendran (2012) consider the same problem, however, the authors focus on the optimisation of the sum of the total weighted flowtime and total weighted tardiness. They use TS with different strategies that focus either on intensification or diversification of solutions during optimisation. The authors perform an extensive experimental analysis of different algorithm variants and compare the results with those of an exact method to demonstrate that the method can achieve optimal solutions for smaller problems. Unfortunately, no comparisons with other metaheuristics were used for comparison, and the examination was done using mostly smaller problem instances. Shahvari and Logendran (2015) include setup times to the previous problem and optimise the sum of the weighted flowtime and total weighted tardiness. The authors apply a TS procedure with several solution initialisation techniques, and LS operators that insert or swap jobs between batches. Shahvari and Logendran (2017b) extend the previous model by considering that batches have lower bound on the size, meaning that batches below a certain size cannot be constructed. An extended TS algorithm is proposed, which in the first level joins jobs into batches, then determines sequence of batches and their allocations to machines, and finally sequences jobs within the same batch. The proposed TS was compared to a an exact method and showed good performance on the generated instances. All previous works focused on applying TS and incrementally improving it for the considered problem. However, there is no clear comparison between all the versions of the proposed TS methods and as such it is difficult to judge on the efficiency of the individual variants.

Na et al. (2006) consider the problem with the goal of minimising the weighted tardiness with setup times. Thea apply a SA method initialised with a simple DR, and compare its performance against several simple DRs, which it all outperforms. However, because the comparison of the method is performed mostly against simple DRs, it is difficult to asses its performance. W.-L. Wang, Wang, Zhao, Zhang, and Xu (2013) investigate a similar problem but consider only job eligibility constraints and minimise the makespan, for which they use an improved DE. They compare their proposed algorithm to other standard metaheuristics to demonstrate its performance. The authors also provide a demonstration of this algorithm on a practical problem from steel coil production. Apart from the previous studies, no further investigation was performed in the serial batch scheduling problem with jobs splitting. Unfortunately, the methods proposed in the previous studies are mostly not comparable between each other due to each of them considering a different problem. However, the studies focus on problems with different additional constraints (gradually added to the problem) which increases the complexity of the problem and as such these studies provide valuable insights on how to deal with such problems, especially since most are motivated by real world examples.

Parallel batch scheduling problem

The parallel batch scheduling problem variant received a much larger attention and more problem variants were investigated. S. Xu and Bean (2007) investigate a simple problem with makespan minimisation. A GA with random key encoding is proposed, in which the integer part of the number determines on which machine the job is executed, whereas the fractional part determines in which sequence the jobs are executed. Jobs are collected into batches in the order of their execution until the batch of the desired size is constructed. As no other research was previously done on this problem, the authors compare the proposed algorithm to an exact method. They show that the GA can match the performance for smaller and improve it for larger instances over the exact solving method, in a smaller amount of time. Klemmt et al. (2009) examine a problem with incompatible job families, release times, and the minimisation of the total weighted tardiness objective, which is motivated by wafer fabrication facilities. They propose a VNS algorithm, which uses several neighbourhood operators that work both with individual jobs and job batches. VNS was compared to a mixed integer programming approach, and demonstrated it can slightly outperform the MIP approach and a simple DR, which speaks in favour of applying metaheuristics. Celano, Costa, and Fichera (2008) model a scheduling problem from cell manufacturing, in which they consider setup times and additional resources in the form of workers. The objective was to optimise the total tardiness as a primary objective, and the makespan as the secondary, for which the authors use SA. Based on the obtained results, the authors make conclusions about how different process scenarios affect the optimised criteria, and draw conclusions based on it. This study provides

interesting insights on which elements in the problem are the most critical for performance (like the capacity of workers), which can be used to put more focus on certain elements of the problem when designing new solution methods.

After the previous studies, the parallel batch scheduling problem was not considered for some years until it was revisited by H. Lu and Qiao (2017), who model a scheduling problem of a heating process. Instead of the standard processing time, all jobs have heating and soaking times associated to them, but with no additional constraints. The objective is to minimise the total energy cost incurred from the energy consumed during the heating and soaking processes. The authors apply a standard GA, and an adapted GA which uses genetic operators that consider more information about the problem. However, no other methods are considered for comparison, which makes it difficult to asses the quality of the proposed methods. Joo and Kim (2017) consider a scheduling problem of delivery trucks with heterogeneous capacities that need to be scheduled to deliver jobs to customers. The objective is to minimise the makespan and the authors propose a single stage GA that uses a permutation array to represent the sequence of jobs. Based on that sequence DRs are used to assign jobs to machines, group jobs into batches and assign them to delivery trucks for transportation. The study deals with an interesting and difficult problem, and the authors investigate a very interesting concept in which a metaheuristic does not generate the entire solution, but rather uses simple heuristics to help it build the entire schedule. Although in that way it might not be possible to reach optimal solutions since the search space is reduced, it simplifies the application of metaheuristics for a given problem as not all decisions need to be handled by it, which could be difficult to encode and ensure the feasibility of the solution.

Shahvari and Logendran (2017a) consider a problem with job and machine release times, machine eligibility constraints, setup times, unequal batch sizes, job sizes and machine capacities. In addition, they introduce additional resources in the form of operators that have different skill levels, which are required to perform setups and run the batches on machines. Two objectives, the makespan and production cost, are minimised, for which the authors apply a hybrid MO PSO algorithm for the optimisation of this problem. The goal of it is to split the solution construction, so that the assignment of jobs on machines is determined by PSO, and the allocation of jobs to batches and their sequencing is determined by an auxiliary heuristic. Due to the high number of additional constraints, the considered problem is quite difficult to tackle which is also reflected in the added complexity of the proposed method. However, the results demonstrate that even such difficult problems can be efficiently solved using metaheuristics.

Arroyo and Leung (2017a) consider the problem with release times, varied job size, and makespan minimisation. The authors propose an IG algorithm which iteratively applies construction and destruction phases on a solution to improve it. They also implement several other metaheuristics including GA, ACO, and SA that are adapted from the literature. The results show that the

proposed IG method outperformed other metaheuristic methods consistently over the considered problem instances. Arroyo, Leung, and Tavares (2019) consider the same problem, but with total flowtime minimisation. The authors apply the IG of Arroyo and Leung (2017a), however, they use different solution construction and LS procedures. The authors also apply DDE, SA, and ACO, however, the results show that the proposed IG method can outperform all of them. Since the IG method is quite simple, the previous two studies demonstrates that even simple LS based methods can match the performance of more complex metaheuristics even for more complex problems which include a wide range of additional constraints. As such, it could make more sense to focus more on the design of LS based procedures and solution construction methods, than on new metaheuristic methods or extending and adapting them for a considered problem.

S. Zhou, Xie, Du, and Pang (2018) consider the same problem as Arroyo et al. (2019), however, they consider the objective of minimising the makespan. The authors apply a GA with random key encoding to determine the sequencing of jobs and their allocation to machines. During solution evaluation the jobs are allocated to batches and sequenced on machines using a first fit heuristic adapted from the bin packing problem. The compare their method against the IG of Arroyo and Leung (2017a), and show that the proposed GA achieves a better performance. Since the GA does not use any LS operator, these results are quite surprising as they demonstrate that a GA with no problem information can match the performance of LS based methods which include such knowledge. The reason why this happened is not clear, but could be that for such a problem variant a simple GA is already powerful enough to find good solutions. However, other studies usually came to the opposite conclusion, therefore it is difficult to outline whether the conclusions of this study are generally applicable. Shahidi-Zadeh, Tavakkoli-Moghaddam, Taheri-Moghadam, and Rastgar (2017) consider the same problem in a multi-objective scenario. They perform a comparison of different MO methods for simultaneous minimisation of the makespan, weighted tardiness and earliness, and purchasing cost. The authors test the MO HS, ACO, and PSO variants, as well as the NSGA-II algorithm. An extensive analysis has been performed which shows that the MO HS algorithm achieves the best performance. This research is important as it constitutes one of the rare investigations of multi-objective problems in the context of batch scheduling, in which the authors try to obtain a Pareto front of solutions.

S. Lu, Liu, Pei, T. Thai, and M. Pardalos (2018) consider deteriorating job processing times and mandatory machine maintenance periods that have to be scheduled for makespan minimisation. After a maintenance period the execution time of jobs will be shorter. The authors propose a hybridisation between ABC and TS, in which TS is executed in each iteration on several individuals in the population to improve them. The solution encoding specifies the assignment of jobs to machines, whereas the grouping and sequencing of jobs on machine is done by an additional procedure. The proposed approach

obtained better results and demonstrated a better convergence than other similar metaheuristic algorithms. A batch scheduling problem with release times and unequal job sizes is investigated by Zarook, Yaser et al. (2021). The authors apply a GA which uses the real key encoding to represent solutions. However, what the authors use is not really a real key encoding, but rather some kind of a matrix encoding in which only the association of jobs to machines is specified. The construction of batches and the sequence of batches are then determined by two simple DRs. The GA is compared with several proposed DRs, and the results show that it cannot outperform all the DRs, but also that it has a smaller computational time than DRs. Unfortunately, it is not clear why this happens, as no explanations are provided in the study, as already stated in a previous section.

Conclusion

The overview shows that out of the two batch variants, the serial batch scheduling problem is much less researched in comparison to the parallel batch scheduling variant. In the serial batch scheduling problem, the total weighted tardiness objective was optimised most often, and usually only the machine eligibility and setup time constraints have been considered. The parallel batch scheduling problem received significantly more attention in the last several years, where different problem types were investigated. The reason for this is that different problems from the real world were modelled and solved. This lead to the definition of quite complicated problems, which required different adaptations of metaheuristics to achieve the best possible results. Unfortunately, the many problem variants make it difficult to compare different methods, however, some researchers did focus on common problems and incrementally improved the proposed methods for that problem. As this area received a wide attention in the last several years, it is likely that research in this area will be further intensified in the following years by considering new problem variants and solution methods.

3.3.6 Problems with additional resources

The research which considered additional resources can be divided into two distinct groups based on the kind of resources that are considered. In the first group the authors consider resources in the form of human workers, whereas in the second case the resources represent some materials that are required for the execution of jobs

Worker based resource problems

The next several studies consider a problem with limited human resources, setup times, and the objective of minimising the makespan. Cappadonna, Costa, and Fichera (2013) propose a GA that uses a single permutation encoding scheme that specifies the order of jobs. The association of machines and workers is done by a heuristic rule that assigns the job to a machine and worker that can finish its processing the fastest. Initial comparisons with a

MILP model demonstrate the effectiveness of the proposed GA. Cappadonna., Costa., and Fichera. (2012) use three GA variants for the problem, which differ in the encoding schemes they use. The first encoding is the same as the one proposed by Cappadonna et al. (2013). The second encoding uses multiple arrays to represent the order of jobs, but also their allocation to machines and workers. The final GA combines these two, so that the algorithm first uses the permutation scheme and then at one point switches to the multiple array encoding. The hybrid GA achieved a better performance than both individual variants, whereas the other two perform equally well for larger problems and the representation with multiple arrays performs better for smaller instances. This research is further extend by Costa, Cappadonna, and Fichera (2013) with a deeper experimental analysis.

The previous studies provided several important contributions. First of all they showed the effectiveness of using simple encodings combined with heuristic decoding schemes. They also demonstrated that such schemes can sometimes impose limitations on the solution, which is not present when a complete encoding of the solution is used. On the other hand, the complete solution encoding allows the algorithm to investigate the entire solution space, however, the operators and decoding schemes are more complicated and usually increase the computational time. Finally, the authors proposed a novel hybrid GA that uses two encoding schemes during evolution. The motivation behind this idea is to incrementally solve the problem, by first finding a good sequence of jobs with the simpler encoding, and then trying to fine tune it and find the best association to machines and workers with the extended encoding. This idea is interesting as it suggests that more complex problems could be efficiently tackled in a way to incrementally solve each part of the problem. Since such an idea was seldom used, it it would be interesting to see how such a method would also perform on other problem types as well.

L. Zhang, Deng, Lin, Gong, and Han (2021) consider a problem with setup times and worker learning effects. In this problem, the workers perform setup tasks and become more skilled over time, which means that they perform those tasks in a shorter time. Additionally, not all workers are equal, and some perform better than others. The objective is to minimise the makespan and total energy consumption (machines consume different amount of energy when being idle or processing jobs). The authors apply CEA to solve the problem, which uses a three vector representation for determining the job sequence, machine allocation, and worker allocation. The results demonstrate that the method performs better than other MO algorithms. The significance of this work is that it models a real world problem in more detail, by considering the fact that the performance of the workers will improve to a certain degree during time as they become more skilled.

Material based resource problems

J.-F. Chen and Wu (2006) consider the minimisation of total tardiness in a problem where secondary resources cannot be used to an arbitrary extent since

they are expensive. In addition, setup times are associated to the attachment and deattachment of these resources to machines, and each machine cannot process all jobs. A metaheuristic method based on the combination of TS and improvement algorithms is proposed, and shows superior performance when compared to SA and the ATCS rule. A similar problem is considered by J.-F. Chen (2006), where the maximum tardiness is minimised. A metaheuristic procedure RRT, based on guided search and tabu lists, is proposed and compared to EDD, and SA. The proposed algorithm significantly outperforms both of these procedures. However, the the previous studies propose algorithms which seem to consist of quite similar building blocks (LS operators, underlying heuristics), and as such they main difference is only in the main metaheuristic they use (TS or RRT), as well as the objective they minimise.

For a few years auxiliary resources were not considered in any studies until Bitar, Dauzère-Pérès, Yugma, and Roussel (2014) modelled a problem from the photolithography workshop as an UPMSP. These resources need to be transported to machines and jobs that require the same resource cannot be used on two machines in parallel. The problem also includes setup times and the goal is to minimise the total weighted flowtime. The authors use a GA enhanced with a LS procedure to solve such a problem, but only examine the performance of the proposed algorithm on different scenarios, and provide no comparison to other existing methods. Low, Li, and Wu (2013) and Low and Wu (2016) minimise the makespan criterion for a problem with non-renewable resources and machine eligibility constraints. In this problem, processing times are controllable and depend on the amount of resources used during processing. The authors proposed two ACO variants. The first simultaneously determines the assignment of jobs to machines and allocation of additional resources. The second algorithm first schedules all jobs by their default processing times, and then allocates the resources. The results shows that the first variant that considers both decisions simultaneously achieves better results. This is inline with previous observations of other studies which also demonstrated that considering all scheduling decision simultaneously leads to better results. The study is interesting from the point that it considered resources in a different way, meaning that they can be used to improve the execution of the schedule, but the entire problem could also be constructed without any resource utilisation. Thus, the problem is basically finding the best allocation of resources to obtain the largest decrease in the optimised criterion.

Afzalirad and Shafipour (2015) consider a problem with makespan minimisation and apply two GA variants. In the first variant, a three level solution encoding is used, in which the first part represents the sequence of jobs, the second their allocation to machines, and the third the priority of assigning additional resources to jobs. The second GA uses a two level encoding without the priorities of additional resources and applies a heuristic for resource allocation. The results suggest that the first encoding scheme which includes all the information performs better, but the algorithm required more computational time when using it. This is consistent with other studies like

that of Costa et al. (2013) which demonstrate that considering all scheduling choices leads to a better performance of the method. Afzalirad and Rezaeian (2016b) examine a problem with release times, precedence constraints, machine eligibility restrictions and setup times, which is based on a problem from shipbuilding. The goal is to optimise the makespan objective, and the authors apply GA and AIS (which includes parts of an IG procedure) for the considered problem. The authors use an encoding scheme that represents the entire solution, and demonstrate that good solutions can be obtained even on such a problem which considers several additional constraints.

A scheduling problem in which the processing times and due dates are not known with certainty and are modelled as fuzzy variables is considered by Torabi, Sahebjamnia, Mansouri, and Bajestani (2013). The problem includes setup times, release times and additional resources need to be available for a machine to process the job. Three objectives are minimised: makespan, total weighted tardiness, and total weighted flowtime. The authors propose the application of a MO PSO algorithm, which includes a concept of dominance to obtain a Pareto front of solutions. They compare the proposed MO PSO with a standard MO PSO algorithm, but not with other MO algorithms like NSGA-II to really obtain a notion of the quality of the proposed method. In addition, the authors only consider diversity metrics when comparing the MO algorithm variants, but not convergence metrics, which does not give a definite answer on which algorithm converged better to the real Pareto front.

MANUPATI, RAJYALAKSHMI, CHAN, and THAKKAR (2017) examine the problem with setup times, job ready times, and auxiliary resources with the objective of minimising the makespan, total weighted flowtime, total weighted tardiness and machine load variation. Similar to Torabi et al. (2013), they also consider that scheduling environments are usually uncertain, therefore the processing times and due dates are modelled using fuzzy sets. A novel MO algorithm is proposed, which is based on the NSGA-II algorithm with integrated elements from the immune based metaheuristics. The algorithm is compared to standard NSGA-II and a standard MO PSO, and performs better than standard methods from the literature in an analysis of the obtained Pareto fronts. However, no MO performance measures are used to numerically examine the quality of the obtained Pareto fronts. Both previous studies are interesting from the point that they considered uncertainties for various job properties, and adapted standard metaheuristics for such an environment. Since uncertainties have rarely been considered in existing studies, this direction has a lot of potential for further investigations.

Özpeynirci, Gökgür, and Hnich (2016) consider a problem in which certain tools need to be loaded to machines. Each tool can be used by a single machine and has to be removed after processing, but tool switching times are negligible and the tools never break down. The authors use TS and show it obtains good solutions for makespan when compared to some exact methods. However, the authors consider only smaller instances, with up to 15 jobs and 3 machines, because of which the scalability of the approach was not demonstrated. long

Zheng and Wang (2016) consider a problem with makespan optimisation, in which a finite number of renewable resources is available. A job cannot be processed if at least one unit of a resource is not available during its entire execution phase. They propose the application of FA with an initial solution generation procedure based on two DRs. The algorithm is compared to a standard GA and shows a better performance across the considered problems. Zheng and Wang (2018) extend the previous problem by considering the total energy consumption objective together with the makespan. Although the machines are unrelated, they have different operating speeds that can be changed. Increasing the speed lowers the processing times of jobs, however, it increases the energy consumption. An additional constraint is that machine speeds can be changed only before a job starts executing, and the authors use an encoding scheme in which for each job the speed of the machine is specified. The authors propose a collaborative MO FA enhanced with initial solution construction and three LS operators. A comparison with the standard NSGA-II algorithm demonstrates that the proposed algorithm achieves a better performance. This work is also important as it switches the focus to green scheduling problems, a topic which is gaining more attention in a wide range of optimisation problems, but was as of yet not considered significantly in the UPMSP, especially combined with auxiliary resource constraints.

Vallada, Villa, and Fanjul-Peyro (2019) study a problem with a single renewable resource and the objective of makespan minimisation. The authors propose two metaheuristics, an enriched SS and IG algorithm, in which a restricted LS is embedded. Additionally, the starting solution is constructed using a heuristic from a previous study (Villa et al., 2018). The results demonstrate better performance of the proposed methods in comparison to previous methods (Villa et al., 2018), but when compared to each other the IG method achieved significantly better results. However, the comparison with the previous method is not completely fair, since the method was used to generate the starting solution for both SS and IG, which means that both methods will perform at least equally good as that method. Al-harkan and Qamhan (2019) consider several resource types with a given amount, and each job requires a certain amount of resources to be executed. They minimise the makespan and apply a two stage metaheuristic that is a combination of VNS and SA. The algorithm starts with an initial solutions constructed by several DRs and iteratively applies a LS operator, however, it uses a SA acceptance test for the accepting the solution. The authors only investigate different VNS variants they proposed and do not perform a comparison to other methods from the literature. However, the authors show an interesting conclusion in which a VNS that used only one neighbourhood structure achieves the best performance, and not the one which uses all of them. Unfortunately, the authors did not investigate more neighbourhood operator combinations, which could give a definitive answer whether only one operator is enough or the set containing all operators simply included operators that were not efficient. The problem of makespan minimisation subject to release times, setup times, and

renewable resources is examined by Al-harkan, Qamhan, Badwelan, Alsamhan, and Hidri (2021). The authors propose the application of a modified HS, in which the solution encoding was adapted to consider the resource constraints. The algorithm managed to outperform other methods from the previous study, but not consistently as the proposed method showed some limitations for problem instances with certain properties.

Yepes-Borrero, Villa, Perea, and Caballero-Villalobos (2020) investigate a problem with setup times, renewable resources and makespan minimisation. The authors first propose several heuristic algorithms that construct the initial solution and apply a repair procedure to ensures that the resource constraints are not violated. They also apply a GRASP procedure to solve the considered problem. The authors consider two metaheuristic variants, the first which does not consider resource constraints during the construction of the solution and uses a procedure to repair the final solution, and the second which takes the resource constraints into account while constructing the solution. The results demonstrate that for smaller problems there is no difference between the two variants, however, for the larger problems it is demonstrated that the method that takes resources into account during optimisation performs better. This shows that it is important to consider all problem elements during the construction phase of the problem, and that using a solution repairing procedure might not be efficient enough for larger problem instances. Yepes-Borrero, Perea, Ruiz, and Villa (2021) consider a problem with setup times, in which the resources are not tied with processing jobs, but rather with setup times. The goal is to minimise the makespan and maximum resource consumption. The authors apply several MO methods like NSGA-II, MO iterated greedy search (MOIGS), and restarted iterated Pareto algorithm (RIPG). Additionally, they also propose a truncated RIPG (T-RIPG) method which incorporates several improvements over the original, like an additional repairing mechanism that inserts idle times into the schedule to avoid resource overload. An extensive experimental analysis shows that the proposed algorithm outperforms others from the literature.

M.-Z. Wang, Zhang, and Choi (2020) consider a problem in which resources represent raw materials that are stored in special containers. When the containers are opened, the resource it contained starts to deteriorate until a certain deadline when it becomes unusable and another container needs to be opened. The authors propose a MO hybrid PSO with a greedy solution construction and embedded LS operators for the minimisation of the total completion time and material cost. The method is compared to other MO algorithms (NSGA-II and SPEA2) and shows a better performance. The paper investigated an interesting variant of UPMSP with resource constraints, as such a formulation was not investigated previously, and the authors motivate the need for such a variant from several real world examples in which resources have a limited life (e.g. food industry). Pinheiro, Arroyo, and Fialho (2020) deal with a problem with setup times in which the total tardiness objective is minimised. In the considered problem the resources are being supplied to

machines with a constant rate. The motivation for such a problem comes from the steel-making industry, in which liquid steel is supplied for production. Therefore, it is required to insert idle times into the schedule to delay the execution of jobs until the required amount of resources has been supplied to the machine needed for the entire execution of the job. The authors apply SA and IG to solve the problem, out of which the IG method performed better, but its computational time is higher than that of the SA method. Therefore, it would have been interesting to see how both methods compare to each other if given the same budget.

Conclusion

The review demonstrated that most studies focused on resources in the form of additional materials that need to be allocated to machines. Most often, these resources are renewable, which means that they are replenished over time, and usually only a single type of resource is required. However, more and more research focuses on specific resource types, for example non renewable resources, or resources which can deteriorate and similar. Since resources are a very general constraint and appear in different real world scenarios, it is expected that more and more research will be focused on these problem, but also that more resource types will be considered.

3.3.7 Problems with release times

This section will examine research which deals with problems that include release times.

Optimisation of due date related criteria

Bank and Werner (2001) consider the minimisation of the total weighted tardiness and earliness penalties with a common due date. The authors apply several iterative algorithms which include SA, TA, iterative improvement, and a multi-start heuristic. They conclude that multi-start heuristics usually do not perform as well as single start heuristics, but that there is no clear winner among the other metaheuristics. This conclusion is consistent with the no free lunch theorem in that it suggests that no single method will achieve the best results across all problems.

S.-I. Kim, Choi, and Lee (2006) consider the minimisation of the total tardiness objective with setup times, and they apply a TS method. Three initial solution construction methods are tested, which are based on DRs to order the jobs and then schedule them on machines. Since a solution can have a huge neighbourhood that would need to be searched, two candidate list strategies are introduced to limit the neighbourhood. The first strategy considers only a single job per machine for swapping or inserting, while in the second strategy all tardy or non tardy jobs (depending on the iteration) are considered for being exchanged. The experiments show that better results are obtained by using the second strategy. The authors compare their results against a TS method from the literature, and demonstrate that the proposed extensions

lead to a better performance. S.-I. Kim, Choi, and Lee (2007) extend the previous research by additionally considering SA. For both methods the initial solutions are generated using 3 DRs, whereas for the improvement phase two neighbourhood operators are used for interchanging and inserting jobs. The experimental results show that TS performed best when searching a larger neighbourhood similar as shown previously, and that SA could not match its performance. However, it would have been interesting if the authors gave a reasoning why SA performed so poorly in comparison to TS regardless of the fact that both methods used the same operators. Logendran, McDonell, and Smucker (2007) consider a problem with the objective of optimising the total tardiness, in which the authors consider both job and machine release times. The authors investigate the TS algorithm and focus on evaluating its performance when using different initial solution generation methods, tabu list sizes and strategies. As such, this study is interesting from the point that it tries to provide insights on the most important decisions of TS to achieve the best performance. Unfortunately, the proposed TS was not compared to TS variants from the previous studies, therefore it is not known which would perform the best.

Y.-K. Lin and Hsieh (2014) consider a problem with setup times, release times, and the total weighted tardiness objective. They propose the application of EMA coupled with the ATCSR rule for generating the initial solution, and two local search operators to increase its efficiency. The results demonstrate that the proposed method achieves a better performance than other state of the art metaheuristics used for this problem, like ACO and TS. Although the considered algorithm achieved a good performance, it was not used in any further studies. R.O. Diana, de Souza, and Filho (2018) consider the problem with setup times and total weighted tardiness minimisation. The authors propose a hybrid metaheuristic that combines ILS with VND in a way that VND is executed as a LS operator in each iteration of the algorithm. They are compared to previous metaheuristics (Y.-K. Lin & Hsieh, 2014) and show a better performance. This research seems to indicate that it makes sense to combine the concepts of ILS (using perturbation operators) and VNS (changing neighbourhood operators) to obtain better results. Marinho Diana and de Souza (2020) examine the problem of minimising the total weighted tardiness with setup times. The authors perform an analysis of including VND in different metaheuristics (IGS, ABC, GA) instead of a using a simple LS operator. They test three neighbourhood operators and VND strategies, as well as examine the influence of different parameters like the order of the application of the neighbourhood operators. The general conclusion was that integration of VND lead to significantly better results when compared to the original algorithms. The contribution of the study lies in the fact that it provided an in depth analysis of different VND methods and LS operators to outline which method variant works best. The previous two studies outline that it might not be enough to only include LS operators into metaheuristic algorithms,

but rather that a more sophisticated procedure needs to be integrated which controls the application of those operators.

Vlašic et al. (2019) examine the effects of initialising a population of a GA randomly or using different kinds of DRs for the problem of total weighted tardiness minimisation. Both, manually and automatically designed DRs were tested, and it was demonstrated that initialising populations with solutions obtained any kind of DRs significantly improved the performance and convergence speed of the GA. Therefore, this study outlines the importance of using initialisation techniques for GAs to significantly improve their performance, since using even a few rules to initialise some individuals of the starting population leads to improved results. In addition, it also demonstrates that existing DRs already provide good initial solutions, and that there is no need to design novel solution initialisation methods. Ulaga, Đurasević, and Jakobović (2022) focus on the same problem and investigate the application of simple LS based methods which iteratively improve the solution. The motivation of this research was to examine if simple LS based methods could perform equally well or better than more complicated metaheuristic methods. Different elements of the LS methods were analysed to determine their influence on the overall performance of the method. The algorithms were compared to other metaheuristic methods used in the literature (ACO, TS, GA, VNS) and demonstrated to achieve significant improvements over those methods. As such, this research demonstrated that with quite simple LS based approaches it is possible to obtain good results, although it is required to invest some time in their design. This further provides motivation for an adaptive method that can select LS operators that should be applied similar as Cota et al. (2017). With this it would maybe be possible to create a general method based solely on LS operators that would adapt the operators it uses based on the current problem type it is solving.

C.-L. Chen (2008) propose a hybrid metaheuristic which combines the concepts of VNS and TS for minimising the number of tardy jobs. The method generates starting solutions by three DRs and applies 4 neighbourhood operators to improve the solution. The neighbourhood operators are applied in succession by TS, meaning that the next one is used if the previous was unable to improve the solutions. If a better solution is found, then the method is restarted from a previous neighbourhood operators. The method is applied for a problem considering setup times with the objective of minimising the weighted number of tardy jobs. The obtained results demonstrate that the proposed method achieved a better performance that the basic TS method, which suggests that changing the neighbourhood operators during the search is helpful for the algorithm to achieve a better performance. C.-L. Chen (2011) consider the same problem and propose a novel metaheuristic, which is similar to the previous method but extended with additional elements, like a perturbation operator to escape local minima. In addition, several dispatching rules are used to initialise the starting solutions. The method achieves a better result than the previous TS variant proposed by the authors. As such, this

again demonstrates that combining concepts from different metaheuristics for difficult problems is beneficial.

Optimisation of makespan

Y.-K. Lin (2013) apply a PSO algorithm for makespan minimisation. The authors use a representation with partitioning symbols to denote which jobs belong to which machine. Since the representation is discrete, the authors apply a discrete PSO variant which adapts the operators in velocity calculation, and include a LS procedure to exchange jobs positions. The proposed method shows a better performance in comparison to an existing SA method. However, the SA method used by the authors for comparison was quite general as it was initially proposed for the identical parallel machines problem, and still obtained results slightly worse than the PSO. As such, it is questionable whether the effort to design a discrete PSO variant is worth it and if with additional adaptation SA could match its performance. T. Liao, Chang, Kuo, and Liao (2014) model a scheduling problem based on sequencing inbound trucks in multi-door cross docking systems, which includes includes setup times and the goal is to minimise the makespan. Five metaheuristic methods are proposed for solving the problem, which include three variants of ACO (that differ in the solution representation), a SA-TS and SA-DE hybrids. The first ACO variant uses a random key encoding, the second uses the two stage decoding procedure from J.-P. Arnaout et al. (2012), and the third uses a representation based on node clusters proposed by Keskinturk et al. (2012). The results demonstrate that the second ACO variant achieves the best results. However, if all algorithms are given the same execution time, the SA-TS hybrid method achieves the best results, since ACO required a much larger computational time. This outlines the importance of using a fair stopping criterion in comparisons, since some methods can seem to be more powerful, but they require significantly more computational time to reach those solutions than simple ones. This research is significant as it provides an investigation on different solution representation schemes for ACO, thus outlining which solution representations work best with the considered problem.

Optimisation of multiple individual objectives

Kramer and Subramanian (2017) propose a unified heuristic that can be applied for a wide class of scheduling problems that include release and setup times, and focus on the minimisation of the weighted earliness and tardiness, total weighted flowtime and total weighted tardiness. The proposed method is based on multi start ILS that, depending on the problem it solves, invokes VND with appropriate LS operators. The proposed method showed a good performance across all the considered problems and even other machine environments. The authors outline an important problem with the existing research in scheduling problems, which is that there is a wide variety of problem variants that appear in the literature and that each of them is solved by very specific heuristic or metaheuristic methods. Therefore, the authors stress

out the need for defining a unified metaheuristic method that could be used for different problem variants. A valuable extension of this work would be in the direction to perform an extensive comparison of such a general method with specific metaheuristic methods that were developed for all the different problem variants to demonstrate the gap that would exist between such a general method and the various specialised metaheuristics.

Đurasevic and Jakobovic (2016) did a study on methods for minimising four scheduling objectives, which included the makespan, total flowtime, total weighted tardiness, and weighted number of tardy jobs. They applied 4 DRs and a GA using a permutation and floating point encoding. The authors measure the time required for the GA to achieve solutions of equal quality as DRs and show that when starting from random solutions GAs require substantially more time to reach solutions of equal or better quality than DRs. Out of the two representations used by GAs, the one using real numbers achieved a better performance. Vlašic, Đurasevic, and Jakobovic (2020) extend the previous study by comparing 7 representations that are either permutation based or real number based. The representations were tested on four criteria (makespan, total flowtime, total weighted tardiness, weighted number of tardy jobs) and it was demonstrated that permutation based representations achieved the best results, with the best result being obtained by the representation that encodes the sequence of jobs and their allocation to machines is determined using a simple heuristic. However, the authors leave out a commonly used solution representation out that uses delimiters in the permutation array, and as such it is not possible to determine how that solution representation would compare to the others. But as several other studies this one also demonstrates the importance of selecting the appropriate solution representation for the problem under consideration. Unfortunately, only one problem type was investigated, and as such it would be interesting to demonstrate how general the conclusions of this study are, and if the investigated representations would achieve a similar performance also on different problems.

3.3.8 Machine based constraints

Among machine based constraints, machine eligibility and machine deterioration constraints were the most commonly investigated.

Research dealing with machine eligibility constraints

Rojanasoonthon and Bard (2005) optimise the total weighted number of tardy jobs with release times, deadlines, and sequence dependant setup times. The authors use a GRASP method, which consists of two phases. In the first phase a feasible solution is constructed by ranking all jobs at each decision point by a greedy function, based on which job is selected and scheduled. In the second phase a neighbourhood search is performed to refine the solutions obtained in the first phase. A detailed analysis of the entire algorithm is performed and experimental evaluation demonstrated it performs better than a dynamic

programming approach. However, the algorithm was only tested on small problem instances up to 20 jobs, and thus the scalability of the algorithm is not analysed. Bektur and Sarac (2019) also consider the total weighted tardiness optimisation with setup times and a common server. The idea of the server is that it performs the setup times for machines and that it can perform the setup for only a single job at a time. However, the setup of a job for a machine cannot be started until the machine is free. The authors adapt the ATC DR for such a problem, and use a TS and two SA methods (with the only difference being if the initial solution was generated by ATCS or randomly). Although the problem including a single server considered in this study is interesting and new, it was not considered further in any study. Perez-Gonzalez et al. (2019) consider a problem with setup times, machine eligibility constraints and total tardiness minimisation. The authors apply CLONALG with GRASP and VND, and compare their performance to certain heuristic methods, against which it achieved a good performance. Although the metaheuristic method is slightly more complicated due to it including other methods, the results still demonstrate that with some adaptation metaheuristics can easily outperform problem specific heuristics.

Yildirim, Duman, Krishnan, and Senniappan (2007) study a problem with setup times, and load balancing constraints for optimising the flowtime. In this study a load balancing constraint is introduced which restricts the imbalance between all machines. The authors define a structure for a simple DR and use different strategies to select the next job and machine on which it should be scheduled. The authors also propose a GA which uses the aforementioned rules to generate the initial population. The results demonstrate that GA improved significantly the solutions obtained by the proposed DRs. However, such a result is expected as DRs cannot match the performance of metaheuristics, especially if they are used to generate the initial population, the GA cannot achieve worse results. Therefore, it would have been beneficial for the authors to also apply other metaheuristic methods to demonstrate the performance of the algorithm. Joo and Kim (2015) examine a problem with setup times and the objective of minimising the total flowtime. The authors propose a GA coupled with DRs to obtain feasible solutions after genetic operators. The chromosome encodes only the sequence of jobs, and 3 DRs are used to schedule the jobs on the appropriate machines. The rules schedule the jobs based either based on their processing times, completion times, or setup times. The results demonstrate that the encoding which uses processing times for decoding achieves the best results, which is certainly surprising as using only processing times is quite myopic and could result in poor schedules in certain cases (when all jobs execute the fastest on a single machine). This study is interesting from the point of analysing how different heuristics can affect the quality of the solution during the decoding process. However, its main drawback is that the authors did not also consider a solution representation that encodes the entire solution to be used as a baseline and demonstrate how the proposed heuristic decoding scheme compares to it.

C.-Y. Cheng and Huang (2017) consider the problem of minimising the total weighted earliness and tardiness and considering dedicated machines. The authors apply a GA and extend it with a distributed release time control (RTC) mechanism which plans the job sequences and machine allocations. The goal in this research is to define the release times of machines, so that the machines start processing jobs in a way that the jobs complete as close as possible to their due dates. In that way, it is not only required to determine the best possible sequence and allocation of jobs, but also to determine the times at which each job should start with its processing. Thus, the problem is extended with an additional dimension that increases its complexity. However, such a problem variant was not further considered in subsequent studies.

D.-Y. Lin and Huang (2021) consider a scheduling problem in which additional burn-in operations are performed, and the goal is to optimise a weighted sum of makespan and violations for burn-in operations. The equipment that performs these operations is scarce and represents the bottleneck of the system. The authors consider release times, machine eligibility constraints, machine availability constraints, and setup times. The machine availability constraints are deterministic and presented in a form that each machine has a maximum number of working hours per day. A population based SA, which uses VND as the LS operator, is applied to solve the considered problem. The authors examine the proposed method on several problem cases, but provide no comparison with existing metaheuristic methods to demonstrate its effectiveness against other population based metaheuristics.

Research dealing with machine deterioration costs

Ebrahimi and Rezaeian (2015) consider a scheduling problem with makespan minimisation in which machines deteriorate over time and require that maintenance activities are scheduled on them. During a maintenance activity the machines are unavailable and cannot be used to execute any job. The authors propose the application of a GA and ICA with an extended representation that allows scheduling maintenance periods to machines. However, the authors do not perform detailed experimental comparison with other methods to demonstrate the effectiveness of their proposed approaches. The same problem is investigated by Abedi, Seidgar, and Fazlollahtabar (2017) with the addition of machine eligibility constraints and the objective of minimising the sum of the total weighted tardiness and earliness and maintenance costs. The author couple the algorithms with an artificial neural network to determine the parameter values that should be used by the metaheuristics when solving the considered problem instances. However, the results did not show evidence that such a methods can lead to significantly better results, and thus it is questionable whether such a learning method has merit.

Avalos-Rosales, Angel-Bello, Alvarez, and Cardona-Valdes (2018) consider a problem with setup times and makespan minimisation. Jobs need to be allocated in a way that they do not overlap with maintenance periods that

are fixed in the schedule. The authors propose a multi-start algorithm which consists of a solution construction phase, and two improvement phases, both of which use the same operators but are applied in different ways. Due to a lack of studies on this problems, the authors adapted their method slightly for the problem without maintenance periods and compare it to the state of the art methods used to solve that problem. Through these experiments the authors demonstrated that their proposed method works well even on a problem for which it was not designed. This is a great idea as it serves to really demonstrate that the method the authors developed performs well, due to a lack of comparable methods. M. Wang and Pan (2019) investigate the problem with setup times and machine maintenance periods where the goal is to optimise the makespan and total tardiness. The authors apply ICA which includes a multi-elite guidance strategy and several LS operators and concepts from EDA. The algorithm is also adapted for MO optimisation, and shows good performance when compared to other MO algorithms. In the considered problem the maintenance periods are known beforehand and appear at regular intervals.

Lei and Liu (2020) examine the makespan minimisation problem with fixed preventive maintenance periods. The authors propose a distributed ABC method for solving the problem, which divides the bees into several colonies that differ in the search strategies (neighbourhood search methods) that are used. However, the paper focuses more on the algorithm part than on the problem part. Ghaleb, Taghipour, and Zolfagharinia (2020) consider two problem types, fixed maintenance (the time when they occur is given), or distributing maintenance to either maximise the availability of machines or their reliability. The authors consider that machines deteriorate over time, and that at a certain point they can break down and a corrective maintenance needs to be performed. The objective is to minimise the total cost of production (due to maintenance costs and late deliveries). The authors apply SA to solve the considered problem. The main conclusion of this study is that performing maintenance activities at predefined periods of time leads to worse results than determining these periods only during the generation of the solution. This is naturally expected as such a problem variant gives more freedom and allows for better solutions to be obtained.

3.3.9 MO optimisation

The research that considers multiple objectives for the UPMSP can be roughly divided into two distinct groups. The first group consists of research which considered a multiple objectives that were combined into a single objective using a weighted linear combination of criteria and then solved using a standard algorithm for single objective optimisation. The second group consists of MO problems that are solved using optimisation methods adapted for such problems and which can obtain a Pareto front of solutions.

Linear weighed combination based methods

One of the first studies to deal with a linear combination of objectives is the study conducted by Tamaki et al. (1993). This study is also one of the first to consider the application of metaheuristics for a complicated problem variant, which included setup times, machine eligibility constraints, release times, and additional resources in the form of dies that need to be attached to machines for processing jobs. The authors minimise a linear combination of the makespan, total tardiness, and maximum tardiness. They use a binary representation of the problem and provide a method for producing feasible solutions out of it. The authors applied GA, SA, LS and concluded that based on the results SA achieved the best performance. The significance of this study lies in the fact that it is one of the seminal studies for the UPMSP, since it is one of the first to apply metaheuristics for a difficult problem in which several objectives need to be optimised simultaneously. Since at the time of publishing the study MO algorithms did not yet reach a wide attention, the application of the weighted linear combination is justified.

Jou (2005) propose a GA with sub-indexed partitioning genes, which is used to minimise the total weighted earliness and tardiness together with machine utilisation. The authors tested different weight values for the earliness and tardiness criteria, but only one for the machine utilisation. Several genetic operators were proposed and the method was compared to an existing GA to demonstrate its performance. The author applied the algorithm on a production scheduling system from a real world electronic plant. Cao, Chen, and Wan (2005) consider a problem of optimising the total weighted tardiness and machine holding costs. The holding cost is incurred when a machine is used for processing jobs, and the goal in this study is to to minimise the number of machines that are used for executing jobs. The TS method is applied with three local search operators that apply job swaps and insertions. The authors demonstrate the effectiveness of the method on smaller problem instances that were solved also using an exact approach. It would be interesting to model this problem as a real MO problem and solve with some Pareto based methods, which would generate an entire front of solutions that represent different trade offs between the total weighted tardiness criterion and the amount of machines that are used for processing.

VNS was applied by de Paula, Ravetti, Mateus, and Pardalos (2007) for a problem of minimising the sum of makespan and total weighted tardiness. The proposed algorithm uses an initial solution obtained by an adapted NEH procedure (Nawaz, Enscore, & Ham, 1983), and applies three LS operators. Three GRASP versions, each of which uses a different LS operator, with PR are also applied. The experiments demonstrate that VNS produces better results compared to GRASP, however, since each GRASP version used only a single LS operator this could potentially be limiting for the algorithm and the reason why VNS achieved better results. Ravetti, Mateus, Rocha, and Pardalos (2007) consider the same objective with setup times and propose a GRASP for solving it. The algorithm constructs the solution by assigning jobs to machines on

which they would finish the soonest, and then applies a LS. Additionally, PR is used to intensify the search and several design choices in the GRASP are evaluated. Unfortunately, the authors only compare the different GRASP variants they propose between each other, and provide no comparison with other metaheuristics. Caniyilmaz, Benli, and Ilkay (2014) consider the problem of minimising the sum of the makespan and total tardiness with setup times and machine eligibility constraints. The authors propose the application of ABC and GA with integrated LS operators, and the results demonstrate that the ABC algorithm achieved a better performance than GA.

Gao, He, and Wang (2008) consider a problem with the objective of minimising the sum of the makespan and weighed earliness and tardiness criterion, which included machine eligibility constraints. A novel GA, called PIGA, is proposed to deal with this problem, which transforms the MO problem to a single objective problem using a weighted sum of objectives. However, the weights are self-adapting during the evolution process to ensure that neither criterion starts to dominate. A LS method is also introduced in the algorithm, and the entire algorithm is additionally parallelised to improve its execution time. Although the weights in the algorithm are adaptive, the algorithm only outputs the best individual, and not a set of Pareto dominant solutions. This makes it similar to the other single objective algorithms with the addition of the weight adaptation part. Nevertheless, PIGA became commonly used as a baseline for other researchers when proposing novel MO GAs.

The same problem was considered by Gao (2010) by using AIS. The algorithm uses the same transformation scheme which was used in the previous study, and is denoted as VAIS. The authors compare the proposed algorithm with PIGA, and show that VAIS achieved a better performance. Unfortunately, in both previous studies the proposed algorithms were not compared to other MO algorithms in order to demonstrate the benefits of the transformation scheme that the authors propose. Kayvanfar and Teymourian (2014) optimise the same criteria with machine eligibility constraints and setup times, which is solved using IWDA. The algorithms is coupled with VNS and 3 LS operators, and the results show that the hybrid algorithm performs better than the standard IWDA variant. As in many other studies, the results confirm that a combination of a metaheuristic with LS operators results in both a better convergence of the algorithm, and improved final results. Van and Hop (2021) focus on the same MO problem with the addition of setup times. The authors propose a simple DR that assigns the jobs closest to their due dates. This might not be optimal and as such the authors propose that a GA is used to generate the sequence of jobs and the proposed DR their assignment to machines. However, to evaluate its performance the authors compare the GA only to simple DRs and using smaller problem set.

Mehravaran and Logendran (2011) examine a bi-criteria problem of minimising the sum of the weighted flowtime and weighted tardiness, with setup times, machine eligibility constraints, and job release times. The authors

apply TS which uses DRs to construct the initial solution. The authors provide an in depth investigation of several design choices of the algorithm, although no comparison with other metaheuristics have been performed.

Liang, dong Yang, sheng Liu, and hua Guo (2015) consider the optimisation of the sum of the total weighted tardiness and energy consumption with job release times. In the considered problem machines consume a certain amount of energy when standing idle. Therefore, machines can be turned on or off, however, turning the machine on consumes a certain amount of energy, therefore it is required to determine in which cases it is better for the machine to remain idle, and when it is better to turn it off. The authors propose the application of ACO based on the ATC rule. Solution construction is performed in three steps, first machine selection, then job selection, and in the end machine reselection. In addition, the authors also adapt a GRASP that was proposed for the single machine environment, which did not achieve as good performance as the ACO method. This study represents one of the first studies in the direction of green scheduling for the UPMSP, in which a quite detailed problem variant is used. Unfortunately, apart from Z. Li et al. (2015), no subsequent studies focused on this problem variant.

Nanthapodej, Liu, Nitisiri, and Pattanapairoj (2021b) minimise a weighted sum of the total energy cost, number of tardy jobs, and makespan. In this problem each machine has a different energy cost that is consumed during execution. A VNS method with adaptive search is applied, and several new neighbourhood strategies are proposed and tested. The method achieves better results than other state of the art methods. Nanthapodej, Liu, Nitisiri, and Pattanapairoj (2021a) focus on a similar problem. The optimisation objective is the minimisation of the weighted sum of makespan and total energy consumption. The problem includes a constraint that the execution time between all machines are not greater than a given threshold, which serves to obtain a similar load across all machines. The authors apply DE coupled with VNS that is performed after its mutation phase. VNS uses several destruction moves to remove job allocations and then applies repair moves to construct feasible schedule once again. Both works focus on energy consumption, and as such again move towards the direction of green computing.

The minimisation of the total weighted tardiness and total energy consumed is investigated by Soleimani, Ghaderi, Tsai, Zarbakhshnia, and Maleki (2020). The problem includes several properties like setup times and deteriorating job times. In this context, the real processing time is calculated based on two concepts, job deterioration and learning effects. Job deterioration leads to a longer execution time as the time from release increases, whereas learning effects lead to a lower execution times the latter the job is executed. In this study, the authors model energy consumption as a Poisson distribution, since the authors outline that the machines usually use more energy at the start until they stabilise and then the consumption is lower. The authors apply three metaheuristics, CSO, ABC, and GA, and show that CSO achieved the best overall performance. The authors also outline an interesting finding

which suggests that when considering the energy consumption criteria, not all machines are used in the best solution, which highlights that the algorithm tries to find a balance between the tardiness and energy minimisation.

An important issue of the previous studies is that the MO problem is formulated as a single objective problem. Although this has the benefit that standard methods can be used for optimisation, it inevitably also leads to certain issues. One issue is that such algorithms obtain only one solution, although if the objectives are conflicting than a single solution does not exist but rather a set of solutions that provide a trade off between different objectives. Furthermore, since all objectives are combined in a linear sum, it is required to determine the weights and thus the influence of each objective in the aggregated objective. This can also be a difficult task, since without an thorough experimental analysis it might be difficult to obtain the right values for these weights.

Pareto based MO optimisation methods

Suresh and Chaudhuri (1996) are one of the first to consider algorithms for real MO optimisation. The authors consider a bi-objective problem where the makespan and maximum tardiness need to be optimised simultaneously. The TS method is adapted for this problem by keeping a set of nondominated solutions that are obtained during execution. This study can be considered one of the first in which the authors used an adapted metaheuristic to construct a set of nondominated solutions, instead of performing a linear combination of the criteria and optimising it with a standard single objective optimisation metaheuristic. The results are compared to heuristic method which it can easily outperform.

A multi-population MO GA is proposed by Cochran, Horng, and Fowler (2003) for optimising the makespan, total weighted flowtime, and total weighted tardiness criteria. The idea of this algorithm is to define a weighted sum between those objectives and obtain initial solutions. These solutions are used to initialise the starting populations of a multi-population GA, where each population is evolved for optimising a single objective. However, the algorithm keeps a Pareto set of solutions and thus it does not provide only a single solution at the end of execution. The proposed method achieved a better performance than other MO algorithms at that time. Therefore, the authors effectively demonstrate that a strategy in which first the objectives are optimised collectively and then individually seems to be quite efficient for MO optimisation.

Chyu and Chang (2009) consider a bi-criteria problem of optimising the total weighted flowtime and total weighted tardiness with setup times. The authors apply a Pareto converging GA using two solution representations, a random key encoding and a list encoding scheme. The results show that the algorithms perform better with the real key encoding. However, the list encoding which the authors apply is very limited, as it sequences the jobs by their due dates, and mostly only works on finding the right assignment of

jobs to machines. Therefore, such an encoding significantly limits the search space, and is likely to perform worse than the real key encoding that can represent the entire solution. The authors performed a comparison with two SA MO methods, but not some other popular MO algorithms to demonstrate the performance of their proposed method. Chyu and Chang (2010) consider the same problem and apply a competitive ES memetic algorithm, SPEA2, and NSGA-II. All methods use the random key encoding, and some improvements for the methods are proposed by combining them with a weighted bipartite matching method. The proposed evolution strategy achieved better results than other competitors since it lead to a better search of the solution space.

Y.-K. Lin, Fowler, and Pfund (2013) examine several MO problems, which include the makespan, total weighted flowtime, and total weighted tardiness criteria. The authors apply a GA with permutation encoding, LS, and the idea of fitness ranking in which solutions do not dominate each other receive the same rank. The proposed algorithm is compared to some other MO algorithms, like PIGA, which it outperformed in terms of the number of nondominated solutions and their quality. S.-W. Lin and Ying (2015) expand on the problem by using a MO multi-point SA method that uses a permutation representation with partitioning symbols, and in which the initial solution is constructed by a modified ATC rule. The procedure uses several neighbourhood structures to generate new solutions and adds them to the set of non dominated solutions. The algorithm is compared to other multi-objective algorithms applied for this problem, and shows that it can achieve a better performance.

The same problem is also considered by S.-W. Lin, Ying, Wu, and Chiang (2016), where the application of an iterated Pareto greedy algorithm for MO optimisation is proposed. This algorithm keeps a list of Pareto solutions on which it performs destruction and construction operators to obtain new solutions and add them to the set of Pareto optimal solutions. This procedure is enhanced with elements from TS to restrict the possible moves in future iterations. The proposed method is quite simple and only relies on the concepts of Pareto fronts and non-dominated sorting. Unfortunately, all the previous studies do not compare the proposed methods to standard and state of the art MO methods like NSGA-II or SPEA2, which would demonstrate how standard MO perform on the considered scheduling problems, and whether there is a need to design new MO methods. Regardless of that, in each of those studies the authors performed a thorough experimental analysis and compared their methods with methods used previously for solving scheduling problems, thus outlining the improvement obtained by their proposed methods.

A MO problem of minimising makespan, total tardiness cost and machine deterioration cost is examined by Bandyopadhyay and Bhattacharya (2013). The problem also includes setup times and was solved using NSGA-II, SPEA2 and a modified NSGA-II. The modified NSGA-II uses a mutation operator that is applied on the entire population and the goal of which is to divide the jobs evenly across all the machines. This modified algorithm demonstrated to find a better distribution of Pareto optimal solutions. However, this was

only assessed visually and using the average values for each criterion, but the authors did not use MO based metrics to better examine the diversity and convergence of the Pareto fronts.

Y.-K. Lin and Lin (2015) examine a MO optimisation problem of minimising the makespan and total weighted tardiness. The authors propose a heuristic procedure that combines the NEH algorithm with ATCS, which is adapted for solving a bi-criteria problem. In addition, the authors also use TS that is extended for bi-objective optimisation by keeping a list of non-dominated solutions. They compare it to other MO GA variants and show that their method performs better, and thus presents a viable choice for the considered MO problems. This is a valuable contribution as it demonstrates that by simply combining the concept of nondominated sets with simple metaheuristics can result high quality methods than can outperform other algorithms designed especially for MO optimisation.

Rezaeian Zeidi, Zarei, and Shokoufi (2017) examine a MO problem of minimising the total completion time and the weighted earliness and tardiness criterion, which also includes setup times, release times, and machine eligibility constraints. The authors apply two MO algorithms to solve the considered problem, namely NSGA-II and CENSGA. The authors use several metrics and show that for some of them CENSGA achieves a better result. However, the experimental analysis is strange in a sense that the authors use two statistical tests to perform the comparison, the reason of which is not clear. Therefore, in the end it is difficult to state that CENSGA really performs better than NSGA-II for the considered problem.

Lei, Yuan, and Cai (2020) examine a MO optimisation problem of makespan and total tardiness minimisation. The authors consider a distributed version of the problem, in which jobs are distributed over the different factories. However, this problem is completely equal to the variant where all machines are in a single factory, as the authors do not impose any additional constraints. An improved ABC method is presented which includes problem specific LS operators, and modifications to several algorithm operators. The method is compared to several other previously proposed MO algorithms, and the authors provide visualisations of the Pareto fronts obtained by different algorithms to denote the differences between them. However, again no other state of the art MO algorithms are considered to demonstrate how general MO algorithms would perform on the considered problem.

Afzalirad and Rezaeian (2017) examine a problem of minimising the total weighted flowtime and total weighted tardiness. The problem includes job precedence constraints, machine eligibility constraints, setup and job release times. The authors propose a MO ACO algorithm which uses two pheromone trails for sequencing and scheduling of jobs similar as Keskinturk et al. (2012). The algorithm also embeds heuristic information based on job processing times for the selection of machines. The NSGA-II algorithm is also applied and compared to the proposed algorithm, which achieved better results base on several MO performance measures.

X. Wu and Che (2019) study the problem of makespan and total energy consumption minimisation. The problem assumes that machines can process jobs at different speeds, however, all machines have the same speed at a certain level. The energy consumed depends on the processing speed that is utilised during job execution. DE is applied for the considered problem. The solution representation does not encode the allocation of jobs to machines, rather this is done by a simple heuristic during evaluation. Additionally, the algorithm also uses several LS operators for swapping jobs on machines and adjusting machine speeds. Since the order of these operators can have an influence on the results, the authors apply a meta-Lamarckian learning strategy that learns and adjusts the order in which the operators should be applied. The authors demonstrate that such an adaptive strategy improves the performance of the method, however, it would have been beneficial if the authors provided a deeper analysis of the learning method. Nevertheless, the paper investigates an interesting concept which could help with the selection of appropriate LS operators.

The MO problem of minimising the total tardiness and energy consumption is studied by Pan, Lei, and Zhang (2018), in which each machine consumes a certain amount of energy per time. This is the first and unfortunately only study that deals with such a green scheduling model for the UPMSP. The authors apply ICA and add several improvements to the algorithm specific operators. However, the authors do not use a real MO representation of the problem, but rather a lexicographical ordering to rank the solutions. This means that the first objective is always used to determine the better individual, except in cases when the individuals have the same values for it and the tie is then broken by considering the second objective. As such, no real Pareto front is obtained, but rather only single solutions. Pan, Lei, and Wang (2020) consider the problem with the same objectives, however, now they extend it to true MO optimisation and do not use lexical ordering anymore. In addition, the authors use a different green scheduling problem model, in which machines have different speeds tha can be changed prior to executing jobs, which is similar to the model examined by Zheng and Wang (2018) and X. Wu and Che (2019). The authors propose a new algorithm denoted as KTPO and compare it to NSGA-II to demonstrate the better performance of their algorithm. KPTO is a two-population based algorithm which uses NSGA-II and DE in cooperation to evolve solutions. It uses two heuristics to create initial population, and also integrates two LS methods in the search process. Although both studies considered the problem variant with multiple factories, similar as Lei, Yuan, Cai, and Bai (2020) the authors disregard the placement of machines in factories. The previous two studies put more focus on the energy consumption objective, which is considered to reduce the effect of the schedule on the environment.

Conclusion

The review shows that both MO types are researched with equal intensity. Even though the methods using a weighed linear combination of criteria were considered more at the time when MO algorithms were still not as popular and the entire field was still, however, some recent studies also considered this problem variant. These studies usually combine one of the standard scheduling objectives, such as total weighted tardiness, with other criteria that focus on reducing cost or energy consumption. Especially the later variant with energy costs has been commonly considered in the are of green scheduling problems. As previously outlined, considering the optimisation of several objectives as a linear sum imposes several limitations, one of the most important being that only a single solution can be obtained and that the weights that determine the influence of each criterion need to be specified. Since standard MO problems do not have such problems, they are more easier to apply for MO problems. As such, they have been applied more often in recent years and the probability is that such a trend will continue.

3.3.10 Other problem variants

In this section studies which deal with some specific constraints or those that could not be classified in any of the previous sections are reviewed.

A scheduling problem with a common undetermined due date is analysed by Min and Cheng (2006). The objective is to find a common due date for all jobs which minimises the weighted tardiness and earliness criterion. The authors propose 4 GA variants: a standard GA, combination of a GA with SA, combination of a GA with an improvement heuristic, and a combination of a GA, SA and the improvement heuristic. The results demonstrate that all variants achieve a similar performance. However, the problem of finding common due dates was not tackled in any subsequent studies, and as such it seems that such a problem variant does not have any relevance.

Tavakkoli-Moghaddam, Taheri, Bazzazi, Izadi, and Sassani (2009) consider a scheduling problem with precedence constraints, setup times, and the objective of individually minimising the number of tardy jobs and flowtime. A GA, which performs the optimisation in two steps, is proposed. First, the number of tardy jobs objective is optimised, and in the second phase the flowtime is optimised. However, in the second phase the objective value obtained in the first phase is used as a constraint that is considered by the GA. The results demonstrate that the GA obtains results only a few percent worse than the optimal solutions. The considered problem is interesting as it shows effectiveness of considering different criteria in a hierarchical way. Unfortunately, the authors do not make a comparison with pure MO algorithms to show how such an approach would compare to standard MO optimisation, and which benefits it can offer.

A problem with precedence constraints and the total tardiness minimisation is considered by Liu (2013). The authors first propose a simple DR based heuristic used to solve the problem. After that, the authors propose

a hybrid GA with two subpopulations, one initialised with the DR based heuristic, and the second one initialised randomly. On both these populations the standard genetic operators are applied. The results demonstrate that the GA with the population initialised by the DR performs better, however, only for problems with tighter due dates, which is expected since if the due dates are not tight then the problem becomes easier to solve and even a simpler algorithm will perform well. This research shows that generating good solutions for the initial population using DRs can significantly improve the performance of the GA, which is further researched later on by (Vlašic et al., 2019).

Hassan Abdel-Jabbar, Kacem, and Martin (2014) also consider precedence constraints but for a problem with makespan minimisation. The authors apply a GA and compare it to a simple DR which schedules the job that can be completed the soonest, and show that the GA achieves a better performance. However, this is expected as simple DRs cannot match the performance of metaheuristic methods, therefore the contribution of this study is minor. Afzalirad and Rezaeian (2016a) consider the problem of minimising the total late work setup times, precedence constraints, job release times, and machine eligibility restrictions. This criterion is similar to the total tardiness, except that in this case if a job starts after its due date, it receives a constant penalty, which does not depend on when it started. If only a part of the job is executed after the due date, then it is calculated in the same way as for tardiness. Although the authors motivate this problem by a real world agricultural problem, it is not completely clear as to why the total late work is considered rather than the total weighted tardiness. The authors propose a hybrid GA that uses the acceptance strategy of SA, so that children can advance to the next generation even if they are not better than their parents. The objective considered in this paper was, however, not considered in any future studies.

Salehi Mir and Rezaeian (2016) analyse a problem with deteriorating job effects, where processing times of jobs increases after they are released into the system. The authors optimise the total flowtime using a GA, PSO, and a hybrid method that combines the previous two metaheuristics. The hybrid metaheuristic uses the crossover and mutation operators, but also upon finding a better solution it tries to move all the solutions to that solution similar as in PSO. The hybrid method achieved a better performance than any of the individual methods. However, the GA algorithm did not use any LS method embedded in it, and as such it would be interesting to see whether including additional domain knowledge could help the GA to match the hybrid method.

Bilyk and Mönch (2010) consider a problem adapted from a printed wiring board manufacturing environment with the objective of optimising the total weighted tardiness. In it jobs need to be transported into a storage space by a vehicle with a given maximum capacity. Also, multiple scheduling periods are considered. The authors apply VNS for the considered problem with 8 local search operators and an initial solution constructed by the ATC rule. The extensions of the problem are interesting in the sense that the authors also consider the transportation of jobs to machines, which was not considered in

any other study. Unfortunately, no further examinations on this problem have been made.

J.-F. Chen (2013) consider the total weighted flowtime minimisation, in which a certain number of jobs have fixed deadlines which must not be violated, and therefore this group of jobs needs to be prioritised. The authors apply a RRT metaheuristic and a random descent search coupled with a procedure that ensures that the constructed schedules are feasible. Although the RRT metaheuristic achieved a good performance, it has been rarely used for solving the UPMSP, as the method is quite exotic. Bhardwaj, Gajpal, Surti, and Gill (2020) consider a problem from the cloud computing domain, in which each job consists out of several tasks which need to be executed in a sequential manner. Two independent optimisation objectives are considered, makespan and total completion time minimisation. The authors propose an ACO method coupled with LS operators that is compared with several existing DRs. Naturally, ACO achieved a significantly better performance than any of the individual DRs, however, such a result is expected and unfortunately does not demonstrate the effectiveness of the proposed method.

Rambod and Rezaeian (2014) apply GA and ABC for makespan minimisation with rework processes, setup times, and machine eligibility constraints. A solution representation, which encodes the expected rework processes (that are considered finite) and schedules them accordingly, is used. Since the rework processes are probabilistic, the schedule is then rebuilt if a certain job does not require additional rework processes. The results demonstrate that the ABC algorithms outperformed the GA. However, since rework processes are a dynamic element in scheduling, it would have been interesting to see how the algorithms would perform under different problem conditions and compared to DRs that perform the decisions dynamically. X. Wang, Li, Chen, and Mao (2020) examine the optimisation of the total weighted tardiness with rework processes. The authors propose that the problem, which is stochastic in a way that it is not known which tasks will have to be reworked, is transformed into several deterministic problems using task estimation. These models are then solved either by using metaheuristics (SA and GA), or a simple DR. The results demonstrate that the proposed task estimation procedure is more efficient than the existing ones (used by Rambod and Rezaeian (2014)), and that the methods proposed based on it achieve a better performance than standard DRs used for that problem. As such, this study has proposed an effective alternative to deal with scheduling problems that include rework processes.

Arık (2019) consider the problem of minimising the weighted tardiness and earliness criterion with a common due date. The authors tested three metaheuristics, ABC, GA, and SA to examine which kind of metaheuristic type (swarm intelligence, evolutionary algorithm, single solution algorithm) achieves the best results. The algorithms did not include any problem specific modifications, and the results demonstrated that ABC achieved the best performance. This study is interesting from the point that only the basic

algorithms are considered, since the authors wanted to asses the suitability of such algorithms. However, more metaheuristics should have been included to determine the best one, since the sample of only three selected ones is not big enough to give a definitive conclusion. Therefore, this study could serve as a motivation to widen the examination of different basic metaheuristic algorithms to determine which might be the most suitable for solving various UPMSPs.

4 Classification of research

In this section all the studies will be roughly classified based on three categories to obtain a better notion of the research which has been performed in each area. The classification is carried out by grouping the research based on the optimised criteria, existence of additional constraints, and the solution methods used for solving the problem. Naturally, if several constraints, criteria or solution methods are used in a single study, then the study is enumerated in each group it belongs to.

Table 2 shows the classification of the research based on the optimised criteria, with Figure 4 denoting the percentage of research that consider each individual criterion. In around 60% of studies the makespan criterion is optimised, which makes it the most investigated objective for the UPMSP. The dominance of this objective is not unexpected, as reducing the total duration of the schedule has always been of importance. The second most popular optimisation objective is the total (weighted) tardiness, which is optimised in around 26% of the studies. This objective is important in cases when due dates are defined for jobs, and in such scenarios this objective is almost always considered. Other due date related objectives like the number of tardy jobs or maximum tardiness have received significantly less attention. Other criteria that are considered in a larger number of studies are the total (weighted) flowtime with share of around 19%, and total weighted earliness and tardiness with a share of around 8%. The total weighted earliness and tardiness objective is important in problems where the jobs need to be completed as close to their due dates as possible, like in cases where the jobs represent goods that can get spoiled. On the other hand the flowtime related criteria are directed towards reducing the amount of time that the jobs spend in the system waiting to be executed. All other objectives were rarely considered, with some very specialised ones being examined in only one or two studies, like the total late time or number of jobs completed just in time. As can be seen, most studies focus on optimising standard scheduling criteria. Two non-standard objectives which received more attention are production cost (5%) and total energy consumption (4%). The production cost objective was quite popular because in several studies it was important to minimise a certain cost incurred by using additional resources. On the other hand total energy consumption became more popular in recent years with the aim to improve energy efficiency as green manufacturing has been gaining more importance in the industry. As

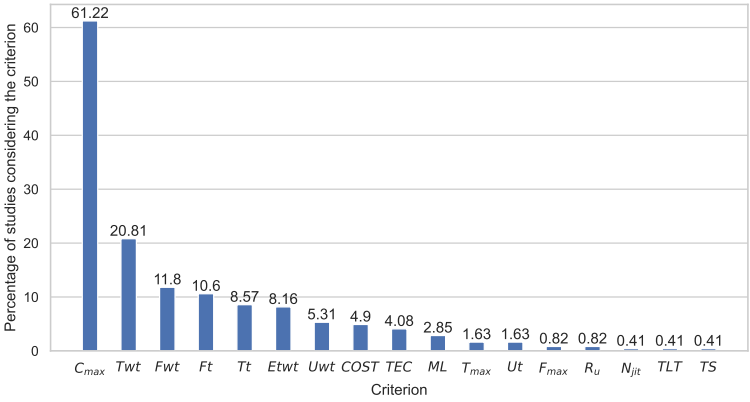

Fig. 4 Distribution of research dealing with the considered criteria

more studies focus both on problems which include additional resources as well as on the goal of minimising the total energy consumption, it is likely that the consideration of both objectives will increase in future studies.

The table also includes entries denoted as WS and MO in which several objectives are optimised simultaneously. WS outlines those studies which optimise several criteria using a weighted sum of objectives and thus focus on optimising a single objective value. Such an approach usually has drawbacks, since the weights of each objective need to be fine-tuned to obtain a solution that provides a good balance between all optimised objectives. On the other hand, MO denotes studies which deal with optimisation of several objectives simultaneously using specialised algorithms. These algorithms are adapted so that they do not obtain only a single solution, but rather a set of solutions that provide different trade-offs between the optimised objectives. Each problem variant was investigated in around 10% of studies. In most cases two objectives were optimised simultaneously, although some studies considered 3 or more objectives. Even though the share of studies which deal with MO optimisation is still quite small, around half of them were published during the last several years. This shows that MO problems are becoming more studied in the UPMSP and that such a trend could continue in the future as MO is gaining more importance and new advances are being made in this area. This will be especially true for problems considering green scheduling and minimising costs since these two criteria are conflicting with all other scheduling criteria, and as such are always optimised in combination with them.

Table 2: Research classification based on the optimised criteria

Criterion	References
C_{max}	Ibarra and Kim (1977), De and Morton (1980), Hariri and Potts (1991), Glass et al. (1994), Piersma and van Dijk (1996), Suresh and Chaudhuri (1996), SURESH and GHAUDHURI (1996), Herrmann et al. (1997), Srivastava (1998), Maheswaran et al. (1999), T. Braun et al. (1999), T.D. Braun et al. (2001), M.-Y. Wu and Shu (2001), Dhaenens-Flipo (2001), Al-Salem and Armacost (2002), Anagnostopoulos and Rabadi (2002), Cochran et al. (2003), Ritchie and Levine (2003), Peng and Liu (2004), J.-F. Chen (2004), Du Kim and Kim (2004), Gao (2005), Helal et al. (2006), Rabadi et al. (2006), GUO et al. (2007), de Paula et al. (2007), S. Xu and Bean (2007), Xhafa et al. (2007), Luo et al. (2007), Ravetti et al. (2007), Ravetti et al. (2007), J.-P. Arnaout et al. (2008), Gao et al. (2008), Munir et al. (2008), Celano et al. (2008), Tseng et al. (2009), Dolgui et al. (2009), Izakian et al. (2009), J.-P. Arnaout et al. (2009), Charalambous et al. (2010), Fanjul-Peyro and Ruiz (2010), Gao (2010), Ying et al. (2010), Balin (2011), P.-C. Chang and Chen (2011), Fanjul-Peyro and Ruiz (2011), Y. Lin et al. (2011), Vallada and Ruiz (2011), Niu et al. (2011), Fleszar et al. (2011), Liu and Yang (2011), Haddad et al. (2012), Fanjul-Peyro and Ruiz (2012), I.-L. Wang et al. (2012), Ramezanian and Saidi-Mehrabad (2012), Rafsanjani and Bardsiri (2012), Briceño et al. (2012), Keskinturk et al. (2012), Cappadonna. et al. (2012), J.-P. Arnaout et al. (2012), Cappadonna et al. (2013), Y.-K. Lin (2013), Costa et al. (2013), Yang-Kuei and Chi-Wei (2013), X. Li et al. (2013), Y.-K. Lin et al. (2013), W.-L. Wang et al. (2013), Low et al. (2013), Torabi et al. (2013), Bandyopadhyay and Bhattacharya (2013), R.O. Diana et al. (2013), Cota et al. (2014), Avalos-Rosales et al. (2014), Caniyilmaz et al. (2014), Kayvanfar and Teymourian (2014), S.-W. Lin and Ying (2014), Eroglu et al. (2014), T. Liao et al. (2014), Rambod and Rezaeian (2014), e Santos and Madureira (2014), Nohra Haddad et al. (2014), Hassan Abdel-Jabbar et al. (2014), R.O.M. Diana et al. (2015), Sels et al. (2015), X. Xu et al. (2015), S.-W. Lin and Ying (2015), Afzalirad and Shafipour (2015), Y.-K. Lin and Lin (2015), Ebrahimi and Rezaeian (2015), S.-W. Lin et al. (2016), C.-J. Liao et al. (2016), Afzalirad and Rezaeian (2016b), L. Wang et al. (2016), Özpeynirci et al. (2016), long Zheng and Wang (2016), Santos et al. (2016), Low and Wu (2016), Đurasevic and Jakobovic (2016), Đurasevic et al. (2016), Đurasević and Jakobović (2017b), Đurasević and Jakobović (2017a), Joo and Kim (2017), Arroyo and Leung (2017a), Shahidi-Zadeh et al. (2017), Shahvari and Logendran (2017a), MANUPATI et al. (2017), Cota et al. (2017), R.O.M. Diana et al. (2017), Arroyo and Leung (2017b), Fanjul-Peyro et al. (2017), Villa et al. (2018), Đurasevic and Jakobovic (2018), Zheng and Wang (2018), S. Lu et al. (2018), Ezugwu et al. (2018), Avalos-Rosales et al. (2018), S. Zhou et al. (2018), Tozzo et al. (2018), Ezugwu and Akutsah (2018), Ezugwu (2019), X. Wu and Che (2019), Vallada et al. (2019), Đurasević and Jakobović (2019), M. Wang and Pan (2019), Al-harkan and Qamhan (2019), Jouhari et al. (2019), J.-P. Arnaout (2019), Yepes-Borrero et al. (2020), Terzi et al. (2020), Lei, Yuan, and Cai (2020), Lei, Yuan, Cai, and Bai (2020), de Abreu and de Athayde Prata (2020), Vlašic et al. (2020), Jouhari et al. (2020), Lei and Liu (2020), Orts et al. (2020), Bhardwaj et al. (2020), Yepes-Borrero et al. (2021), Van and Hop (2021), Ewees et al. (2021), D.-Y. Lin and Huang (2021), Al-harkan et al. (2021), Al-qaness et al. (2021), Nanthapodej et al. (2021b), Nanthapodej et al. (2021a), Jovanovic and Voß (2021), C.-Y. Cheng et al. (2021), Zarook, Yaser et al. (2021), L. Zhang et al. (2021), Planinić et al. (2022)

$Etwt$	Bank and Werner (2001), Jou (2005), Min and Cheng (2006), Raja et al. (2008), Gao et al. (2008), Gao (2010), Kayvanfar and Teymourian (2014), de C. M. Nogueira et al. (2014), Polyakovskiy and M'Hallah (2014), Zeidi and MohammadHosseini (2015), Rezaeian Zeidi et al. (2017), Shahidi-Zadeh et al. (2017), Abedi et al. (2017), C.-Y. Cheng and Huang (2017), Kramer and Subramanian (2017), Đurasević and Jakobović (2017b), Đurasevic and Jakobovic (2018), Arık (2019), Van and Hop (2021)
Ft	RANDHAWA and SMITH (1995), Randhawa and Kuo (1997), Yildirim et al. (2007), Xhafa et al. (2007), Tavakkoli-Moghaddam et al. (2009), Izakian et al. (2009), Ruiz and Andrés-Romano (2011), Rafsanjani and Bardsiri (2012), Siepak and Józefczyk (2014), C.-H. Lee et al. (2014), Joo and Kim (2015), Salehi Mir and Rezaeian (2016), Đurasevic and Jakobovic (2016), Strohhecker et al. (2016), Đurasevic et al. (2016), Đurasević and Jakobović (2017b), Đurasević and Jakobović (2017a), Rezaeian Zeidi et al. (2017), MANUPATI et al. (2017), Đurasevic and Jakobovic (2018), Arroyo et al. (2019), Đurasević and Jakobović (2019), Vlašic et al. (2020), M.-Z. Wang et al. (2020), Bhardwaj et al. (2020), Planinić et al. (2022)
Fwt	Weng et al. (2001), Vredeveld and Hurkens (2002), Cochran et al. (2003), J.-P. Arnaout and Rabadi (2005), J.-P.M. Arnaout et al. (2006), Cruz-Chavez et al. (2009), Chyu and Chang (2009), Chyu and Chang (2010), Y. Lin et al. (2011), Mehravaran and Logendran (2011), Bozorgirad and Logendran (2012), Rodriguez, García-Martínez, et al. (2012), Rodriguez, Blum, et al. (2012), Y.-K. Lin et al. (2013), Torabi et al. (2013), Rodriguez et al. (2013), J.-F. Chen (2013), Yang-Kuei and Chi-Wei (2013), Bitar et al. (2014), Y.-C. Chang et al. (2014), S.-W. Lin and Ying (2015), Shahvari and Logendran (2015), S.-W. Lin et al. (2016), Đurasević and Jakobović (2017b), Afzalirad and Rezaeian (2017), Shahvari and Logendran (2017b), Kramer and Subramanian (2017), Đurasevic and Jakobovic (2018), H. Wang and Alidaee (2019)
F_{max}	Đurasević and Jakobović (2017b), Đurasevic and Jakobovic (2018)
ML	RANDHAWA and SMITH (1995), Peng and Liu (2004), Jou (2005), Xhafa et al. (2007), MANUPATI et al. (2017), Đurasević and Jakobović (2017b), Đurasevic and Jakobovic (2018)
Ut	RANDHAWA and SMITH (1995), Randhawa and Kuo (1997), Golconda et al. (2004), Silva and Magalhaes (2006), Tavakkoli-Moghaddam et al. (2009), I.-L. Wang et al. (2012)
Uwt	Rojanasoonthon and Bard (2005), C.-L. Chen and Chen (2008), C.-L. Chen (2008), C.-L. Chen (2011), Đurasevic and Jakobovic (2016), Đurasevic et al. (2016), Đurasević and Jakobović (2017b), Đurasević and Jakobović (2017a), Đurasevic and Jakobovic (2018), Đurasević and Jakobović (2019), Vlašic et al. (2020) Nanthapodej et al. (2021b), Planinić et al. (2022)
TT	Randhawa and Kuo (1997), D.-W. Kim et al. (2002), J.-F. Chen and Wu (2006), S.-I. Kim et al. (2006), S.-I. Kim et al. (2007), Logendran et al. (2007), Celano et al. (2008), J.-F. Chen (2009), S.-W. Lin et al. (2010), Ying and Lin (2012), Liu (2013), J.-H. Lee et al. (2013), Caniyilmaz et al. (2014), Z. Li et al. (2015), MANUPATI et al. (2017), Pan et al. (2018), M. Wang and Pan (2019), Perez-Gonzalez et al. (2019), Lei, Yuan, and Cai (2020), Pinheiro et al. (2020), Pan et al. (2020)

TWT	D.-W. Kim et al. (2003), Cochran et al. (2003), LOGENDRAN and SUBUR (2004), Cao et al. (2005), Na et al. (2006), Z. Zhang et al. (2006), de Paula et al. (2007), H. Zhou et al. (2007), Klemmt et al. (2009), Tseng et al. (2009), Chyu and Chang (2009), Bilyk and Mönch (2010), Chyu and Chang (2010), Y. Lin et al. (2011), Mehravaran and Logendran (2011), Bozorgirad and Logendran (2012), C.-W. Lin et al. (2013), Y.-K. Lin et al. (2013), Torabi et al. (2013), Bandyopadhyay and Bhattacharya (2013), Yang-Kuei and Chi-Wei (2013), Y.-K. Lin and Hsieh (2014), Liang et al. (2015), S.-W. Lin and Ying (2015), Shahvari and Logendran (2015), Y.-K. Lin and Lin (2015), S.-W. Lin et al. (2016), Đurasevic and Jakobovic (2016), Đurasevic et al. (2016), Đurasević and Jakobović (2017b), Đurasević and Jakobović (2017a), Afzalirad and Rezaeian (2017), Shahvari and Logendran (2017b), Kramer and Subramanian (2017), R.O. Diana et al. (2018), Đurasevic and Jakobovic (2018), Bektur and Sarac (2019), Vlašic et al. (2019), Đurasević and Jakobović (2019), Đurasevic and Jakobovic (2020), Đurasević and Jakobović (2020), Soleimani et al. (2020), Vlašic et al. (2020), Marinho Diana and de Souza (2020), X. Wang et al. (2020), Jaklinovic et al. (2021), Planinić, Đurasević, and Jakobović (2021), Planinić, Đurasević, and Jakobović (2021), Đurasević and Jakobović (2022), Ulaga et al. (2022), Planinić et al. (2022)
T_{max}	Suresh and Chaudhuri (1994), Suresh and Chaudhuri (1996), Peng and Liu (2004), J.-F. Chen (2006), Đurasević and Jakobović (2017b), Đurasevic and Jakobovic (2018)
$COST$	Dhaenens-Flipo (2001), Cao et al. (2005), Tseng et al. (2009), Bandyopadhyay and Bhattacharya (2013), Z. Li et al. (2015), Shahidi-Zadeh et al. (2017), Shahvari and Logendran (2017a), Abedi et al. (2017), H. Lu and Qiao (2017), Ghaleb et al. (2020), M.-Z. Wang et al. (2020), D.-Y. Lin and Huang (2021)
N_{jit}	Jolai et al. (2009)
R_u	Ruiz and Andrés-Romano (2011), Yepes-Borrero et al. (2021)
TLT	Afzalirad and Rezaeian (2016a)
TEC	Liang et al. (2015), Che et al. (2017), Zheng and Wang (2018), Pan et al. (2018), X. Wu and Che (2019), Soleimani et al. (2020), Pan et al. (2020), Nanthapodej et al. (2021b), Nanthapodej et al. (2021a), L. Zhang et al. (2021)
Ts	Strohhecker et al. (2016)
WS	Dhaenens-Flipo (2001), Jou (2005), Cao et al. (2005), de Paula et al. (2007), Ravetti et al. (2007), Gao et al. (2008), Gao (2010), Mehravaran and Logendran (2011), Ruiz and Andrés-Romano (2011), I.-L. Wang et al. (2012), Bozorgirad and Logendran (2012), Caniyilmaz et al. (2014), Kayvanfar and Teymourian (2014), Z. Li et al. (2015), Liang et al. (2015), Shahvari and Logendran (2015), Shahvari and Logendran (2017b), Shahidi-Zadeh et al. (2017), Abedi et al. (2017), Soleimani et al. (2020), Van and Hop (2021), D.-Y. Lin and Huang (2021), Nanthapodej et al. (2021b), Nanthapodej et al. (2021a)
MO	Suresh and Chaudhuri (1996), Randhawa and Kuo (1997), Cochran et al. (2003), Chyu and Chang (2009), Chyu and Chang (2010), Y.-K. Lin et al. (2013), Torabi et al. (2013), Bandyopadhyay and Bhattacharya (2013), S.-W. Lin and Ying (2015), Y.-K. Lin and Lin (2015), S.-W. Lin et al. (2016), Afzalirad and Rezaeian (2017), Rezaeian Zeidi et al. (2017), Shahvari and Logendran (2017a), MANUPATI et al. (2017), Đurasević and Jakobović (2017b), Zheng and Wang (2018), Tozzo et al. (2018), Pan et al. (2018), X. Wu and Che (2019), M. Wang and Pan (2019), Lei, Yuan, and Cai (2020), Pan et al. (2020), M.-Z. Wang et al. (2020), Yepes-Borrero et al. (2021), L. Zhang et al. (2021)

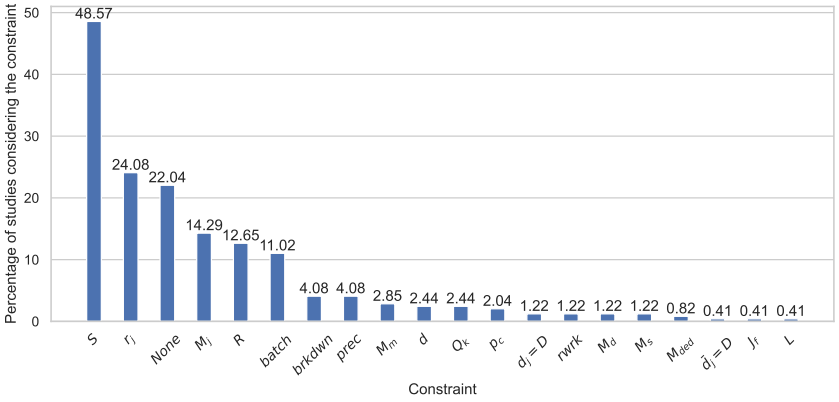

Fig. 5 Distribution of research dealing with the considered constraints

Table 3 shows the additional constraints that have been considered in the reviewed studies, while Figure 5 denotes the percentage by which each constraint was considered in the reviewed research. Most research has focused on problems that include setup times, with a share of almost 50% of the studies. Since setup times usually have a direct effect on the performance of the scheduling system Allahverdi (2016), it is good that most research in the UPMSP take them into account. However, setup times are quite simple constraints compared to some others since they do not have an effect the feasibility of the schedule, and therefore they can be easily taken into account by different methods, especially metaheuristics. The second most commonly considered constraint is job release times with a share of around 24%. Release times are especially common in cases when dynamic scheduling problems are considered (Đurasevic et al., 2016), in which it is not known when jobs are released into the system and as such it is not possible to create the schedule in advance. Some other more frequently considered objectives are machine eligibility, limited resources, and batch scheduling, which are considered in between 10% and 14% of the studies. Other constraints are only sparsely applied, although this does not mean that they are less important. Some constraints like dedicated machines or job families have been considered in only one or two studies, which seems to demonstrate that such constraints are not significant in the context of the UPMSP. Naturally, it should be outlined that the problem types are more numerous than denoted with this classification, since some constraints have different types. For example, the setup times can depend on various things (previous jobs, next job, machine, assigned worker), or resource constraints can be of different types (renewable or nonrenewable), characteristics, or quantities. Therefore, the number of different problem variants is even larger.

It is interesting to observe that around 23% of research focused on the most basic problem without considering any additional constraints. However,

a great deal of such research was done in the earlier period and recent studies almost always include at least one additional constraint. More specifically, 43% of studies consider problems with exactly one constraint. Not only that, but around 35% of studies deal with problems that include several constraints. This is especially evident in studies that model a real-world problem as an UPMSP. However, as the focus of studies shifts to more complex problems, the number of studies dealing with multiple constraints will probably increase in the next years. In addition, it is also likely that new constraints or constraint variants will be proposed and examined to model problems from the real world.

Table 3: Research classification based on additional properties

Constraints	References
None	Ibarra and Kim (1977), De and Morton (1980), Hariri and Potts (1991), Suresh and Chaudhuri (1994), Glass et al. (1994), Piersma and van Dijk (1996), Suresh and Chaudhuri (1996), Srivastava (1998), Maheswaran et al. (1999), T. Braun et al. (1999), T.D. Braun et al. (2001), M.-Y. Wu and Shu (2001), Vredeveld and Hurkens (2002), Ritchie and Levine (2003), Cochran et al. (2003), Peng and Liu (2004), Golconda et al. (2004), Cao et al. (2005), Gao (2005), Xhafa et al. (2007), Luo et al. (2007), GUO et al. (2007), H. Zhou et al. (2007), Munir et al. (2008), Izakian et al. (2009), Cruz-Chavez et al. (2009), Jolai et al. (2009), Charalambous et al. (2010), Fanjul-Peyro and Ruiz (2010), Balin (2011), Fanjul-Peyro and Ruiz (2011), Y. Lin et al. (2011), Rafsanjani and Bardsiri (2012), Briceño et al. (2012), Fanjul-Peyro and Ruiz (2012), Rodriguez, García-Martínez, et al. (2012), Rodriguez, Blum, et al. (2012), C.-W. Lin et al. (2013), Y.-K. Lin et al. (2013), Rodriguez et al. (2013), e Santos and Madureira (2014), Y.-C. Chang et al. (2014), Siepak and Józefczyk (2014), Polyakovskiy and M'Hallah (2014), S.-W. Lin and Ying (2015), Sels et al. (2015), S.-W. Lin et al. (2016), Kramer and Subramanian (2017), Che et al. (2017), Pan et al. (2018), H. Wang and Alidaee (2019), Lei, Yuan, Cai, and Bai (2020), Lei, Yuan, and Cai (2020), Orts et al. (2020)
$\bar{d}$	Rojanasoonthon and Bard (2005), J.-F. Chen (2009), S.-W. Lin et al. (2010), Ying and Lin (2012), J.-H. Lee et al. (2013), J.-F. Chen (2013)
$d_j = D$	Bank and Werner (2001), Min and Cheng (2006), Arık (2019)
$\bar{d}_j = D$	C.-H. Lee et al. (2014)
M_j	Randhawa and Kuo (1997), Al-Salem and Armacost (2002), LOGENDRAN and SUBUR (2004), Rojanasoonthon and Bard (2005), J.-F. Chen and Wu (2006), J.-F. Chen (2006), Z. Zhang et al. (2006), Silva and Magalhaes (2006), Yildirim et al. (2007), Gao et al. (2008), Dolgui et al. (2009), Gao (2010), Bozorgirad and Logendran (2012), I.-L. Wang et al. (2012), W.-L. Wang et al. (2013), Low et al. (2013), Caniyilmaz et al. (2014), Kayvanfar and Teymourian (2014), Rambod and Rezaeian (2014), Joo and Kim (2015), Shahvari and Logendran (2015), Afzalirad and Rezaeian (2016a), Afzalirad and Rezaeian (2016b), Low and Wu (2016), Afzalirad and Rezaeian (2017), Rezaeian Zeidi et al. (2017), Shahvari and Logendran (2017b), Shahvari and Logendran (2017a), Abedi et al. (2017), Bektur and Sarac (2019), Perez-Gonzalez et al. (2019), D.-Y. Lin and Huang (2021), Nanthapodej et al. (2021a), Nanthapodej et al. (2021b), Jaklinovic et al. (2021)
M_{ded}	C.-H. Lee et al. (2014), C.-Y. Cheng and Huang (2017)

s	RANDHAWA and SMITH (1995), Randhawa and Kuo (1997), Weng et al. (2001), Dhaenens-Flipo (2001), D.-W. Kim et al. (2002), Anagnostopoulos and Rabadi (2002), D.-W. Kim et al. (2003), J.-F. Chen (2004), Jou (2005), Rojanasoonthon and Bard (2005), J.-P. Arnaout and Rabadi (2005), J.-P.M. Arnaout et al. (2006), Na et al. (2006), Z. Zhang et al. (2006), J.-F. Chen and Wu (2006), J.-F. Chen (2006), Helal et al. (2006), Silva and Magalhaes (2006), Rabadi et al. (2006), S.-I. Kim et al. (2006), de Paula et al. (2007), Ravetti et al. (2007), S.-I. Kim et al. (2007), Yildirim et al. (2007), C.-L. Chen (2008), J.-P. Arnaout et al. (2008), Raja et al. (2008), C.-L. Chen and Chen (2008), Celano et al. (2008), Tseng et al. (2009), J.-F. Chen (2009), J.-P. Arnaout et al. (2009), Tavakkoli-Moghaddam et al. (2009), Chyu and Chang (2009), Dolgui et al. (2009), Chyu and Chang (2010), Ying et al. (2010), S.-W. Lin et al. (2010), P.-C. Chang and Chen (2011), Vallada and Ruiz (2011), Mehravaran and Logendran (2011), Niu et al. (2011), C.-L. Chen (2011), Fleszar et al. (2011), Ruiz and Andrés-Romano (2011), Haddad et al. (2012), I.-L. Wang et al. (2012), J.-P. Arnaout et al. (2012), Cappadonna. et al. (2012), Bozorgirad and Logendran (2012), Keskinturk et al. (2012), R.O. Diana et al. (2013), Cappadonna et al. (2013), Costa et al. (2013), Torabi et al. (2013), Bandyopadhyay and Bhattacharya (2013), J.-H. Lee et al. (2013), J.-F. Chen (2013), Cota et al. (2014), Bitar et al. (2014), Y.-K. Lin and Hsieh (2014), Avalos-Rosales et al. (2014), Caniyilmaz et al. (2014), Kayvanfar and Teymourian (2014), C.-H. Lee et al. (2014), S.-W. Lin and Ying (2014), Eroglu et al. (2014), T. Liao et al. (2014), Rambod and Rezaeian (2014), de C. M. Nogueira et al. (2014), Nohra Haddad et al. (2014), R.O.M. Diana et al. (2015), Zeidi and MohammadHosseini (2015), Joo and Kim (2015), Shahvari and Logendran (2015), C.-J. Liao et al. (2016), Afzalirad and Rezaeian (2016a), Salehi Mir and Rezaeian (2016), Afzalirad and Rezaeian (2016b), L. Wang et al. (2016), Santos et al. (2016), Strohhecker et al. (2016), Joo and Kim (2017), Afzalirad and Rezaeian (2017), Rezaeian Zeidi et al. (2017), Shahvari and Logendran (2017b), Shahvari and Logendran (2017a), MANUPATI et al. (2017), Cota et al. (2017), R.O.M. Diana et al. (2017), Kramer and Subramanian (2017), Tozzo et al. (2018), Ezugwu et al. (2018), Avalos-Rosales et al. (2018), R.O. Diana et al. (2018), Ezugwu and Akutsah (2018), M. Wang and Pan (2019), Al-harkan and Qamhan (2019), Perez-Gonzalez et al. (2019), Bektur and Sarac (2019), Jouhari et al. (2019), J.-P. Arnaout (2019), Ezugwu (2019), Yepes-Borrero et al. (2020), de Abreu and de Athayde Prata (2020), Terzi et al. (2020), Marinho Diana and de Souza (2020), Soleimani et al. (2020), Jouhari et al. (2020), Yepes-Borrero et al. (2021), Van and Hop (2021), Ewees et al. (2021), D.-Y. Lin and Huang (2021), Al-harkan et al. (2021), Al-qaness et al. (2021), Jovanovic and Voß (2021), C.-Y. Cheng et al. (2021), Jaklinovic et al. (2021), L. Zhang et al. (2021)
brkdwn	SURESH and GHAUDHURI (1996), LOGENDRAN and SUBUR (2004), Logendran et al. (2007), Bozorgirad and Logendran (2012), Shahvari and Logendran (2015), Shahvari and Logendran (2017b), Shahvari and Logendran (2017a), Ghaleb et al. (2020), Jaklinovic et al. (2021), D.-Y. Lin and Huang (2021)
prec	Herrmann et al. (1997), Tavakkoli-Moghaddam et al. (2009), Liu and Yang (2011), Liu (2013), Hassan Abdel-Jabbar et al. (2014), Afzalirad and Rezaeian (2016a), Afzalirad and Rezaeian (2016b), Afzalirad and Rezaeian (2017), Bhardwaj et al. (2020), Jaklinovic et al. (2021)

r_j	Bank and Werner (2001), Du Kim and Kim (2004), LOGENDRAN and SUBUR (2004), Rojanasoonthon and Bard (2005), S.-I. Kim et al. (2006), S.-I. Kim et al. (2007), Logendran et al. (2007) Klemmt et al. (2009), Tavakkoli-Moghaddam et al. (2009), Bilyk and Mönch (2010), C.-L. Chen (2011), Bozorgirad and Logendran (2012), Ramezanian and Saidi-Mehrabad (2012), I.-L. Wang et al. (2012), Y.-K. Lin (2013), Yang-Kuei and Chi-Wei (2013), Torabi et al. (2013), T. Liao et al. (2014), Rambod and Rezaeian (2014), Y.-K. Lin and Hsieh (2014), Z. Li et al. (2015), Liang et al. (2015), Shahvari and Logendran (2015), Y.-K. Lin and Lin (2015), Afzalirad and Rezaeian (2016a), Salehi Mir and Rezaeian (2016), Đurasevic and Jakobovic (2016), Afzalirad and Rezaeian (2016b), Đurasevic et al. (2016), Arroyo and Leung (2017a), Afzalirad and Rezaeian (2017), Rezaeian Zeidi et al. (2017), Shahvari and Logendran (2017b), Shahidi-Zadeh et al. (2017), Shahvari and Logendran (2017a), MANUPATI et al. (2017), Kramer and Subramanian (2017), Đurasević and Jakobović (2017b), Đurasević and Jakobović (2017a), Arroyo and Leung (2017b), R.O. Diana et al. (2018), Đurasevic and Jakobovic (2018), Al-harkan and Qamhan (2019), Arroyo et al. (2019), Vlašic et al. (2019), Đurasević and Jakobović (2019), Đurasevic and Jakobovic (2020), Đurasević and Jakobović (2020), Vlašic et al. (2020), Marinho Diana and de Souza (2020), D.-Y. Lin and Huang (2021), Jaklinovic et al. (2021), Zarook, Yaser et al. (2021), Al-harkan et al. (2021), Planinić, Đurasević, and Jakobović (2021), Planinić, Đurasević, and Jakobović (2021), Đurasević and Jakobović (2022), Ulaga et al. (2022), Planinić et al. (2022)
batch	D.-W. Kim et al. (2002), D.-W. Kim et al. (2003), LOGENDRAN and SUBUR (2004), J.-P. Arnaout and Rabadi (2005), J.-P.M. Arnaout et al. (2006), Na et al. (2006), Silva and Magalhaes (2006), S. Xu and Bean (2007), Celano et al. (2008), Dolgui et al. (2009), Klemmt et al. (2009), Bozorgirad and Logendran (2012), W.-L. Wang et al. (2013), X. Li et al. (2013), Shahvari and Logendran (2015), X. Xu et al. (2015), Arroyo and Leung (2017a), Shahvari and Logendran (2017b), Shahidi-Zadeh et al. (2017), Shahvari and Logendran (2017a), H. Lu and Qiao (2017), Arroyo and Leung (2017b), Joo and Kim (2017), S. Lu et al. (2018), S. Zhou et al. (2018), Arroyo et al. (2019), Zarook, Yaser et al. (2021)
R	J.-F. Chen (2004), J.-F. Chen and Wu (2006), J.-F. Chen (2006), Celano et al. (2008), Ruiz and Andrés-Romano (2011), Cappadonna. et al. (2012), Cappadonna et al. (2013), Costa et al. (2013), Low et al. (2013), Torabi et al. (2013), Bitar et al. (2014), Afzalirad and Shafipour (2015), X. Xu et al. (2015), Afzalirad and Rezaeian (2016b), Özpeynirci et al. (2016), long Zheng and Wang (2016), Low and Wu (2016), Shahvari and Logendran (2017a), MANUPATI et al. (2017), Fanjul-Peyro et al. (2017), Villa et al. (2018), Zheng and Wang (2018), Vallada et al. (2019), Al-harkan and Qamhan (2019), Yepes-Borrero et al. (2020), M.-Z. Wang et al. (2020) Pinheiro et al. (2020), D.-Y. Lin and Huang (2021), Al-harkan et al. (2021), Yepes-Borrero et al. (2021), L. Zhang et al. (2021)
p_c	Low et al. (2013), Salehi Mir and Rezaeian (2016), Low and Wu (2016), S. Lu et al. (2018), Soleimani et al. (2020)
j_f	Klemmt et al. (2009)
$rwrk$	Ramezanian and Saidi-Mehrabad (2012), Rambod and Rezaeian (2014), X. Wang et al. (2020)
L	Bilyk and Mönch (2010)
M_d	Bandyopadhyay and Bhattacharya (2013), Abedi et al. (2017), Ghaleb et al. (2020)

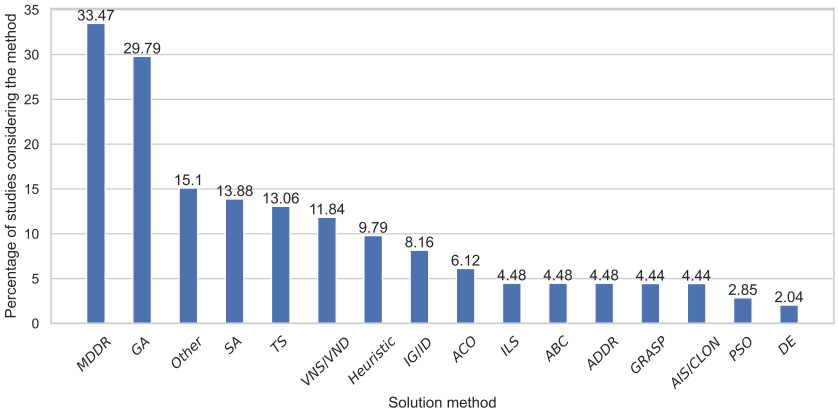

Fig. 6 Distribution of research dealing with different solution methods

M_m	Ebrahimi and Rezaeian (2015), Abedi et al. (2017), S. Lu et al. (2018), Avalos-Rosales et al. (2018), M. Wang and Pan (2019), Lei and Liu (2020), Ghaleb et al. (2020)
j_s	X. Li et al. (2013), Arroyo and Leung (2017b), Arroyo and Leung (2017a), Shahidi-Zadeh et al. (2017), Shahvari and Logendran (2017a), S. Zhou et al. (2018), Arroyo et al. (2019), Zarook, Yaser et al. (2021)
Q_k	Arroyo and Leung (2017a), Shahidi-Zadeh et al. (2017), Shahvari and Logendran (2017a), S. Zhou et al. (2018), Arroyo et al. (2019), Zarook, Yaser et al. (2021)
M_s	Zheng and Wang (2018), X. Wu and Che (2019), Pan et al. (2020)

Table 4 shows the application of different methods in the reviewed studies, while Figure 4 outlines the percentage by which each solution method was applied in the surveyed studies. Since many papers usually combine several methods, these studies are classified in a way that they are enumerated aside each method that is applied. This is especially true for DRs which are often used to generate initial solutions for metaheuristic methods. DRs are applied on average in every third paper, where in around half of these DRs are used in combination with other methods to improve their results. This is most often done by generating initial solutions, or using DRs in some solution construction methods. This provides further motivation to continue the research in designing better DRs. In most cases, DRs were designed manually, however, there is a small number of studies in the last few years which propose that DRs are generated automatically, and such approaches demonstrated promising results. In comparison, other problem specific heuristics that are developed for the UPMSP make up only around 10% of all research. This shows that for this type of problem the design of heuristic methods goes more into the direction of DRs due to their simplicity.

Out of the metaheuristic methods, GAs are most commonly used, with around 30% of research using GAs for solving this problem. Their popularity

comes from the fact that GA is a quite powerful, yet simple and very flexible method that can be used for optimisation. Aside from GAs, the next two most popular methods are usually simple single solution based metaheuristics like SA and TS, being used in around 13% of studies. This shows that relatively simple methods can obtain quite good results if designed well. LS based methods, like VNS, GRASP and similar are also very popular, either being used by themselves or in combination with other metaheuristics. Such methods have jointly been used in more than a quarter of all research. Naturally, simple LS operators by themselves have been used in even more studies, because they are commonly included in different metaheuristic methods which by default need not use them. The reason why LS operators have been adopted so frequently is because of their simplicity and effectiveness across various problem types.

Other methods received less attention, and were usually applied in only a few studies. Among these algorithms ACO was among the most frequently used. This is quite surprising as the algorithm needs some adaptation to be applied for the UPMSP. However, regardless of this, several studies applied this algorithm and focused especially on how to adapt the solution representation to the problem. Other classical metaheuristic methods include ABC, DE, CLONALG, AIS, and PSO; each of them was used in less than 5% of papers. Other methods which could not be classified to any of these categories are applied in around 15% of the research. Most of the studies which use them have been published in recent years and focused on applying new metaheuristics methods on the UPMSP, like FA, SCA, HHA, and similar. However, majority of these algorithms are used only in a single study and have not achieved the same popularity and adaption as classical metaheuristic methods. It has yet to be seen if any of these methods will gain more popularity and a larger importance in the UPMSP. For now, it seems that most research is still oriented towards classical metaheuristic methods.

Table 4: Research classification based on the solution methods

Algorithm	References
MDDRs	Ibarra and Kim (1977), De and Morton (1980), Hariri and Potts (1991), Glass et al. (1994), RANDHAWA and SMITH (1995), Maheswaran et al. (1999), T. Braun et al. (1999), T.D. Braun et al. (2001), Weng et al. (2001), M.-Y. Wu and Shu (2001), D.-W. Kim et al. (2003), Ritchie and Levine (2003), LOGENDRAN and SUBUR (2004), Du Kim and Kim (2004), Golconda et al. (2004), J.-P. Arnaout and Rabadi (2005), Jou (2005), J.-F. Chen and Wu (2006), J.-F. Chen (2006), Helal et al. (2006), S.-I. Kim et al. (2006), J.-P.M. Arnaout et al. (2006), Na et al. (2006), Z. Zhang et al. (2006), Xhafa et al. (2007), Luo et al. (2007), S.-I. Kim et al. (2007), Yildirim et al. (2007), C.-L. Chen (2008), Munir et al. (2008), Klemmt et al. (2009), Tseng et al. (2009), J.-F. Chen (2009), Izakian et al. (2009), Bilyk and Mönch (2010), Liu and Yang (2011), Y. Lin et al. (2011), Ruiz and Andrés-Romano (2011), Mehravaran and Logendran (2011), C.-L. Chen (2011), Haddad et al. (2012), Keskinturk et al. (2012), Ramezanian and Saidi-Mehrabad (2012), Rafsanjani and Bardsiri (2012), Briceño et al. (2012), I.-L. Wang et al. (2012), Yang-Kuei and Chi-Wei (2013), X. Li et al. (2013), C.-W. Lin et al. (2013), Y.-K. Lin et al. (2013), Cota et al. (2014), Hassan Abdel-Jabbar et al. (2014), Y.-K. Lin and Hsieh (2014), e Santos and Madureira (2014), Liang et al. (2015), S.-W. Lin and Ying (2015), Joo and Kim (2015), Y.-K. Lin and Lin (2015), Z. Li et al. (2015), Strohhecker et al. (2016), Đurasevic et al. (2016), Đurasevic and Jakobovic (2016), Arroyo and Leung (2017b), Arroyo and Leung (2017a), Joo and Kim (2017), Đurasević and Jakobović (2017b), Đurasević and Jakobović (2017a), Đurasevic and Jakobovic (2018), Zheng and Wang (2018), Bektur and Sarac (2019), Arroyo et al. (2019), Vlašic et al. (2019), de Abreu and de Athayde Prata (2020), Đurasevic and Jakobovic (2020), Đurasević and Jakobović (2020), Bhardwaj et al. (2020), Marinho Diana and de Souza (2020), Orts et al. (2020), X. Wang et al. (2020), Zarook, Yaser et al. (2021), Van and Hop (2021), Jaklinovic et al. (2021), L. Zhang et al. (2021)
ADDRs	Đurasevic et al. (2016), Đurasević and Jakobović (2017b), Đurasević and Jakobović (2017a), Đurasević and Jakobović (2019), Vlašic et al. (2019), Đurasevic and Jakobovic (2020), Đurasević and Jakobović (2020), Jaklinovic et al. (2021), Planinić, Đurasević, and Jakobović (2021), Planinić, Đurasević, and Jakobović (2021), Đurasević and Jakobović (2022), Planinić et al. (2022)
Heuristic	Suresh and Chaudhuri (1994), SURESH and GHAUDHURI (1996), Herrmann et al. (1997), Randhawa and Kuo (1997), Bank and Werner (2001), Dhaenens-Flipo (2001), Al-Salem and Armacost (2002), D.-W. Kim et al. (2003), J.-F. Chen (2004), Silva and Magalhaes (2006), Dolgui et al. (2009), Jolai et al. (2009), Y. Lin et al. (2011), Liu (2013), C.-H. Lee et al. (2014), X. Xu et al. (2015), Z. Li et al. (2015), Y.-K. Lin and Lin (2015), C.-J. Liao et al. (2016), Fanjul-Peyro et al. (2017), Che et al. (2017), Villa et al. (2018), Perez-Gonzalez et al. (2019), Yepes-Borrero et al. (2020)

GA	Glass et al. (1994), T. Braun et al. (1999), T.D. Braun et al. (2001), Cochran et al. (2003), Ritchie and Levine (2003), Peng and Liu (2004), Jou (2005), Gao (2005), Min and Cheng (2006), S. Xu and Bean (2007), Yildirim et al. (2007), Raja et al. (2008), Gao et al. (2008), Tavakkoli-Moghaddam et al. (2009), Chyu and Chang (2009), Jolai et al. (2009), Chyu and Chang (2010), Balin (2011), P.-C. Chang and Chen (2011), Vallada and Ruiz (2011), Y. Lin et al. (2011), Haddad et al. (2012), Keskinturk et al. (2012), Briceño et al. (2012), Cappadonna. et al. (2012), Cappadonna et al. (2013), Costa et al. (2013), Liu (2013), Y.-K. Lin et al. (2013), Bandyopadhyay and Bhattacharya (2013), Caniyilmaz et al. (2014), Bitar et al. (2014), Eroglu et al. (2014), Rambod and Rezaeian (2014), Hassan Abdel-Jabbar et al. (2014), Zeidi and MohammadHosseini (2015), Sels et al. (2015), Afzalirad and Shafipour (2015), Ebrahimi and Rezaeian (2015), Joo and Kim (2015), C.-J. Liao et al. (2016), Đurasevic et al. (2016), Afzalirad and Rezaeian (2016a), Salehi Mir and Rezaeian (2016), Đurasevic and Jakobovic (2016), Afzalirad and Rezaeian (2016b), long Zheng and Wang (2016) Arroyo and Leung (2017a), Joo and Kim (2017), Afzalirad and Rezaeian (2017), Rezaeian Zeidi et al. (2017), Shahidi-Zadeh et al. (2017), Abedi et al. (2017), C.-Y. Cheng and Huang (2017), MANUPATI et al. (2017), H. Lu and Qiao (2017), Tozzo et al. (2018), S. Zhou et al. (2018), Arık (2019), Vlašic et al. (2019), Soleimani et al. (2020), de Abreu and de Athayde Prata (2020), Vlašic et al. (2020), Marinho Diana and de Souza (2020), Orts et al. (2020), X. Wang et al. (2020), Đurasević and Jakobović (2020), Pan et al. (2020), Zarook, Yaser et al. (2021), Jaklinovic et al. (2021), Yepes-Borrero et al. (2021), Van and Hop (2021), L. Zhang et al. (2021)
SA	Glass et al. (1994), Herrmann et al. (1997), T. Braun et al. (1999), T.D. Braun et al. (2001), Bank and Werner (2001), D.-W. Kim et al. (2002), Anagnostopoulos and Rabadi (2002), D.-W. Kim et al. (2003), J.-F. Chen (2004), Na et al. (2006), J.-F. Chen and Wu (2006), J.-F. Chen (2006), Min and Cheng (2006), GUO et al. (2007), S.-I. Kim et al. (2007), Celano et al. (2008), J.-F. Chen (2009), Cruz-Chavez et al. (2009), Chyu and Chang (2010), Ying et al. (2010), P.-C. Chang and Chen (2011), T. Liao et al. (2014), Zeidi and MohammadHosseini (2015), S.-W. Lin and Ying (2015), Santos et al. (2016), Arroyo and Leung (2017a), Ezugwu (2019), Al-harkan and Qamhan (2019), Bektur and Sarac (2019), Jouhari et al. (2019), Arık (2019), X. Wang et al. (2020), Ghaleb et al. (2020), Pinheiro et al. (2020), D.-Y. Lin and Huang (2021), Ulaga et al. (2022)
TS	Glass et al. (1994), Piersma and van Dijk (1996), Suresh and Chaudhuri (1996), Srivastava (1998), T. Braun et al. (1999), T.D. Braun et al. (2001), Vredeveld and Hurkens (2002), LOGENDRAN and SUBUR (2004), Cao et al. (2005), J.-F. Chen and Wu (2006), Helal et al. (2006), S.-I. Kim et al. (2006), GUO et al. (2007), S.-I. Kim et al. (2007), Logendran et al. (2007), C.-L. Chen and Chen (2008), C.-L. Chen (2008), Mehravaran and Logendran (2011), C.-L. Chen (2011), Bozorgirad and Logendran (2012), J.-H. Lee et al. (2013), T. Liao et al. (2014), Sels et al. (2015), Shahvari and Logendran (2015), Y.-K. Lin and Lin (2015), S.-W. Lin et al. (2016), Özpeynirci et al. (2016), Shahvari and Logendran (2017b), S. Lu et al. (2018), H. Wang and Alidaee (2019), Bektur and Sarac (2019), Ulaga et al. (2022)
DE	W.-L. Wang et al. (2013), T. Liao et al. (2014), X. Wu and Che (2019), Pan et al. (2020), Nanthapodej et al. (2021a)
AIS/CLON	Gao (2010), Niu et al. (2011), R.O. Diana et al. (2013), R.O.M. Diana et al. (2015), Afzalirad and Rezaeian (2016b), R.O.M. Diana et al. (2017), Perez-Gonzalez et al. (2019), Ulaga et al. (2022)

Other	T. Braun et al. (1999), T.D. Braun et al. (2001), Bank and Werner (2001), Z. Zhang et al. (2006), J.-F. Chen (2006), Rabadi et al. (2006), GUO et al. (2007), Chyu and Chang (2009), J.-F. Chen (2013), Cota et al. (2014), Avalos-Rosales et al. (2014), Kayvanfar and Teymourian (2014), Siepak and Józefczyk (2014), Nohra Haddad et al. (2014), Y.-K. Lin and Hsieh (2014), Polyakovskiy and M'Hallah (2014), L. Wang et al. (2016), long Zheng and Wang (2016), Shahidi-Zadeh et al. (2017), Abedi et al. (2017), Ezugwu et al. (2018), Ezugwu and Akutsah (2018), Zheng and Wang (2018), Pan et al. (2018), Ezugwu (2019), Vallada et al. (2019), M. Wang and Pan (2019), Jouhari et al. (2019), J.-P. Arnaout (2019), Lei, Yuan, Cai, and Bai (2020), Soleimani et al. (2020), Jouhari et al. (2020), Ewees et al. (2021), Al-harkan et al. (2021), Al-qaness et al. (2021), Jovanovic and Voß (2021), L. Zhang et al. (2021)
ILS	Vredeveld and Hurkens (2002), Ritchie and Levine (2003), Siepak and Józefczyk (2014), de C. M. Nogueira et al. (2014), Nohra Haddad et al. (2014), Santos et al. (2016), Cota et al. (2017), Kramer and Subramanian (2017), Avalos-Rosales et al. (2018), R.O. Diana et al. (2018), de Abreu and de Athayde Prata (2020), Ulaga et al. (2022)
GRASP	Rojanasoonthon and Bard (2005), de Paula et al. (2007), Ravetti et al. (2007), Rodriguez, Blum, et al. (2012), R.O. Diana et al. (2013), Cota et al. (2014), de C. M. Nogueira et al. (2014), R.O.M. Diana et al. (2015), Liang et al. (2015), Perez-Gonzalez et al. (2019), Yepes-Borrero et al. (2020)
VNS/VND	de Paula et al. (2007), C.-L. Chen (2008), C.-L. Chen and Chen (2008), Klemmt et al. (2009), Charalambous et al. (2010), Fanjul-Peyro and Ruiz (2010), Bilyk and Mönch (2010), Fanjul-Peyro and Ruiz (2011), C.-L. Chen (2011), Fleszar et al. (2011), Haddad et al. (2012), Fanjul-Peyro and Ruiz (2012), R.O. Diana et al. (2013), Cota et al. (2014), Avalos-Rosales et al. (2014), Kayvanfar and Teymourian (2014), Nohra Haddad et al. (2014), R.O.M. Diana et al. (2015), Kramer and Subramanian (2017), Tozzo et al. (2018), R.O. Diana et al. (2018), Al-harkan and Qamhan (2019), Perez-Gonzalez et al. (2019), de Abreu and de Athayde Prata (2020), Terzi et al. (2020), Marinho Diana and de Souza (2020), Nanthapodej et al. (2021b), Nanthapodej et al. (2021a), Ulaga et al. (2022)
ACO	H. Zhou et al. (2007), J.-P. Arnaout et al. (2008), J.-P. Arnaout et al. (2009), J.-P. Arnaout et al. (2012), Keskinturk et al. (2012), C.-W. Lin et al. (2013), Low et al. (2013), Y.-C. Chang et al. (2014), T. Liao et al. (2014), Liang et al. (2015), Low and Wu (2016), Arroyo and Leung (2017a), Afzalirad and Rezaeian (2017), Shahidi-Zadeh et al. (2017), Bhardwaj et al. (2020)
IG/ID	Hariri and Potts (1991), Glass et al. (1994), Piersma and van Dijk (1996), Bank and Werner (2001), J.-F. Chen (2009), S.-W. Lin et al. (2010), Rodriguez et al. (2013), J.-F. Chen (2013), Santos et al. (2016), S.-W. Lin et al. (2016), L. Wang et al. (2016), Arroyo and Leung (2017a), H. Wang and Alidaee (2019), Vallada et al. (2019), Arroyo et al. (2019), Marinho Diana and de Souza (2020), Pinheiro et al. (2020), Terzi et al. (2020) Yepes-Borrero et al. (2021), C.-Y. Cheng et al. (2021)
PSO	Y.-K. Lin (2013), Torabi et al. (2013), Salehi Mir and Rezaeian (2016), Shahidi-Zadeh et al. (2017), Shahvari and Logendran (2017a), MANUPATI et al. (2017), M.-Z. Wang et al. (2020)
ABC	Ying and Lin (2012), Rodriguez, García-Martínez, et al. (2012), Caniyilmaz et al. (2014), S.-W. Lin and Ying (2014), Rambod and Rezaeian (2014), S. Lu et al. (2018), Arık (2019), Lei, Yuan, and Cai (2020), Soleimani et al. (2020), Lei and Liu (2020), Marinho Diana and de Souza (2020), C.-Y. Cheng et al. (2021)

Table 5: Classification of UPMSP research

Reference	Scheduling problem	Solution method
Ibarra and Kim (1977)	$R \mid\mid C_{max}$	MDDRs
De and Morton (1980)	$R \mid\mid C_{max}$	MDDRs
Hariri and Potts (1991)	$R \mid\mid C_{max}$	MDDRs, ID
Tamaki et al. (1993)	$R \mid s_{ij}, r_j, M_j, R \mid C_{max} + \sum w_j T_j + T_{max}$	MDDRs, ID
Glass et al. (1994)	$R \mid\mid C_{max}$	GA, TS, SA, ID, MDDR
Suresh and Chaudhuri (1994)	$R \mid\mid T_{max}$	heuristic
RANDHAWA and SMITH (1995)	$R \mid s_{ij} \mid \sum F_j$	MDDRs
	$R \mid s_{ij} \mid ML$	
	$R \mid s_{ij} \mid \sum U_j$	
SURESH and GHAUDHURI (1996)	$R \mid brkdwn \mid C_{max}$	heuristic
Piersma and van Dijk (1996)	$R \mid\mid C_{max}$	ID, TS
Suresh and Chaudhuri (1996)	$R \mid\mid C_{max}, T_{max}$	TS
Herrmann et al. (1997)	$R \mid prec \mid C_{max}$	heuristic, SA
Randhawa and Kuo (1997)	$R \mid s_{ijk}, M_j \mid \sum F_j, \sum T_j, \sum U_j$	heuristic
Srivastava (1998)	$R \mid\mid C_{max}$	TS
Maheswaran et al. (1999)	$R \mid\mid C_{max}$	MDDRs
T. Braun et al. (1999)	$R \mid\mid C_{max}$	MDDRs, GA, SA, TS, GSA, A*
T.D. Braun et al. (2001)	$R \mid\mid C_{max}$	MDDRs, GA, SA, TS, GSA, A*
Weng et al. (2001)	$R \mid s_{ij} \mid \sum w_j F_j$	MDDRs
Bank and Werner (2001)	$R \mid r_j, d_j = d \mid \sum (w_j E_j + w_j T_j)$	heuristics, TS, TA, ID
Dhaenens-Flipo (2001)	$R \mid s_{ijk} \mid C_{max} + COST$	heuristic
M.-Y. Wu and Shu (2001)	$R \mid\mid C_{max}$	MDDR
Al-Salem and Armacost (2002)	$R \mid M_j \mid C_{max}$	heuristic
D.-W. Kim et al. (2002)	$R \mid s_{ijk}, s-batch \mid \sum T_j$	SA
Anagnostopoulos and Rabadi (2002)	$R \mid s_{ijk} \mid C_{max}$	SA
Vredeveld and Hurkens (2002)	$R \mid\mid \sum w_j F_j$	SA
Cochran et al. (2003)	$R \mid\mid C_{max}, \sum W_j T_j$	GA
	$R \mid\mid C_{max}$	
	$\sum w_j F_j, \sum W_j T_j$	

D.-W. Kim et al. (2003)	$R \mid s_{ij}, s-batch \mid \sum w_j T_j$	SA, MDDDRs, heuristic
Ritchie and Levine (2003)	$R \mid\mid C_{max}$	MDDRs, GA, ILS
J.-F. Chen (2004)	$R \mid s_k, R \mid C_{max}$	heuristic, SA
Peng and Liu (2004)	$R \mid\mid C_{max}$	GA
	$R \mid\mid Ml$	
	$R \mid\mid T_{max}$	
LOGENDRAN and SUBUR (2004)	$R \mid s-batch, brkdwn, M_j, r_j \mid \sum w_j T_j$	TS, MDDRs
Du Kim and Kim (2004)	$R \mid r_j \mid C_{max}$	MDDRs
Golconda et al. (2004)	$R \mid\mid \sum U_j$	MDDRs
J.-P. Arnaout and Rabadi (2005)	$R \mid s_{ij}, s-batch \mid \sum w_j F_j$	MDDRs
Jou (2005)	$R \mid s_{ik} \mid \sum (w_j E_j + w_j T_j) + ML$	MDDRs, GA
Cao et al. (2005)	$R \mid\mid \sum w_j T_j + COST$	TS
Gao (2005)	$R \mid\mid C_{max}$	GA
Rojanasoonthon and Bard (2005)	$R \mid r_j, \bar{d}, M_j, s_{ij} \mid \sum w_j U_j$	GRASP
J.-P.M. Arnaout et al. (2006)	$R \mid s_{ij}, s-batch \mid \sum w_j F_j$	MDDRs
Na et al. (2006)	$R \mid s_{ij}, p-batch \mid \sum w_j T_j$	MDDRs, SA
Z. Zhang et al. (2006)	$R \mid s_{ij}, M_j \mid \sum w_j T_j$	Q-learning, MDDRs
J.-F. Chen and Wu (2006)	$R \mid s_{ij}, R, M_j \mid \sum T_j$	MDDRs, TS, SA, TA
J.-F. Chen (2006)	$R \mid s_{ij}, R, M_j \mid T_{max}$	RRT, TS, SA, MDDR
Min and Cheng (2006)	$R \mid d_j = d \mid \sum (w_j E_j + w_j T_j)$	GA, SA
Helal et al. (2006)	$R \mid s_{ijk} \mid C_{max}$	TS, MDDR
Rabadi et al. (2006)	$R \mid s_{ijk} \mid C_{max}$	ILS
Silva and Magalhaes (2006)	$R \mid s_{ij}, s-batch, M_j \mid \sum U_j$	heuristic
S.-I. Kim et al. (2006)	$R \mid s_{ij}, r_j \mid \sum T_j$	TS
Xhafa et al. (2007)	$R \mid\mid C_{max}$	MDDRs
	$R \mid\mid \sum F_j$	
	$R \mid\mid ML$	
Luo et al. (2007)	$R \mid\mid C_{max}$	MDDRs
GUO et al. (2007)	$R \mid\mid C_{max}$	TS, SA, SWO
de Paula et al. (2007)	$R \mid s_{ijk} \mid C_{max} + \sum w_j T_j$	GRASP, VNS
S. Xu and Bean (2007)	$R \mid p-batch \mid C_{max}$	GA
H. Zhou et al. (2007)	$R \mid\mid \sum w_j T_j$	ACO
Ravetti et al. (2007)	$R \mid s_{ijk} \mid C_{max} + \sum w_j T_j$	GRASP
Logendran et al. (2007)	$R \mid r_j, brkdwn \mid \sum T_j$	TS

S.-I. Kim et al. (2007)	$R \mid s_{ij}, r_j \mid \sum T_j$	TS, SA, MDDRs
Yildirim et al. (2007)	$R \mid s_{ijk}, M_j \mid \sum F_j$	GA, MDDRs
Munir et al. (2008)	$R \mid\mid C_{max}$	MDDRs
C.-L. Chen and Chen (2008)	$R \mid s_{ijk} \mid \sum w_j U_j$	VNS, TS, MDDR
J.-P. Arnaout et al. (2008)	$R \mid s_{ijk} \mid C_{max}$	ACO
Raja et al. (2008)	$R \mid s_{ij} \mid \sum w_j E_j + w_j T_j$	GA
J.-F. Chen (2009)	$R \mid s_{ijk}, deadline \mid \sum T_j$	SA, MDDR
Gao et al. (2008)	$R \mid M_j \mid \sum (w_j E_j + w_j T_j) + C_{max}$	GA
Celano et al. (2008)	$R \mid s_{ijk}, R, p-batch \mid C_{max}$	SA
	$R \mid s_{ijk}, R, p-batch \mid \sum T_j$	
Klemmt et al. (2009)	$R \mid r_j, p-batch, J_f \mid \sum w_j T_j$	VNS and MDDR
J.-P. Arnaout et al. (2009)	$R \mid s_{ijk} \mid C_{max}$	ACO
Dolgui et al. (2009)	$R \mid s_{ijk}, s-batch, M_j \mid C_{max}$	heuristic
Tseng et al. (2009)	$R \mid s_{ijk} \mid \sum C_{max}$	MDDRs
	$R \mid s_{ijk} \mid \sum w_j T_j$	
	$R \mid s_{ijk} \mid COST$	
C.-L. Chen (2008)	$R \mid s_{ijk} \mid \sum w_j U_j$	MDDR, VNS, TS
Izakian et al. (2009)	$R \mid\mid C_{max}$	MDDR
	$R \mid\mid \sum F_j$	
Tavakkoli-Moghaddam et al. (2009)	$R \mid s_{ij}, r_j, prec \mid \sum U_j$	GA
	$R \mid s_{ij}, r_j, prec \mid \sum F_j$	
Cruz-Chavez et al. (2009)	$R \mid\mid \sum w_j F_j$	SA
Chyu and Chang (2009)	$R \mid s_{ijk} \mid \sum w_j F_j, \sum w_j T_j$	GA, ES
Jolai et al. (2009)	$R \mid\mid N_{jit}$	GA, heuristic
Charalambous et al. (2010)	$R \mid\mid \sum C_{max}$	VND
Fanjul-Peyro and Ruiz (2010)	$R \mid\mid C_{max}$	VND
Gao (2010)	$R \mid M_j \mid \sum (w_j E + w_j T_j) + C_{max}$	AIS
Ying et al. (2010)	$R \mid s_{ijk} \mid C_{max}$	SA
Chyu and Chang (2010)	$R \mid s_{ijk} \mid \sum w_j F_j, \sum w_j T_j$	GA, SA
S.-W. Lin et al. (2010)	$R \mid s_{ijk} \mid \sum T_j$	IG
Bilyk and Mönch (2010)	$R \mid r_j, MP, L \mid \sum w_j T_j$	VNS, MDDR
Vallada and Ruiz (2011)	$R \mid s_{ijk} \mid C_{max}$	GA
Liu and Yang (2011)	$R \mid prec \mid C_{max}$	MDDR

Ruiz and Andrés-Romano (2011)	$R \mid s_{ijk}, R \mid \sum F_j + R_u$	MDDRs
Balin (2011)	$R \mid\mid C_{max}$	GA
P.-C. Chang and Chen (2011)	$R \mid s_{ijk} \mid C_{max}$	GA, SA
Fanjul-Peyro and Ruiz (2011)	$R \mid\mid C_{max}$	VND
Mehravaran and Logendran (2011)	$R \mid s_{ijk} \mid \sum w_j F_j + \sum w_j T_j$	TS, MDDRs
Y. Lin et al. (2011)	$R \mid\mid \sum w_j T_j$	heuristics, MDDRs, GA
	$R \mid\mid \sum w_j F_j$	
	$R \mid\mid C_{max}$	
Niu et al. (2011)	$R \mid s_{ijk} \mid C_{max}$	CLONALG
C.-L. Chen (2011)	$R \mid s_{ijk}, r_j \mid \sum w_j U_j$	TS, VSN, MDDRs
Fleszar et al. (2011)	$R \mid s_{ijk} \mid C_{max}$	VNS
Haddad et al. (2012)	$R \mid s_{ijk} \mid C_{max}$	VND, GA, MDDR
Fanjul-Peyro and Ruiz (2012)	$R \mid\mid C_{max}$	VND
Bozorgirad and Logendran (2012)	$R \mid r_j, M_j, brkdwn, s-batch \mid \sum w_j F_j + \sum w_j T_j$	TS
J.-P. Arnaout et al. (2012)	$R \mid s_{ijk} \mid C_{max}$	ACO
Ramezanian and Saidi-Mehrabad (2012)	$R \mid r_j, rwrk \mid C_{max}$	MDDRs
Rafsanjani and Bardsiri (2012)	$R \mid\mid C_{max}$	MDDR
	$R \mid\mid \sum F_j$	
Briceño et al. (2012)	$R \mid\mid C_{max}$	GA, MDDRs
I.-L. Wang et al. (2012)	$R \mid r_j, M_j, s_{ijk} \mid \sum U_j + C_{max}$	MDDR
Rodriguez, García-Martínez, et al. (2012)	$R \mid\mid \sum w_j F_j$	ABC
Keskinturk et al. (2012)	$R \mid s_{ijk} \mid C_{max}$	ACO, GA, MDDRs
Rodriguez, Blum, et al. (2012)	$R \mid\mid \sum w_j F_j$	GRASP
Ying and Lin (2012)	$R \mid s_{ijk}, d \mid \sum T_j$	ABC
Cappadonna. et al. (2012)	$R \mid s_{ijk}, R \mid C_{max}$	GA
Y.-K. Lin (2013)	$R \mid r_j \mid C_{max}$	PSO
Yang-Kuei and Chi-Wei (2013)	$R \mid r_j \mid C_{max}$	MDDRs
	$R \mid r_j \mid \sum w_j F_j$	
	$R \mid r_j \mid \sum w_j T_j$	
X. Li et al. (2013)	$R \mid p-batch, J_s \mid C_{max}$	MDDRs
Cappadonna et al. (2013)	$R \mid s_{ij}, R \mid C_{max}$	GA
Costa et al. (2013)	$R \mid s_{ijkl}, R \mid C_{max}$	GA
C.-W. Lin et al. (2013)	$R \mid\mid \sum w_j T_j$	ACO, MDDR
Liu (2013)	$R \mid prec \mid \sum T_j$	GA, heuristic

Y.-K. Lin et al. (2013)	$R \mid\mid C_{max}, \sum w_j T_j$	GA, MDDR
	$R \mid\mid \sum w_j F_j, \sum w_j T_j$	
	$R \mid\mid C_{max}, \sum w_j F_j \sum w_j T_j$	
W.-L. Wang et al. (2013)	$R \mid s-batch, M_j \mid C_{max}$	DE
Low et al. (2013)	$R \mid M_j, R, p_c \mid C_{max}$	ACO
Torabi et al. (2013)	$R \mid s_{ijk}, r_j, R \mid C_{max}, \sum w_j T_j, \sum w_j F_j$	PSO
Bandyopadhyay and Bhattacharya (2013)	$R \mid s_{ik}, M_d \mid COST, \sum w_j * T_j, C_{max}$	GA
J.-H. Lee et al. (2013)	$R \mid s_{ijk}, d \mid \sum T_j$	TS
Rodriguez et al. (2013)	$R \mid\mid \sum w_j F_j$	IG
J.-F. Chen (2013)	$R \mid s_{ijk}, d \mid \sum w_j F_j$	RRT, ID, MDDRs
R.O. Diana et al. (2013)	$R \mid s_{ijkl} \mid C_{max}$	CLONALG, VND, GRASP
Bitar et al. (2014)	$R \mid s_{ijk}, R \mid \sum w_j F_j$	GA
Y.-K. Lin and Hsieh (2014)	$R \mid s_{ijk}, r_j \mid \sum w_j T_j$	MDDR, EMA
Y.-C. Chang et al. (2014)	$R \mid\mid \sum w_j F_j$	ACO
Avalos-Rosales et al. (2014)	$R \mid s_{ijk} \mid C_{max}$	multi start, VND
Caniyilmaz et al. (2014)	$R \mid s_{ijk}, M_j \mid C_{max} + \sum T_j$	ABC, GA
Kayvanfar and Teymourian (2014)	$R \mid s_{ijk}, M_j \mid C_{max} + \sum (w_j T_j + w_j E_j)$	IWDA, VNS
S.-W. Lin and Ying (2014)	$R \mid s_{ijk} \mid C_{max}$	ABC
Eroglu et al. (2014)	$R \mid s_{ijk} \mid C_{max}$	GA
Siepak and Józefczyk (2014)	$R \mid\mid \sum F_j$	SS, ILS
T. Liao et al. (2014)	$R \mid r_j, s_{ijk} \mid C_{max}$	ACO, TS, SA, DE
Rambod and Rezaeian (2014)	$R \mid r_j, rwrk, M_j, s_{ijk} \mid C_{max}$	ABC, GA
de C. M. Nogueira et al. (2014)	$R \mid s_{ijk} \mid \sum w_j T_j + w_j E_j$	GRASP, ILS
Nohra Haddad et al. (2014)	$R \mid s_{ijk} \mid C_{max}$	AIV
Hassan Abdel-Jabbar et al. (2014)	$R \mid prec \mid C_{max}$	GA, MDDR
Polyakovskiy and M'Hallah (2014)	$R \mid\mid \sum (w_j E + w_j T_j)$	multi-agent model
C.-H. Lee et al. (2014)	$R \mid s_{ij}, M_d, \bar{d}_j = D \mid \sum F_j$	heuristic
e Santos and Madureira (2014)	$R \mid\mid C_{max}$	MDDR
Cota et al. (2014)	$R \mid s_{ijk} \mid C_{max}$	AIRP, MDDR
Zeidi and MohammadHosseini (2015)	$R \mid s_{ijk} \mid \sum w_j T_j + w_j E_j$	GA, SA
Sels et al. (2015)	$R \mid\mid C_{max}$	GA, TS
R.O.M. Diana et al. (2015)	$R \mid s_{ijkl} \mid C_{max}$	CLONALG, VND, GRASP
Z. Li et al. (2015)	$R \mid r_j \mid \sum T_j + COST$	MDDRs, heuristics
X. Xu et al. (2015)	$R \mid s-batch, R \mid C_{max}$	heuristics

Liang et al. (2015)	$R \mid r_j \mid \sum w_j T_j + TEC$	ACO, GRASP, MDDRs
S.-W. Lin and Ying (2015)	$R \mid\mid \sum w_j T_j, \sum w_j F_j, C_{max}$	SA, MDDRs
Ebrahimi and Rezaeian (2015)	$R \mid M_d \mid C_{max}$	GA
Joo and Kim (2015)	$R \mid s_{ijk}, M_j \mid \sum F_j$	GA, MDDRs
Shahvari and Logendran (2015)	$R \mid s_{ijk}, M_j, brkdwn, r_j, batch \mid \sum w_j C_j + \sum w_j T_j$	TS
Y.-K. Lin and Lin (2015)	$R \mid r_j \mid \sum w_j T_j, C_{max}$	MDDR, heuristic, TS
Afzalirad and Shafipour (2015)	$R \mid R \mid C_{max}$	GA
S.-W. Lin et al. (2016)	$R \mid\mid \sum w_j T_j, \sum w_j F_j, C_{max}$	Pareto IG, TS, MDDRs
C.-J. Liao et al. (2016)	$R \mid s_{ijk} \mid C_{max}$	GA, heuristic
Afzalirad and Rezaeian (2016a)	$R \mid s_{ij}, M_j, r_j, prec \mid TLT$	GA
Afzalirad and Rezaeian (2016b)	$R \mid r_j, s_{ij}, M_j, prec, R \mid C_{max}$	GA, AIS
Đurasevic et al. (2016)	$R \mid r_j \mid C_{max}$	ADDRs, MDDRs, GA
	$R \mid r_j \mid \sum F_j$	
	$R \mid r_j \mid \sum w_j T_j$	
	$R \mid r_j \mid \sum w_j U_j$	
Strohhecker et al. (2016)	$R \mid s_{ijk} \mid \sum F_j$	MDDRs
Salehi Mir and Rezaeian (2016)	$R \mid r_j, s_{ijk}, p_c \mid \sum F_j$	GA, PSO
L. Wang et al. (2016)	$R \mid s_{ijk} \mid C_{max}$	EDA, IG
Özpeynirci et al. (2016)	$R \mid R \mid C_{max}$	TS
long Zheng and Wang (2016)	$R \mid R \mid C_{max}$	FA, GA
Santos et al. (2016)	$R \mid s_{ijk} \mid C_{max}$	SA, ILS, hill climbing
Low and Wu (2016)	$R \mid M_j, R, p_c \mid C_{max}$	ACO
Đurasevic and Jakobovic (2016)	$R \mid r_j \mid C_{max}$	GA, MDDRs
	$R \mid r_j \mid \sum F_j$	
	$R \mid r_j \mid \sum w_j T_j$	
	$R \mid r_j \mid \sum w_j U_j$	

Đurasević and Jakobović (2017b)	$R \mid r_j \mid C_{max}, F_{max}, T_{max}$	ADDRs, MDDRs
	$R \mid r_j \mid \sum w_j U_j, F_{max}, \sum w_j T_j$	
	$R \mid r_j \mid \sum F_j, \sum w_j F_j, \sum w_j T_j$	
	$R \mid r_j \mid C_{max}, F_{max}, \sum F_j$	
	$R \mid r_j \mid C_{max}, \sum w_j F_j, \sum w_j T_j$	
	$R \mid r_j \mid \sum w_j F_j, \sum (w_j E_j + w_j T_j), \sum F_j, \sum w_j U_j, \sum w_j T_j$	
	$R \mid r_j \mid C_{max}, \sum w_j F_j, ML, F_{max}, \sum F_j$	
	$R \mid r_j \mid F_{max}, \sum F_j, \sum w_j U_j, T_{max}, \sum w_j T_j$	
	$R \mid r_j \mid C_{max}, \sum (w_j E_j + w_j T_j), \sum F_j, ML, \sum w_j T_j$	
	$R \mid r_j \mid C_{max}, F_{max}, T_{max}, \sum w_j T_j, \sum w_j U_j, \sum F_j$	
	$R \mid r_j \mid C_{max}, F_{max}, T_{max}, \sum w_j T_j, \sum w_j U_j, \sum F_j, \sum w_j F_j$	
	$R \mid r_j \mid C_{max}, \sum w_j F_j, \sum F_j, F_{max}, ML, T_{max}, \sum w_j T_j$	
	$R \mid r_j \mid C_{max}, \sum (w_j E_j + w_j T_j), F_{max}, \sum F_j, ML, \sum w_j U_j, T_{max}$	
	$R \mid r_j \mid C_{max}, F_{max}, T_{max}, \sum w_j T_j, \sum w_j U_j, \sum F_j, \sum w_j F_j, ML, \sum (w_j E_j + w_j T_j)$	
Đurasević and Jakobović (2017a)	$R \mid r_j \mid C_{max}$	ADDRs, MDDRs
	$R \mid r_j \mid \sum F_j$	
	$R \mid r_j \mid \sum w_j T_j$	
	$R \mid r_j \mid \sum w_j U_j$	
Arroyo and Leung (2017a)	$R \mid r_j, p-batch, j_s, Q_k \mid C_{max}$	IG, GA, SA, ACO, MDDR
Arroyo and Leung (2017b)	$R \mid r_j, p-batch, J_s \mid C_{max}$	MDDRs
Joo and Kim (2017)	$R \mid s_{ijk}, p-batch \mid C_{max}$	GA, MDDRs
Afzalirad and Rezaeian (2017)	$R \mid s_{ij}, M_j, r_j, prec \mid \sum w_j T_j, \sum w_j F_j$	GA, ACO
Rezaeian Zeidi et al. (2017)	$R \mid s_{ij}, M_j, r_j \mid \sum C_j, \sum q_j E_j + w_j T_j$	GA
Shahvari and Logendran (2017a)	$R \mid s_{ij}, r_j, M_j, brkdwn, s-batch, j_s, Q_k \mid C_{max}, COST$	PSO
Shahvari and Logendran (2017b)	$R \mid s_{ij}, r_j, M_j, brkdwn, s-batch \mid \sum w_j F_j + \sum w_j T_j$	TS
Shahidi-Zadeh et al. (2017)	$R \mid r_j, p-batch, j_s, Q_k \mid C_{max}, \sum (w_j E_j + w_j T_j) + COST$	HS, PSO, ACO, GA
Abedi et al. (2017)	$R \mid M_j, M_d, M_m \mid \sum w_j E_j w_j T_j + COST$	GA, ICA
C.-Y. Cheng and Huang (2017)	$R \mid M_d \mid \sum E_j + T_j$	GA
MANUPATI et al. (2017)	$R \mid s_{ijk}, r_j, R \mid C_{max}, \sum F_j, \sum T_j, ML$	GA, PSO
Cota et al. (2017)	$R \mid s_{ijk} \mid \sum C_{max}$	ILS

Fanjul-Peyro et al. (2017)	$R \mid R \mid C_{max}$	heuristics
Che et al. (2017)	$R \mid\mid TEC$	heuristic
R.O.M. Diana et al. (2017)	$R \mid s_{ijk} \mid C_{max}$	immune network, VNS
H. Lu and Qiao (2017)	$R \mid p-batch \mid COST$	GA
Kramer and Subramanian (2017)	$R \mid\mid \sum w_j T_j$	ILS
	$R \mid r_j \mid \sum w_j T_j$	
	$R \mid s_{ijk} \mid \sum w_j T_j$	
	$R \mid r_j, s_{ijk} \mid \sum w_j T_j$	
	$R \mid\mid \sum (w_j E_j + w_j T_j)$	
	$R \mid r_j \mid \sum (w_j E_j + w_j T_j)$	
	$R \mid s_{ijk} \mid \sum w_j E_j + w_j T_j$	
	$R \mid r_j, s_{ijk} \mid \sum (w_j E_j + w_j T_j)$	
	$R \mid\mid \sum w_j F_j$	
	$R \mid r_j \mid \sum w_j F_j$	
	$R \mid s_{ijk} \mid \sum w_j F_j$	
	$R \mid r_j, s_{ijk} \mid \sum w_j F_j$	
S. Lu et al. (2018)	$R \mid p-batch, p_c, M_m \mid C_{max}$	ABC, TS
Tozzo et al. (2018)	$R \mid s_{ij} \mid C_{max}$	GA, VNS
Zheng and Wang (2018)	$R \mid R, M_s \mid C_{max}, TEC$	FA, MDDRs
Đurasevic and Jakobovic (2018)	$R \mid r_j \mid C_{max}$	MDDRs
	$R \mid r_j \mid \sum F_j$	
	$R \mid r_j \mid \sum w_j F_j$	
	$R \mid r_j \mid \sum w_j T_j$	
	$R \mid r_j \mid \sum w_j U_j$	
	$R \mid r_j \mid ML$	
	$R \mid r_j \mid F_{max}$	
	$R \mid r_j \mid T_{max}$	
Ezugwu et al. (2018)	$R \mid s_{ijk} \mid C_{max}$	SOS
Avalos-Rosales et al. (2018)	$R \mid s_{ijk}, M_m \mid C_{max}$	Multi-start ILS
S. Zhou et al. (2018)	$R \mid p-batch, j_s, Q_k \mid C_{max}$	GA
R.O. Diana et al. (2018)	$R \mid s_{ijk}, r_j \mid \sum w_j T_j$	VND, ILS
Ezugwu and Akutsah (2018)	$R \mid s_{ijk} \mid C_{max}$	FA

Pan et al. (2018)	$R \mid\mid \sum T_j, TEC$	ICS
Villa et al. (2018)	$R \mid R \mid C_{max}$	heuristics
Ezugwu (2019)	$R \mid s_{ijk} \mid C_{max}$	SA, SOS
H. Wang and Alidaee (2019)	$R \mid\mid \sum w_j F_j$	TS, IG
X. Wu and Che (2019)	$R \mid M_s \mid C_{max}, TEC$	DE
Vallada et al. (2019)	$R \mid R \mid C_{max}$	scatter search, IG
M. Wang and Pan (2019)	$R \mid M_m, s_{ijk} \mid C_{max}, \sum T_j$	ICA
Al-harkan and Qamhan (2019)	$R \mid s_{ij}, r_j, R \mid C_{max}$	VNS, SA
Bektur and Sarac (2019)	$R \mid s_{ij}, M_j \mid \sum w_j T_j$	TS, SA, MDDR
Arroyo et al. (2019)	$R \mid p-batch, r_j, js, Q_k \mid \sum F_j$	MDDRs, IG
Jouhari et al. (2019)	$R \mid s_{ijk} \mid C_{max}$	SCA, SA
Vlašic et al. (2019)	$R \mid r_j \mid \sum w_j T_j$	GA,MDDRs, ADDRs
Arık (2019)	$R \mid d_j = D \mid \sum w_j E_j + w_j T_j$	ABC, GA, SA
Đurasević and Jakobović (2019)	$R \mid r_j \mid C_{max}$	ADDRs, DRs
	$R \mid r_j \mid \sum F_j$	
	$R \mid r_j \mid \sum w_j T_j$	
	$R \mid r_j \mid \sum w_j U_j$	
Perez-Gonzalez et al. (2019)	$R \mid s_{ij}, M_j \mid \sum T_j$	heuristic, CLONALG, VND, GRASP
J.-P. Arnaout (2019)	$R \mid s_{ijk} \mid C_{max}$	WO
Yepes-Borrero et al. (2020)	$R \mid s_{ijk}, R \mid C_{max}$	GRASP, heuristics
Soleimani et al. (2020)	$R \mid s_{ijk}, p_c \mid \sum w_j T_j + TEC$	CSO, GA, ABC
Lei, Yuan, and Cai (2020)	$R \mid\mid C_{max}, \sum T_j$	ABC
Terzi et al. (2020)	$R \mid s_{ijk} \mid C_{max}$	hill climbing, VND
Lei, Yuan, Cai, and Bai (2020)	$R \mid\mid C_{max}$	ICA
de Abreu and de Athayde Prata (2020)	$R \mid s_{ijk} \mid C_{max}$	GA, VND, ILS, MDDRs
Vlašic et al. (2020)	$R \mid r_j \mid C_{max}$	GA
	$R \mid r_j \mid \sum F_j$	
	$R \mid r_j \mid \sum w_j T_j$	
	$R \mid r_j \mid \sum w_j U_j$	
Jouhari et al. (2020)	$R \mid\mid C_{max}$	SSO, HHO
Lei and Liu (2020)	$R \mid M_m \mid C_{max}$	ABC
Marinho Diana and de Souza (2020)	$R \mid r_j, s_{ijk} \mid \sum w_j T_j$	GA, ABC, IG, VND, MDDR
Orts et al. (2020)	$R \mid\mid C_{max}$	GA, MDDRs
X. Wang et al. (2020)	$R \mid rwrk \mid \sum w_j T_j$	GA, SA, MDDR

Ghaleb et al. (2020)	$R \mid M_m, brkdwn, M_d \mid COST$	SA
M.-Z. Wang et al. (2020)	$R \mid R \mid \sum F_j, COST$	PSO
Pinheiro et al. (2020)	$R \mid R \mid \sum T_j$	IG, SA
Bhardwaj et al. (2020)	$R \mid prec \mid \sum C_j$	ACO, MDDR
	$R \mid prec \mid C_{max}$	
Đurasevic and Jakobovic (2020)	$R \mid r_j \mid \sum w_j T_j$	ADDRs, MDDRs
Đurasević and Jakobović (2020)	$R \mid r_j \mid \sum w_j T_j$	GA, MDDRs, ADDRs
Pan et al. (2020)	$R \mid M_s \mid \sum T_j, TEC$	GA and DR
Ewees et al. (2021)	$R \mid s_{ijk} \mid C_{max}$	SSA, FA
Van and Hop (2021)	$R \mid s_{ijk} \mid \sum(w_j E_j + w_j T_j) + C_{max}$	MDDR, GA
Yepes-Borrero et al. (2021)	$R \mid s_{ijk}, R \mid C_{max}, R_u$	GA, IG
D.-Y. Lin and Huang (2021)	$R \mid s_{ijk}, M_j, brkdwn, r_j, R \mid C_{max} + COST$	SA, VNs
Al-harkan et al. (2021)	$R \mid s_{ijk}, r_j, R \mid C_{max}$	HS
Al-qaness et al. (2021)	$R \mid s_{ijk} \mid C_{max}$	WOA, FA
Nanthapodej et al. (2021a)	$R \mid M_j \mid C_{max} + TEC$	DE, VNS
Nanthapodej et al. (2021b)	$R \mid M_j \mid C_{max} + \sum w_j U_j + TEC$	VNS
Jovanovic and Voß (2021)	$R \mid s_{ijk} \mid C_{max}$	FSS
C.-Y. Cheng et al. (2021)	$R \mid s_{ijk} \mid C_{max}$	ABC, IG
L. Zhang et al. (2021)	$R \mid s_{ijk}, R \mid C_{max}, TEC$	CEA, MDDRs
Zarook, Yaser et al. (2021)	$R \mid r_j, p-batch, J_s, Q_k \mid C_{max}$	MDDRs, GA

Jaklinovic et al. (2021)	$R \mid r_j, s_{ij} \mid \sum w_j T_j$	MDDRs, ADDRs, GA
	$R \mid r_j, M_j \mid \sum w_j T_j$	
	$R \mid r_j, prec \mid \sum w_j T_j$	
	$R \mid r_j, brkdwn \mid \sum w_j T_j$	
	$R \mid r_j, s_{ij}, M_j \mid \sum w_j T_j$	
	$R \mid r_j, s_{ij}, prec \mid \sum w_j T_j$	
	$R \mid r_j, s_{ij}, brkdwn \mid \sum w_j T_j$	
	$R \mid r_j, M_j, brkdwn \mid \sum w_j T_j$	
	$R \mid r_j, M_j, prec \mid \sum w_j T_j$	
	$R \mid r_j, brkdwn, prec \mid \sum w_j T_j$	
	$R \mid r_j, s_{ij}, M_j, brkdwn \mid \sum w_j T_j$	
	$R \mid r_j, s_{ij}, M_j, prec \mid \sum w_j T_j$	
	$R \mid r_j, s_{ij}, prec, brkdwn \mid \sum w_j T_j$	
	$R \mid r_j, M_j, prec, brkdwn \mid \sum w_j T_j$	
	$R \mid r_j, s_{ij}, M_j, brkdwn, prec \mid \sum w_j T_j$	
Planinić, Đurasević, and Jakobović (2021)	$R \mid r_j \mid \sum w_j T_j$	ADDRs
Planinić, Đurasević, and Jakobović (2021)	$R \mid r_j \mid \sum w_j T_j$	ADDRs
Planinić et al. (2022)	$R \mid r_j \mid C_{max}$	ADDRs
	$R \mid r_j \mid \sum F_j$	
	$R \mid r_j \mid \sum w_j T_j$	
	$R \mid r_j \mid \sum w_j U_j$	
Ulaga et al. (2022)	$R \mid r_j \mid \sum w_j T_j$	ILS, VNS, GRASP, SA, ACO, TS
Đurasević and Jakobović (2022)	$R \mid r_j \mid \sum w_j T_j$	ADDRs

Finally, Table 5 represents a chronological overview of all the reviewed research for the UPMSP. For each study we provide an overview of the problems considered in the paper, as well as the solution methods that were considered for solving these problems. The chronological overview demonstrates how in the beginning most research focused on using DRs or heuristics. Although first studies applying metaheuristics for the considered problem appeared during the nineties, it was not until after year 2000 that they gained more popularity and started to dominate in the research. A similar thing can be observed also for the problem variants, where during the nineties problems with no constraints ans a single criterion were mostly considered. This trend continued in the early 2000s, but soon research considering additional constraints or multiple constraints started gaining even more attention.

5 Recent trends and future directions

5.1 Problem definitions

One important issue with the current research is the wide range of different problem instances that are used. This presents a quite large problem because it is difficult to compare the results across different studies. Quite often problem instances are generated, and although there are certain suggestions which are usually used for that purpose, there are no guidelines or rules that would specify how to perform this in order to make it as general as possible. Naturally, great obstacle here is the fact that there are many different problem types and variants. Several authors made their problem instances publicly available, which represents a good start in the direction of using a common problem set. A good example of this is the case of the $R \mid s_{ijk} \mid C_{max}$ problem instance sets defined by Rabadi et al. (2006) and Vallada and Ruiz (2011) that are often used for that problem variant. This enables that all new methods and findings are directly comparable to those of previous studies. However, these problem instances are usually designed for a general problem variant and cannot be used in all cases. Additionally, in most studies instances included up to 200 jobs and 20 machines. Larger instances should also be used to stress test the proposed methods and analyse their performance as the problem size increases.

An important step in this direction would be to define a common set of problem instances that could be used in layers. This would mean that all the information for different problem variants would be specified in the set. However, the researchers could then use only the subset which is of interest to them based on the problem variant they investigate. In this way, it would be possible to base all research on the same instances, and only use those instances that are relevant. In addition, this would also allow other researchers to provide extensions to existing datasets, and simply add additional constraints to an already present basis. It would also be imperative that the problem set includes instances generated in different ways (to ensure that the methods are not overspecialised on a very narrow problem type), and include instances of

different sizes to better test the generality and scalability. An additional benefit would also be that it would be possible to compare novel methods across a wider range of problem variants, which would demonstrate how general the proposed method is for solving different problem types. Unfortunately, such an investigation has been performed in only a few studies (Jaklinovic et al., 2021; Kramer & Subramanian, 2017).

Another, more easier direction would be to provide a library of all known instances available for the UPMSP. This library would then serve as the source from which relevant problem instances could be downloaded for available problem variants. The largest problem with such a solution is that not many studies have made their problem instances available, and it would not be possible to provide instances for all problem variants. For that reason, it is important that future studies in the UPMSP make the problem instances used for testing publicly available to stimulate the reusability of problem sets, rather than generating new ones. An additional problem with this approach is that existing problem instances use different formats to represent the data, which incurs overhead when trying to apply a method across a wider range of problem sets. Therefore, another important issue here would be to determine a common format that could be used by future studies.

5.2 Solution methods

This review demonstrated that a wide range of solution methods were proposed, which range from quite simple heuristics, to complicated hybrid metaheuristics which combine several methods into one. All those algorithms offer different benefits and drawbacks, and it is up to the user to determine the one that should be used, which is not always trivial.

DRs have proven to be quite useful, especially in cases when a solution needs to be obtained in a small amount of time, or not all information about the problem is available. However, designing DRs of good quality requires good domain knowledge about the problem, and is usually a time consuming task. Luckily, at least for the standard criteria and problem types, quite good DRs have been developed (Đurasevic & Jakobovic, 2018). However, as more exotic problem variants are considered, especially with more problems focusing on auxiliary resources and green scheduling, it is more probable that an appropriate DR does not exists. Therefore, several studies have investigated how to automatically generate DRs for the different variants of the UPMSP. These ADDRs demonstrated a good performance, and usually outperformed MDDRs in many occasions (Đurasevic et al., 2016). Further investigation in this area could lead to even better DRs that can be used for solving problems by themselves, or in combination with other methods. Unfortunately, in their performance DRs significantly lag behind metaheuristic methods (Balin, 2011; T.D. Braun et al., 2001; D.-W. Kim et al., 2003). As such, when dynamic problems are not considered, and the schedule does not need to be created in a small amount of time, metaheuristics represent a better choice for solving the problem. On the other hand, problem specific heuristics have been gaining

less and less attention in recent years, as they have been overshadowed by metaheuristic methods that are usually much easier to apply for different UPMSPs and achieve the same or even better performance (Perez-Gonzalez et al., 2019).

When considering metaheuristics, one of the first and most important decisions is which algorithm to select. Many algorithms have been applied for solving the UPMSP through the years. Most studies still apply standard metaheuristics algorithms like GA, TS, SA, VNS, and similar. However, several recent studies applied a plethora of novel metaheuristic methods like HHO (Jouhari et al., 2020), WO (J.-P. Arnaout, 2019), SSO (Lei & Liu, 2020), FSS (Jovanovic & Voß, 2021), WOA (Al-qaness et al., 2021), and similar. Recently, Arık (2019) did an interesting analysis on how different metaheuristic method types (swarm intelligence, evolutionary algorithm, single solution algorithm) perform on a considered problem. The authors tested only the basic algorithms with no problem specific adaptations, to identify which algorithm concepts are actually the most suitable for the UPMSP. Only three algorithms were considered, and as such a much more thorough analysis needs to be performed to obtain a better understanding of different methods. In most research standard algorithms (GA, TS, SA) were used, which is due to them being available for a longer time, and not necessarily because they are better. However, there have been cases where some more exotic metaheuristic algorithms were used in older studies, and regardless of their good performance for the considered problem, were not used in any further studies. Examples of these are SWO (Piersma & van Dijk, 1996) and RRT (J.-F. Chen, 2013). In the end, only time will tell which of the more recently applied metaheuristic methods will stand the test of time and become a relevant method for solving the UPMSP.

However, it should be outlined that such new metaheuristics have often been criticised as they usually do not provide enough novelty or function similar as existing ones (Sörensen, 2013). In addition, there is also a line of research which considers hybrid metaheuristics that combine concepts of several metaheuristic methods into one. Many hybrid methods that combine operators from different metaheuristic algorithms have been recently applied for solving the UPMSP, like combining the concepts of GA and PSO (Salehi Mir & Rezaeian, 2016). Several recent studies propose hybrid methods that, based on the fitness value of the individual, apply operators of either method without any real hybridisation (Al-qaness et al., 2021; Ewees et al., 2021; Jouhari et al., 2020), which means that the two methods are mostly applied separately. Although there is merit in such research directions, the application of hybrid or novel algorithms should be done sparingly and examined in detail to demonstrate their effectiveness compared to existing methods. In this case, a common dataset consisting of several problems is required to test how effective these novel approaches are, since due to the no free lunch theorem it will always be possible to find a set of instances on which an algorithm will be more effective than others (Wolpert & Macready, 1997). Therefore, the goal of future

research should not be to find the ideal metaheuristic for each specific UPMSP type, but rather to design methods which perform well over a wide range of problem types and gain better understanding which concepts are important in such algorithms. Unfortunately, most studies that propose hybrid methods are applied on only one problem type, and it is generally not known how they would perform when considering other problem types.

An alternative research direction could be something similar to what was investigated by Kramer and Subramanian (2017), where the authors proposed a unified heuristic that could be applied for a wider range of scheduling problems. Such a research direction presents a direct opposite of the previous research in which more specific methods are designed to solve a particular problem variant. Unfortunately, this direction was not further investigated in any follow up studies. It would be interesting to pursue this research in order to get a notion on how general such methods could be made, and also to analyse how they compare to the metaheuristics specifically adapted for certain problem variants, in order to demonstrate the gap between both approaches.

Many studies demonstrated that individual methods are usually not powerful enough to effectively solve the considered problem (Jolai et al., 2009; Kayvanfar & Teymourian, 2014). Therefore, a lot of research tried to extend standard metaheuristic methods with additional domain knowledge. This was mostly realised in two ways. One way was to apply DRs in initialisation of solutions (Vlašic et al., 2019), or in some algorithm operators when constructing the solution (Costa et al., 2013). The second approach was to include different problem specific LS operators in metaheuristic methods (C.-J. Liao et al., 2016; Vallada & Ruiz, 2011; M.-Z. Wang et al., 2020). Both approaches have proven to be quite effective. However, when combining such methods, several design choices need to be performed in the algorithms, for example which LS operators to use, when to apply them in the algorithm and similar. Some studies have already tried to provide answers to such questions (Ulaga et al., 2022), but still a lot of design choices which need to be performed. An interesting research direction would be to design methods that could by themselves learn which operators to apply in which situations, and relieve the designer of that choice (Cota et al., 2017; X. Wu & Che, 2019).

Aside from the solution method, another important element which needs to be specified is the solution representation that will be used to solve the considered problem. Over the years, many solution representations have been used, ranging from permutation (Costa et al., 2013), matrix (Balin, 2011), and real number (Eroglu et al., 2014) representations. Other studies also used encodings that can be considered nonstandard for the UNPMSP (J.-P. Arnaout et al., 2009). Furthermore, some studies, usually considering more complex problem variants, also propose hybrid encoding schemes that use simple heuristics to decode the solution (Costa et al., 2013; S. Lu et al., 2018). Several studies demonstrated the importance of the right choice for the encoding (T. Liao et al., 2014; Vlašic et al., 2019), however, more research will be required in this area. Especially as problems will include more constraints,

since in such cases encoding feasible solutions becomes more challenging and quite often the solution is only partially encoded. However, many studies demonstrated that such partial representations limit the performance of metaheuristics, and as such there is a need to extend the research in this are and examine whether better solution encoding schemes can be proposed (Afzalirad & Shafipour, 2015; Costa et al., 2013).

5.3 Problem types

The reviewed research demonstrated that over the years a great deal of problem variants were considered, both from the perspective of the optimisation criterion, as well the additional constraints that are used.

During the years, a plethora of constraints have been defined and investigated for the UPMSP. At the beginning, most studies focused on problems that included no constraints, or where only one constraint was considered. The most researched problem type in that regard would be the one including setup times, as it is commonly used by researchers to propose novel algorithms and compare to other existing research. Since this problem variant is used as a certain benchmark, it is likely that further research will be performed on it. Some constraints have been investigated by some studies, but did not gain a wider attention over the years. These constraints include job families (Klemmt et al., 2009), dedicated machines (C.-Y. Cheng & Huang, 2017), or loading times for transporting jobs to machines (Bilyk & Mönch, 2010). As such, it is very likely that these constraints will not be studied significantly in future studies. However, in recent years several other constraints are becoming more popular. For example, batch scheduling problems are gaining attention since they appear in many industrial problems (S. Lu et al., 2018; Zarook, Yaser et al., 2021). Furthermore, resource based constraints are also becoming more widely researched to better model real world environments in which various resources (human or material) are required to perform actions and execute jobs (Pinheiro et al., 2020; Yepes-Borrero et al., 2021). Both of these problem variants add to the complexity of the UPMSP, as they introduce additional decisions that need to be performed during the construction for the schedule (resource allocation, or grouping jobs). Since these problem variants have only recently been gaining a larger attention, further research will probably be focused on combining these constraints with others, or focusing on some special properties for each constraint, like considering different renewable and non renewable resource types, or problems with different job and batch sizes in batch scheduling. Furthermore, an increasing number of studies in recent years tackle problems which include several constraints (Afzalirad & Rezaeian, 2017; Jaklinovic et al., 2021; D.-Y. Lin & Huang, 2021; Shahvari & Logendran, 2017a), which is a trend that should continue as it becomes more important to tackle complex real world problems.

Regarding the optimisation criteria, until now standard optimisation criteria, like the makespan or total weighted tardiness, have been most often

considered. However, certain standard criteria like the maximum flowtime or maximum tardiness have rarely been considered, which seems to indicate that practical applications for such criteria might be scarce. Some studies dealt with non standard criteria like total late time (Afzalirad & Rezaeian, 2016a) or total setup time (Strohhecker et al., 2016) Recent years also saw the rise of new criteria that are considered, most notably production cost (tied with resource constraints) (Ghaleb et al., 2020; Shahvari & Logendran, 2017a) and energy consumption (tied with green scheduling problems) (H. Lu & Qiao, 2017; L. Zhang et al., 2021). However, it makes no sense to consider these criteria by themselves, therefore they are usually considered together with other standard scheduling criteria, which results in MO problem variants (X. Wu & Che, 2019; L. Zhang et al., 2021). Even more, these criteria are usually quite conflicting with other scheduling criteria, which results in difficult MO problems. Although, most of the research until now was done done considering only a single objective, more and more recent research in this area is putting focus on MO optimisation and recognising its importance (Lei, Yuan, & Cai, 2020; Pan et al., 2020; X. Wu & Che, 2019). Therefore, it is expected that in the future a larger focus will be put MO problems and designing methods that can effectively tackle such problems.

5.4 Dynamic and stochastic problems

Until now most problems were deterministic and static, which means that all the system parameters were known beforehand, and the schedule could be constructed up front and then simply executed at a later point in time. Unfortunately, in real world problems all information about the problem are not known up front, or unforeseen situations happen during the execution of the schedule (Ouelhadj & Petrovic, 2008). In most cases, existing studies have considered either stochastic system properties like processing times J.-P.M. Arnaout et al. (2006), machine availability periods (SURESH & GHAUDHURI, 1996), dynamic job arrivals (Đurasevic & Jakobovic, 2018) or the need for rework processes (X. Wang et al., 2020). As a result, the schedules obtained by metaheuristics can become invalid if unexpected situations like these happen, and might need to be changed or adapted. Therefore, it is required to design methods that build schedules incrementally and can this quickly react to changes in the system, or methods which when generate a schedule have the possibility of correcting it and in such a way to adapt the schedule for the unexpected situations. DRs are mostly easily applied to such a problem since they incrementally build the schedule (Đurasevic & Jakobovic, 2018). As such, they have often been used with dynamic problems in which jobs are released over time, but can be used with any dynamic or stochastic problem variant without much need for adaptation. However, since the DRs construct the schedule iteratively, the quality of the schedule they construct is limited. Therefore, one research direction is to improve the performance of DRs that they have a better overview on the problem and to reduce the gap between their results and those of metaheuristics. On the other hand,

metaheuristic methods provide solutions of a better quality, however, it is more difficult to apply them for dynamic scheduling problems. Some studies tried to apply metaheuristics for problems which did include certain uncertainties (mostly concerned with rework processes (X. Wang et al., 2020) or using fuzzy sets to represent uncertainties of some properties (Peng & Liu, 2004; Torabi et al., 2013)), but but such research is still quite sparse. Therefore, an important research direction would be to put more focus on solving dynamic and stochastic scheduling problems, and adapt metaheuristics for unforeseen situations that can happen during the system execution. For example, one possibility would be to define correction methods which could be applied to correct results obtained by metaheuristics given an unexpected situation.

5.5 Application for real world problems

The most important part of every problem is its connection to the real world, and the possibility of applying the proposed methods for problems and in real environments. Therefore, a great deal of studies found motivation for considering certain problems from the real world, and then modelled such real world problems as UPMSPs with different constraints. Examples of such problems include manufacturing of Polyvinyl Chloride pipes (C.-H. Lee et al., 2014; Salehi Mir & Rezaeian, 2016), CNC manufacturing cells (Celano et al., 2008), distributing jobs over heterogeneous processing units (Orts et al., 2020), manufacturing in an electronic plant (Arroyo & Leung, 2017a; Jou, 2005), scheduling in a textile industry (Silva & Magalhaes, 2006), scheduling jobs in grid computing systems (Tseng et al., 2009), and others (Bitar et al., 2014; T. Liao et al., 2014; Van Hop & Nagarur, 2004). Some studies even go a step further, and demonstrate the performance of the proposed methods on data from real world environments, and compare those results to existing methods or schedules. For example, Jou (2005) examine the proposed methods on a problem of scheduling in the electronic plant. Such applications to real world data sets provides several benefits, some of which are the demonstration that the proposed methods can be applied for real complex problems, whereas the second is that such results demonstrate that the proposed methods can provide an improvement in the scheduling processes that are considered.

An important thing that needs to be considered is that the methods that are proposed can be applied for real world problems. Real world problems usually require that not only a single objective is optimised, but rather that several objectives are optimised simultaneously (Taboada & Coit, 2008). In that case a solution which represents a trade off between all objectives needs to be obtained. For example, MO is often considered in problems which deal with green scheduling Pan et al. (2020); L. Zhang et al. (2021), or with problems where additional resources need to be used and their usage needs to be minimised Shahvari and Logendran (2017a); Yepes-Borrero et al. (2021), and problems modelled based on real world scheduling problems from shipyards Afzalirad and Rezaeian (2017) or the electronics assembly industry (D.-Y. Lin & Huang, 2021). However, as was denoted in the literature review, only a

handful of studies focused on MO problems, and a lot of studies used a weighted sum of several objectives. Therefore, it is imperative to put more focus on pore MO problems. However, it is very likely that this direction will get a lot attention in the future since MO optimisation is gaining more momentum across different optimisation problems, and as such it is expected that research in the UPMSP will follow this trend. Additionally, most of the studies dealing with MO optimisation were published in recent years, which suggest that such a trend could continue further.

Apart from multiple objectives, another important thing is that real world problems usually are quite complicated and include many different and specific constraints. Especially in the recent years, batch scheduling problems and problems with auxiliary resources have commonly been investigated as they appear in many real world scenarios. For example, batch scheduling found its use in wafer or cell fabrication facilities (Celano et al., 2008; Klemmt et al., 2009), whereas resource constrained scheduling took its motivation from worker allocation in production (L. Zhang et al., 2021) to steel production and photolithography workshops (Bitar et al., 2014; Pinheiro et al., 2020). However, there are also other constraints that are maybe less often investigated, but are still important in real world scenarios. One example are machine availability periods, which can occur either due to machine breakdowns or scheduled maintenance periods where it is often required to determine when these availability periods will occur (Ghaleb et al., 2020). In future the previously outlined constraints will be examined in more detail, but it is also likely that more complex variants will be considered to better model real world situations.

5.6 Green scheduling

Green manufacturing has become an increasingly important paradigm in recent years, which aims to reduce the impact of the industrial production on the environment (Ahemad & Shrivastava, 2013). In that regard, scheduling also plays an important role as it directly influences the production environment (Akbar & Irohara, 2018). Therefore, during the construction of the schedules it is not only required to optimise the main objective, but also to focus on its impact on the environment.

In the UPMSP the research until now considered the effect on the environment through energy consumption that needed to be minimised together with another criterion. Therefore, all studies focused on a MO problem in which in addition to a standard scheduling criterion the energy consumed during the execution of the schedule is also minimised. However, the basic scheduling model had to be extended in order to consider energy consumption, which has until now been done in several ways. In the first model the machines have different states, for example that can be idle, working, or turned off and in each state the machines consume a different amount of energy (Z. Li et al., 2015; Liang et al., 2015). The second model considered that each job also consumes a certain amount of energy during execution, which depends on the machine on which it executes Che et al. (2017); Nanthapodej et al.

(2021a, 2021b) Finally, in the last model the machines have different operating speeds, with higher speeds consuming more energy (Pan et al., 2020; Zheng & Wang, 2018).

It can be seen that until now only a smaller number of recent studies dealt with this problem type. However, in other similar problems like VRP, green problem variants have already become quite extensively researched (Erdelić & Carić, 2019; Kucukoglu, Dewil, & Cattrysse, 2021; Moghdani, Salimifard, Demir, & Benyettou, 2021). It is expected that with time more focus will also be shifted towards green manufacturing problems. Therefore, it is likely that more constraints and problem variants will be introduced and will have to be considered during scheduling. Very likely the models that simulate the impact of such problems on the environment will become more realistic because of which it will be required to consider more complex models. Another important property of green scheduling problems is that they are MO problems by nature (Nanthapodej et al., 2021b; Pan et al., 2018), as it only makes sense to optimise the energy consumption together with other criteria. This again increases the complexity of such problem variants over standard scheduling problem variants, as MO algorithms need to be applied.

6 Conclusion

This study provides the first exhaustive review on the application of heuristic and metaheuristic methods for the UPMSP, which covers around 250 research papers. The main contributions of this survey is that each study is briefly described and classified according to the considered problem instances as well as the applied solution methods. An extensive overview of heuristic methods that are applied for solving the UPMSP is provided, with the focus equally set to both on problem specific heuristics and metaheuristics, since in a lot of research these methods are used in synergy to increment each other. In addition, an overview of different scheduling problem variants was provided. Furthermore, the study outlines relations between similar research in order to denote the evolution of this research field and show in which directions the research is heading. Finally, the survey provides a brief overview of recent research trends and outlines potential future research directions and problem variants that could prove to be interesting for future research.

The review shows that this problem is tackled with a plethora of different methods, and that during the years several problem variants and optimisation criteria were considered. This has become especially evident in the last years, since around 70% of research in this area has been published in the last 10 years, and around 40% in the last 5 years. This shows that the interest for this problem is increasing, and the number of publications in the recent years show that such a trend could continue even further and even more studies could focus on this problem. As such, it is important to have a good overview of existing research to diminish the possibility of repeating research, but also to outline potential future directions.

Although the research in this area is already extensive, there are still several ways in which it can be improved. Recent research trends show a significant number of studies being directed towards new problem variants which are either concerned with green scheduling, additional resources, MO optimisation and many other. It is expected that these problem variants will receive an even wider attention in further years. Furthermore, most research focused on static problems, but since system parameters change frequently in the real world, a larger shift towards dynamic and stochastic problem variants is required as well. Finally, A lot of studies proposed novel or hybrid algorithms for different problem types, but only a few studies tried to provide a general method for a wider range of problem variants. As more problem variants are introduced, the need for such a general method also becomes even more important as it would not require the adaptation for every new problem variant. All things considered, the area of UPMSP is currently receiving a lot of attention which should in the near future certainly result in new interesting studies that deal with novel and real world problem instances and provide new findings.

Declarations

Funding

This work has been supported in part by Croatian Science Foundation under the project IP-2019-04-4333.

Conflicts of interest/Competing interests

Not applicable

Availability of data and material

Not applicable

Code availability

Not applicable

References

Abedi, M., Seidgar, H., Fazlollahtabar, H. (2017, Jan 01). Hybrid scheduling and maintenance problem using artificial neural network based meta-heuristics. *Journal of Modelling in Management*, *12*(3), 525-550. Retrieved from https://doi.org/10.1108/JM2-02-2016-0011

10.1108/JM2-02-2016-0011

Afzalirad, M., & Rezaeian, J. (2016a). Design of high-performing hybrid meta-heuristics for unrelated parallel machine

scheduling with machine eligibility and precedence constraints. *Engineering Optimization*, *48*(4), 706-726. Retrieved from https://doi.org/10.1080/0305215X.2015.1042475 https://arxiv.org/abs/https://doi.org/10.1080/0305215X.2015.1042475 10.1080/0305215X.2015.1042475

Afzalirad, M., & Rezaeian, J. (2016b). Resource-constrained unrelated parallel machine scheduling problem with sequence dependent setup times, precedence constraints and machine eligibility restrictions. *Computers & Industrial Engineering*, *98*, 40-52. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835216301619 https://doi.org/10.1016/j.cie.2016.05.020

Afzalirad, M., & Rezaeian, J. (2017). A realistic variant of bi-objective unrelated parallel machine scheduling problem: Nsga-ii and moaco approaches. *Applied Soft Computing*, *50*, 109-123. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494616305634 https://doi.org/10.1016/j.asoc.2016.10.039

Afzalirad, M., & Shafipour, M. (2015, June). Design of an efficient genetic algorithm for resource-constrained unrelated parallel machine scheduling problem with machine eligibility restrictions. *Journal of Intelligent Manufacturing*, *29*(2), 423–437. Retrieved from https://doi.org/10.1007/s10845-015-1117-6 10.1007/s10845-015-1117-6

Ahemad, M., & Shrivastava, R. (2013, 01). Green manufacturing (gm): Past, present and future(a state of art review). *World Review of Science*, *10*, 17 - 55. 10.1504/WRSTSD.2013.050784

Akbar, M., & Irohara, T. (2018). Scheduling for sustainable manufacturing: A review. *Journal of Cleaner Production*, *205*, 866-883. Retrieved from https://www.sciencedirect.com/science/article/pii/S0959652618328208 https://doi.org/10.1016/j.jclepro.2018.09.100

Al-harkan, I.M., & Qamhan, A.A. (2019). Optimize unrelated parallel machines scheduling problems with multiple limited additional resources,

sequence-dependent setup times and release date constraints. *IEEE Access*, *7*, 171533-171547.

10.1109/ACCESS.2019.2955975

Al-harkan, I.M., Qamhan, A.A., Badwelan, A., Alsamhan, A., Hidri, L. (2021). Modified harmony search algorithm for resource-constrained parallel machine scheduling problem with release dates and sequence-dependent setup times. *Processes*, *9*(4). Retrieved from https://www.mdpi.com/2227-9717/9/4/654

10.3390/pr9040654

Allahverdi, A. (2015). The third comprehensive survey on scheduling problems with setup times/costs. *European Journal of Operational Research*, *246*(2), 345-378. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221715002763

https://doi.org/10.1016/j.ejor.2015.04.004

Allahverdi, A. (2016). A survey of scheduling problems with no-wait in process. *European Journal of Operational Research*, *255*(3), 665-686. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221716303733

https://doi.org/10.1016/j.ejor.2016.05.036

Allahverdi, A., Gupta, J.N., Aldowaisan, T. (1999). A review of scheduling research involving setup considerations. *Omega*, *27*(2), 219-239. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305048398000425

https://doi.org/10.1016/S0305-0483(98)00042-5

Allahverdi, A., Ng, C., Cheng, T., Kovalyov, M.Y. (2008). A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, *187*(3), 985-1032. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221706008174

https://doi.org/10.1016/j.ejor.2006.06.060

Al-qaness, M.A.A., Ewees, A.A., Elaziz, M.A. (2021, May). Modified whale optimization algorithm for solving unrelated parallel machine scheduling problems. *Soft Computing*, *25*(14), 9545–9557. Retrieved from https://doi.org/10.1007/s00500-021-05889-w

10.1007/s00500-021-05889-w

Al-Salem, A., & Armacost, R. (2002, 01). Unrelated machines scheduling with machine eligibility restrictions. , *15*, 193-210.

Anagnostopoulos, G., & Rabadi, G. (2002). A simulated annealing algorithm for the unrelated parallel machine scheduling problem. *Proceedings of the 5th biannual world automation congress* (Vol. 14, p. 115-120). 10.1109/WAC.2002.1049430

Arık, O.A. (2019, October). Comparisons of metaheuristic algorithms for unrelated parallel machine weighted earliness/tardiness scheduling problems. *Evolutionary Intelligence*, *13*(3), 415–425. Retrieved from https://doi.org/10.1007/s12065-019-00305-7

10.1007/s12065-019-00305-7

Arnaout, J.-P. (2019, February). A worm optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times. *Annals of Operations Research*, *285*(1-2), 273–293. Retrieved from https://doi.org/10.1007/s10479-019-03138-w

10.1007/s10479-019-03138-w

Arnaout, J.-P., Musa, R., Rabadi, G. (2008). Ant colony optimization algorithm to parallel machine scheduling problem with setups. *2008 ieee international conference on automation science and engineering* (p. 578-582). 10.1109/COASE.2008.4626566

Arnaout, J.-P., Musa, R., Rabadi, G. (2012, June). A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines—part II: enhancements and experimentations. *Journal of Intelligent Manufacturing*, *25*(1), 43–53. Retrieved from https://doi.org/10.1007/s10845-012-0672-3

10.1007/s10845-012-0672-3

Arnaout, J.-P., & Rabadi, G. (2005). Minimizing the total weighted completion time on unrelated parallel machines with stochastic times. *Proceedings of the winter simulation conference, 2005.* (p. 7 pp.-). 10.1109/WSC.2005.1574499

Arnaout, J.-P., Rabadi, G., Musa, R. (2009, February). A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times. *Journal of Intelligent Manufacturing*, *21*(6), 693–701. Retrieved from https://doi.org/10.1007/s10845-009-0246-1

10.1007/s10845-009-0246-1

Arnaout, J.-P.M., Rabadi, G., Mun, J.H. (2006). A dynamic heuristic for the stochastic unrelated parallel machine scheduling problem..

Arroyo, J.E.C., & Leung, J.Y.-T. (2017a). An effective iterated greedy algorithm for scheduling unrelated parallel batch machines with non-identical capacities and unequal ready times. *Computers & Industrial Engineering*, *105*, 84-100. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835216305162

https://doi.org/10.1016/j.cie.2016.12.038

Arroyo, J.E.C., & Leung, J.Y.-T. (2017b). Scheduling unrelated parallel batch processing machines with non-identical job sizes and unequal ready times. *Computers & Operations Research*, *78*, 117-128. Retrieved from https://www.sciencedirect.com/science/article/pii/S030505481630209X

https://doi.org/10.1016/j.cor.2016.08.015

Arroyo, J.E.C., Leung, J.Y.-T., Tavares, R.G. (2019, January). An iterated greedy algorithm for total flow time minimization in unrelated parallel batch machines with unequal job release times. *Engineering Applications of Artificial Intelligence*, *77*, 239–254. Retrieved from https://doi.org/10.1016/j.engappai.2018.10.012

10.1016/j.engappai.2018.10.012

Avalos-Rosales, O., Angel-Bello, F., Alvarez, A. (2014, September). Efficient metaheuristic algorithm and re-formulations for the unrelated parallel machine scheduling problem with sequence and machine-dependent setup times. *The International Journal of Advanced Manufacturing Technology*, *76*(9-12), 1705–1718. Retrieved from https://doi.org/10.1007/s00170-014-6390-6

10.1007/s00170-014-6390-6

Avalos-Rosales, O., Angel-Bello, F., Alvarez, A., Cardona-Valdes, Y. (2018). Including preventive maintenance activities in an unrelated parallel machine environment with dependent setup times. *Computers & Industrial Engineering*, *123*, 364-377. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835218303218

https://doi.org/10.1016/j.cie.2018.07.006

Babaei, H., Karimpour, J., Hadidi, A. (2015). A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering*, *86*, 43-59. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835214003714 (Applications of Computational Intelligence and Fuzzy Logic to Manufacturing and Service Systems)

https://doi.org/10.1016/j.cie.2014.11.010

Balin, S. (2011). Non-identical parallel machine scheduling using genetic algorithm. *Expert Systems with Applications*, *38*(6), 6814-6821. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417410014272

https://doi.org/10.1016/j.eswa.2010.12.064

Bandyopadhyay, S., & Bhattacharya, R. (2013). Solving multi-objective parallel machine scheduling problem by a modified nsga-ii. *Applied Mathematical Modelling*, *37*(10), 6718-6729. Retrieved from https://www.sciencedirect.com/science/article/pii/S0307904X13001200

https://doi.org/10.1016/j.apm.2013.01.050

Bank, J., & Werner, F. (2001, February). Heuristic algorithms for unrelated parallel machine scheduling with a common due date, release dates, and linear earliness and tardiness penalties. *Mathematical and Computer Modelling*, *33*(4-5), 363–383. Retrieved from https://doi.org/10.1016/s0895-7177(00)00250-8

10.1016/s0895-7177(00)00250-8

Bektur, G., & Sarac, T. (2019). A mathematical model and heuristic algorithms for an unrelated parallel machine scheduling problem with sequence-dependent setup times, machine eligibility restrictions and a common server. *Computers*

& Operations Research, *103*, 46-63. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054818302697

https://doi.org/10.1016/j.cor.2018.10.010

Bhardwaj, A.K., Gajpal, Y., Surti, C., Gill, S.S. (2020, September). HEART : Unrelated parallel machines problem with precedence constraints for task scheduling in cloud computing using heuristic and meta-heuristic algorithms. *Software: Practice and Experience*, *50*(12), 2231–2251. Retrieved from https://doi.org/10.1002/spe.2890

10.1002/spe.2890

Bilyk, A., & Mönch, L. (2010, October). A variable neighborhood search approach for planning and scheduling of jobs on unrelated parallel machines. *Journal of Intelligent Manufacturing*, *23*(5), 1621–1635. Retrieved from https://doi.org/10.1007/s10845-010-0464-6

10.1007/s10845-010-0464-6

Bitar, A., Dauzère-Pérès, S., Yugma, C., Roussel, R. (2014, October). A memetic algorithm to solve an unrelated parallel machine scheduling problem with auxiliary resources in semiconductor manufacturing. *Journal of Scheduling*, *19*(4), 367–376. Retrieved from https://doi.org/10.1007/s10951-014-0397-6

10.1007/s10951-014-0397-6

Bozorgirad, M.A., & Logendran, R. (2012). Sequence-dependent group scheduling problem on unrelated-parallel machines. *Expert Systems with Applications*, *39*(10), 9021-9030. Retrieved from https://www.sciencedirect.com/science/article/pii/S095741741200276X

https://doi.org/10.1016/j.eswa.2012.02.032

Branke, J., Nguyen, S., Pickardt, C.W., Zhang, M. (2016). Automated design of production scheduling heuristics: A review. *IEEE Transactions on Evolutionary Computation*, *20*(1), 110-124.

10.1109/TEVC.2015.2429314

Braun, T., Siegal, H., Beck, N., Boloni, L., Maheswaran, M., Reuther, A., ... Freund, R. (1999). A comparison study of static mapping heuristics for a class of meta-tasks on heterogeneous computing systems. *Proceedings.*

eighth heterogeneous computing workshop (hcw'99) (p. 15-29). 10.1109/HCW.1999.765093

Braun, T.D., Siegel, H.J., Beck, N., Bölöni, L.L., Maheswaran, M., Reuther, A.I., ... Freund, R.F. (2001). A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computing*, *61*(6), 810-837. Retrieved from https://www.sciencedirect.com/science/article/pii/S0743731500917143

https://doi.org/10.1006/jpdc.2000.1714

Briceño, L.D., Siegel, H.J., Maciejewski, A.A., Oltikar, M. (2012, March). Characterization of the iterative application of makespan heuristics on non-makespan machines in a heterogeneous parallel and distributed environment. *The Journal of Supercomputing*, *62*(1), 461–485. Retrieved from https://doi.org/10.1007/s11227-011-0729-7

10.1007/s11227-011-0729-7

Caniyilmaz, E., Benli, B., Ilkay, M.S. (2014, November). An artificial bee colony algorithm approach for unrelated parallel machine scheduling with processing set restrictions, job sequence-dependent setup times, and due date. *The International Journal of Advanced Manufacturing Technology*, *77*(9-12), 2105–2115. Retrieved from https://doi.org/10.1007/s00170-014-6614-9

10.1007/s00170-014-6614-9

Cao, D., Chen, M., Wan, G. (2005). Parallel machine selection and job scheduling to minimize machine cost and job tardiness. *Computers & Operations Research*, *32*(8), 1995-2012. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054804000036

https://doi.org/10.1016/j.cor.2004.01.001

Cappadonna, F., Costa, A., Fichera, S. (2013). Makespan minimization of unrelated parallel machines with limited human resources. *Procedia CIRP*, *12*, 450-455. Retrieved from https://www.sciencedirect.com/science/article/pii/S221282711300718X (Eighth CIRP Conference on Intelligent Computation in Manufacturing Engineering)

https://doi.org/10.1016/j.procir.2013.09.077

Cappadonna., F.A., Costa., A., Fichera., S. (2012). Three genetic algorithm approaches to the unrelated parallel machine scheduling problem with limited human resources. *Proceedings of the 4th international joint conference on computational intelligence - ecta, (ijcci 2012)* (p. 170-175). SciTePress. 10.5220/0004116501700175

Celano, G., Costa, A., Fichera, S. (2008). Scheduling of unrelated parallel manufacturing cells with limited human resources. *International Journal of Production Research*, *46*(2), 405-427. Retrieved from https://doi.org/10.1080/00207540601138452 https://arxiv.org/abs/ https://doi.org/10.1080/00207540601138452
10.1080/00207540601138452

Chang, P.-C., & Chen, S.-H. (2011). Integrating dominance properties with genetic algorithms for parallel machine scheduling problems with setup times. *Applied Soft Computing*, *11*(1), 1263-1274. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494610000694

https://doi.org/10.1016/j.asoc.2010.03.003

Chang, Y.-C., Li, V.C., Chiang, C.-J. (2014). An ant colony optimization heuristic for an integrated production and distribution scheduling problem. *Engineering Optimization*, *46*(4), 503-520. Retrieved from https://doi.org/10.1080/0305215X.2013.786062 https://arxiv.org/abs/ https://doi.org/10.1080/0305215X.2013.786062
10.1080/0305215X.2013.786062

Charalambous, C., Fleszar, K., Hindi, K.S. (2010). A hybrid searching method for the unrelated parallel machine scheduling problem. H. Papadopoulos, A.S. Andreou, & M. Bramer (Eds.), *Artificial intelligence applications and innovations* (pp. 230–237). Berlin, Heidelberg: Springer Berlin Heidelberg.

Chaudhry, I.A., & Khan, A.A. (2015, August). A research survey: review of flexible job shop scheduling techniques. *International Transactions in Operational Research*, *23*(3), 551–591. Retrieved from https://doi.org/10.1111/itor.12199

10.1111/itor.12199

Che, A., Zhang, S., Wu, X. (2017). Energy-conscious unrelated parallel machine scheduling under time-of-use electricity tariffs. *Journal of Cleaner Production*, *156*, 688-697. Retrieved from https://www.sciencedirect.com/science/article/pii/S0959652617307175

https://doi.org/10.1016/j.jclepro.2017.04.018

Chen, B., Potts, C.N., Woeginger, G.J. (1998). A review of machine scheduling: Complexity, algorithms and approximability. *Handbook of combinatorial optimization* (pp. 1493–1641). Springer US. Retrieved from https://doi.org/10.1007/978-1-4613-0303-9_25 10.1007/978-1-4613-0303-9_25

Chen, C.-L. (2008). An iterated local search for unrelated parallel machines problem with unequal ready times. *2008 ieee international conference on automation and logistics* (p. 2044-2047). 10.1109/ICAL.2008.4636499

Chen, C.-L. (2011, September). Iterated hybrid metaheuristic algorithms for unrelated parallel machines problem with unequal ready times and sequence-dependent setup times. *The International Journal of Advanced Manufacturing Technology*, *60*(5-8), 693–705. Retrieved from https://doi.org/10.1007/s00170-011-3623-9

10.1007/s00170-011-3623-9

Chen, C.-L., & Chen, C.-L. (2008, August). Hybrid metaheuristics for unrelated parallel machine scheduling with sequence-dependent setup times. *The International Journal of Advanced Manufacturing Technology*, *43*(1-2), 161–169. Retrieved from https://doi.org/10.1007/s00170-008-1692-1

10.1007/s00170-008-1692-1

Chen, J.-F. (2004, September). Unrelated parallel machine scheduling with secondary resource constraints. *The International Journal of Advanced Manufacturing Technology*, *26*(3), 285–292. Retrieved from https://doi.org/10.1007/s00170-003-1622-1

10.1007/s00170-003-1622-1

Chen, J.-F. (2006, September). Minimization of maximum tardiness on unrelated parallel machines with process restrictions and setups. *The International Journal of Advanced Manufacturing Technology*, *29*(5-6), 557–563. Retrieved from https://doi.org/10.1007/bf02729109

10.1007/bf02729109

Chen, J.-F. (2009, February). Scheduling on unrelated parallel machines with sequence- and machine-dependent setup times and due-date constraints. *The International Journal of Advanced Manufacturing Technology*, *44*(11-12), 1204–1212. Retrieved from https://doi.org/10.1007/s00170-008-1917-3

10.1007/s00170-008-1917-3

Chen, J.-F. (2013, October). Unrelated parallel-machine scheduling to minimize total weighted completion time. *Journal of Intelligent Manufacturing*, *26*(6), 1099–1112. Retrieved from https://doi.org/10.1007/s10845-013-0842-y

10.1007/s10845-013-0842-y

Chen, J.-F., & Wu, T.-H. (2006). Total tardiness minimization on unrelated parallel machine scheduling with auxiliary equipment constraints. *Omega*, *34*(1), 81-89. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305048304001197

https://doi.org/10.1016/j.omega.2004.07.023

Cheng, C.-Y., & Huang, L.-W. (2017). Minimizing total earliness and tardiness through unrelated parallel machine scheduling using distributed release time control. *Journal of Manufacturing Systems*, *42*, 1-10. Retrieved from https://www.sciencedirect.com/science/article/pii/S027861251630070X

https://doi.org/10.1016/j.jmsy.2016.10.005

Cheng, C.-Y., Pourhejazy, P., Ying, K.-C., Lin, C.-F. (2021). Unsupervised learning-based artificial bee colony for minimizing non-value-adding operations. *Applied Soft Computing*, *105*, 107280. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494621002039

https://doi.org/10.1016/j.asoc.2021.107280

Cheng, T., & Sin, C. (1990). A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research*, *47*(3), 271-292. Retrieved from https://www.sciencedirect.com/science/article/pii/037722179090215W

https://doi.org/10.1016/0377-2217(90)90215-W

Chyu, C.-C., & Chang, W.-S. (2009, November). A pareto evolutionary algorithm approach to bi-objective unrelated parallel machine scheduling problems. *The International Journal of Advanced Manufacturing Technology*, *49*(5-8), 697–708. Retrieved from

https://doi.org/10.1007/s00170-009-2419-7

10.1007/s00170-009-2419-7

Chyu, C.-C., & Chang, W.-S. (2010). A competitive evolution strategy memetic algorithm for unrelated parallel machine scheduling to minimize total weighted tardiness and flow time. *The 40th international conference on computers indutrial engineering* (p. 1-6). 10.1109/ICCIE.2010.5668388

Cochran, J.K., Horng, S.-M., Fowler, J.W. (2003). A multi-population genetic algorithm to solve multi-objective scheduling problems for parallel machines. *Computers & Operations Research*, *30*(7), 1087-1102. Retrieved from https://www.sciencedirect.com/science/article/pii/S030505480200059X

https://doi.org/10.1016/S0305-0548(02)00059-X

Costa, A., Cappadonna, F.A., Fichera, S. (2013, August). A hybrid genetic algorithm for job sequencing and worker allocation in parallel unrelated machines with sequence-dependent setup times. *The International Journal of Advanced Manufacturing Technology*, *69*(9-12), 2799–2817. Retrieved from https://doi.org/10.1007/s00170-013-5221-5

10.1007/s00170-013-5221-5

Cota, L.P., Guimarães, F.G., de Oliveira, F.B., Freitas Souza, M.J. (2017). An adaptive large neighborhood search with learning automata for the unrelated parallel machine scheduling problem. *2017 ieee congress on evolutionary computation (cec)* (p. 185-192). 10.1109/CEC.2017.7969312

Cota, L.P., Haddad, M.N., Freitas Souza, M.J., Coelho, V.N. (2014). Airp: A heuristic algorithm for solving the unrelated parallel machine scheduling problem. *2014 ieee congress on evolutionary computation (cec)* (p. 1855-1862). 10.1109/CEC.2014.6900245

Cruz-Chavez, M.A., Juarez-Perez, F., Avila-Melgar, E.Y., Martinez-Oropeza, A. (2009). Simulated annealing algorithm for the weighted unrelated parallel machines problem. *2009 electronics, robotics and automotive mechanics conference (cerma)* (p. 94-99). 10.1109/CERMA.2009.46

De, P., & Morton, T.E. (1980). Scheduling to minimize makespan on unequal parallel processors. *Decision Sciences*, *11*(4), 586-602. Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-5915.1980.tb01163.x https://arxiv.org/abs/https://onlinelibrary.wiley

.com/doi/pdf/10.1111/j.1540-5915.1980.tb01163.x
https://doi.org/10.1111/j.1540-5915.1980.tb01163.x

de C. M. Nogueira, J.P., Arroyo, J.E.C., Villadiego, H.M.M., Gonçalves, L.B. (2014). Hybrid grasp heuristics to solve an unrelated parallel machine scheduling problem with earliness and tardiness penalties. *Electronic Notes in Theoretical Computer Science*, *302*, 53-72. Retrieved from https://www.sciencedirect.com/science/article/pii/S1571066114000218 (Proceedings of the XXXIX Latin American Computing Conference (CLEI 2013))

https://doi.org/10.1016/j.entcs.2014.01.020

de Abreu, L.R., & de Athayde Prata, B. (2020, January). A genetic algorithm with neighborhood search procedures for unrelated parallel machine scheduling problem with sequence-dependent setup times. *Journal of Modelling in Management*, *15*(3), 809–828. Retrieved from https://doi.org/10.1108/jm2-12-2018-0209

10.1108/jm2-12-2018-0209

de Paula, M.R., Ravetti, M.G., Mateus, G.R., Pardalos, P.M. (2007, April). Solving parallel machines scheduling problems with sequence-dependent setup times using variable neighbourhood search. *IMA Journal of Management Mathematics*, *18*(2), 101–115. Retrieved from https://doi.org/10.1093/imaman/dpm016

10.1093/imaman/dpm016

Dhaenens-Flipo, C. (2001). A bicriterion approach to deal with a constrained single-objective problem. *International Journal of Production Economics*, *74*(1), 93-101. Retrieved from https://www.sciencedirect.com/science/article/pii/S0925527301001104 (Poductive Systems: Strategy, Control, and Management)

https://doi.org/10.1016/S0925-5273(01)00110-4

Diana, R.O., de Souza, S.R., Filho, M.F. (2018). A variable neighborhood descent as ils local search to the minimization of the total weighted tardiness on unrelated parallel machines and sequence dependent setup times. *Electronic Notes in Discrete Mathematics*, *66*, 191-198. Retrieved from https://www.sciencedirect.com/science/article/pii/S1571065318300714 (5th International Conference on Variable Neighborhood Search)

https://doi.org/10.1016/j.endm.2018.03.025

Diana, R.O., Filho, M.F.d.F., Souza, S.R.d., Silva, M.A.L. (2013). A clonal selection algorithm for makespan minimization on unrelated parallel machines with sequence dependent setup times. *2013 brazilian conference on intelligent systems* (p. 57-63). 10.1109/BRACIS.2013.18

Diana, R.O.M., de França Filho, M.F., de Souza, S.R., de Almeida Vitor, J.F. (2015). An immune-inspired algorithm for an unrelated parallel machines' scheduling problem with sequence and machine dependent setup-times for makespan minimisation. *Neurocomputing*, *163*, 94-105. Retrieved from https://www.sciencedirect.com/science/article/pii/S0925231215003732 (Recent Advancements in Hybrid Artificial Intelligence Systems and its Application to Real-World Problems Progress in Intelligent Systems Mining Humanistic Data)

https://doi.org/10.1016/j.neucom.2014.06.091

Diana, R.O.M., de Souza, S.R., Wanner, E.F., Filho, M.F.F. (2017). Hybrid metaheuristic for combinatorial optimization based on immune network for optimization and vns. *Proceedings of the genetic and evolutionary computation conference* (p. 251–258). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3071178.3071269 10.1145/3071178.3071269

Dolgui, A., Eremeev, A.V., Kovalyov, M.Y., Kuznetsov, P.M. (2009). Multi-product lot-sizing and scheduling on unrelated parallel machines to minimize makespan. *IFAC Proceedings Volumes*, *42*(4), 828-833. Retrieved from https://www.sciencedirect.com/science/article/pii/S1474667016338964 (13th IFAC Symposium on Information Control Problems in Manufacturing)

https://doi.org/10.3182/20090603-3-RU-2001.0553

Du Kim, H., & Kim, J.S. (2004). An online scheduling algorithm for grid computing systems. M. Li, X.-H. Sun, Q. Deng, & J. Ni (Eds.), *Grid and cooperative computing* (pp. 34–39). Berlin, Heidelberg: Springer Berlin Heidelberg.

Ebrahimi, E., & Rezaeian, J. (2015, May). Unrelated parallel machines scheduling with the effect of aging and learning under multi maintenance activities. *Manufacturing Science and Technology*, *3*(2), 25–31. Retrieved from https://doi.org/10.13189/mst.2015.030201

10.13189/mst.2015.030201

Erdelić, T., & Carić, T. (2019, May). A survey on the electric vehicle routing problem: Variants and solution approaches. *Journal of Advanced Transportation*, *2019*, 1–48. Retrieved from 10.1155/2019/5075671

10.1155/2019/5075671

Ernst, A., Jiang, H., Krishnamoorthy, M., Sier, D. (2004). Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, *153*(1), 3-27. Retrieved from https://www.sciencedirect.com/science/article/pii/S037722170300095X (Timetabling and Rostering)

https://doi.org/10.1016/S0377-2217(03)00095-X

Eroglu, D.Y., Ozmutlu, H.C., Ozmutlu, S. (2014). Genetic algorithm with local search for the unrelated parallel machine scheduling problem with sequence-dependent set-up times. *International Journal of Production Research*, *52*(19), 5841-5856. Retrieved from https://doi.org/10.1080/00207543.2014.920966 https://arxiv.org/abs/ https://doi.org/10.1080/00207543.2014.920966
10.1080/00207543.2014.920966

e Santos, A.S., & Madureira, A.M. (2014). Ordered minimum completion time heuristic for unrelated parallel-machines problems. *2014 9th iberian conference on information systems and technologies (cisti)* (p. 1-6). 10 .1109/CISTI.2014.6876939

Ewees, A.A., Al-qaness, M.A., Abd Elaziz, M. (2021). Enhanced salp swarm algorithm based on firefly algorithm for unrelated parallel machine scheduling with setup times. *Applied Mathematical Modelling*, *94*, 285-305. Retrieved from https://www.sciencedirect.com/science/article/pii/S0307904X21000263

https://doi.org/10.1016/j.apm.2021.01.017

Ezugwu, A.E. (2019). Enhanced symbiotic organisms search algorithm for unrelated parallel machines manufacturing scheduling with setup times. *Knowledge-Based Systems*, *172*, 15-32. Retrieved from https://www.sciencedirect.com/science/article/pii/S0950705119300504

https://doi.org/10.1016/j.knosys.2019.02.005

Ezugwu, A.E., Adeleke, O.J., Viriri, S. (2018, July). Symbiotic organisms search algorithm for the unrelated parallel machines scheduling with

sequence-dependent setup times. *PLOS ONE*, *13*(7), e0200030. Retrieved from https://doi.org/10.1371/journal.pone.0200030

10.1371/journal.pone.0200030

Ezugwu, A.E., & Akutsah, F. (2018). An improved firefly algorithm for the unrelated parallel machines scheduling problem with sequence-dependent setup times. *IEEE Access*, *6*, 54459-54478.

10.1109/ACCESS.2018.2872110

Fanjul-Peyro, L. (2020). Models and an exact method for the unrelated parallel machine scheduling problem with setups and resources. *Expert Systems with Applications: X*, *5*, 100022. Retrieved from https://www.sciencedirect.com/science/article/pii/S2590188520300019

https://doi.org/10.1016/j.eswax.2020.100022

Fanjul-Peyro, L., Perea, F., Ruiz, R. (2017, July). Models and matheuristics for the unrelated parallel machine scheduling problem with additional resources. *European Journal of Operational Research*, *260*(2), 482–493. Retrieved from https://doi.org/10.1016/j.ejor.2017.01.002

10.1016/j.ejor.2017.01.002

Fanjul-Peyro, L., & Ruiz, R. (2010). Iterated greedy local search methods for unrelated parallel machine scheduling. *European Journal of Operational Research*, *207*(1), 55-69. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221710002572

https://doi.org/10.1016/j.ejor.2010.03.030

Fanjul-Peyro, L., & Ruiz, R. (2011). Size-reduction heuristics for the unrelated parallel machines scheduling problem. *Computers & Operations Research*, *38*(1), 301-309. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054810001188 (Project Management and Scheduling)

https://doi.org/10.1016/j.cor.2010.05.005

Fanjul-Peyro, L., & Ruiz, R. (2012). Scheduling unrelated parallel machines with optional machines and jobs selection. *Computers & Operations Research*, *39*(7), 1745-1753. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054811003005

https://doi.org/10.1016/j.cor.2011.10.012

Fleszar, K., Charalambous, C., Hindi, K.S. (2011, March). A variable neighborhood descent heuristic for the problem of makespan minimisation on unrelated parallel machines with setup times. *Journal of Intelligent Manufacturing*, *23*(5), 1949–1958. Retrieved from https://doi.org/10.1007/s10845-011-0522-8

10.1007/s10845-011-0522-8

Gao, J. (2005). A parallel hybrid genetic algorithm for solving a kind of non-identical parallel machine scheduling problems. *Eighth international conference on high-performance computing in asia-pacific region (hpcasia'05)* (p. 4 pp.-472). 10.1109/HPCASIA.2005.8

Gao, J. (2010). A novel artificial immune system for solving multiobjective scheduling problems subject to special process constraint. *Computers & Industrial Engineering*, *58*(4), 602-609. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835210000033

https://doi.org/10.1016/j.cie.2009.12.009

Gao, J., He, G., Wang, Y. (2008, August). A new parallel genetic algorithm for solving multiobjective scheduling problems subjected to special process constraint. *The International Journal of Advanced Manufacturing Technology*, *43*(1-2), 151–160. Retrieved from https://doi.org/10.1007/s00170-008-1683-2

10.1007/s00170-008-1683-2

Gedik, R., Kalathia, D., Egilmez, G., Kirac, E. (2018). A constraint programming approach for solving unrelated parallel machine scheduling problem. *Computers & Industrial Engineering*, *121*, 139-149. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835218302158

https://doi.org/10.1016/j.cie.2018.05.014

Ghaleb, M., Taghipour, S., Zolfagharinia, H. (2020, August). Joint optimization of maintenance and production scheduling for unrelated parallel-machine system. *2020 asia-pacific international symposium on advanced reliability and maintenance modeling (APARM)*. IEEE. Retrieved from https://doi.org/10.1109/aparm49247.2020.9209399 10

.1109/aparm49247.2020.9209399

Glass, C., Potts, C., Shade, P. (1994). Unrelated parallel machine scheduling using local search. *Mathematical and Computer Modelling*, *20*(2), 41-52. Retrieved from https://www.sciencedirect.com/science/article/pii/0895717794902054

https://doi.org/10.1016/0895-7177(94)90205-4

Golconda, K.S., Dogan, A., Özgüner, F. (2004). Static mapping heuristics for tasks with hard deadlines in real-time heterogeneous systems. C. Aykanat, T. Dayar, & I. Korpeoglu (Eds.), *Computer and information sciences - ISCIS 2004, 19th international symposium, kemer-antalya, turkey, october 27-29, 2004. proceedings* (Vol. 3280, pp. 827–836). Springer. Retrieved from https://doi.org/10.1007/978-3-540-30182-0_83 10.1007/978-3-540-30182-0_83

Graham, R., Lawler, E., Lenstra, J., Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. P. Hammer, E. Johnson, & B. Korte (Eds.), *Discrete optimization ii* (Vol. 5, p. 287-326). Elsevier. Retrieved from https://www.sciencedirect.com/science/article/pii/S016750600870356X https://doi.org/10.1016/S0167-5060(08)70356-X

GUO, Y., LIM, A., RODRIGUES, B., YANG, L. (2007, June). MINIMIZING THE MAKESPAN FOR UNRELATED PARALLEL MACHINES. *International Journal on Artificial Intelligence Tools*, *16*(03), 399–415. Retrieved from https://doi.org/10.1142/s0218213007003175

10.1142/s0218213007003175

Haddad, M.N., Coelho, I.M., Souza, M.J.F., Ochi, L.S., Santos, H.G., Martins, A.X. (2012). Garp: A new genetic algorithm for the unrelated parallel machine scheduling problem with setup times. *2012 31st international conference of the chilean computer science society* (p. 152-160). 10.1109/SCCC.2012.25

Hariri, A., & Potts, C. (1991). Heuristics for scheduling unrelated parallel machines. *Computers & Operations Research*, *18*(3), 323-331. Retrieved from https://www.sciencedirect.com/science/article/pii/030505489190034O

https://doi.org/10.1016/0305-0548(91)90034-O

Hart, E., Ross, P., Corne, D. (2005, June). Evolutionary scheduling: A review. *Genetic Programming and Evolvable Machines*, *6*(2), 191–220.

10.1007/s10710-005-7580-7

Hassan Abdel-Jabbar, M.-A., Kacem, I., Martin, S. (2014). Unrelated parallel machines with precedence constraints: application to cloud computing. *2014 ieee 3rd international conference on cloud networking (cloudnet)* (p. 438-442). 10.1109/CloudNet.2014.6969034

Helal, M., Rabadi, G., Al-Salem, A. (2006, 01). A tabu search algorithm to minimize the makespan for the unrelated parallel machines scheduling problem with setup times. *International Journal of Operations Research*, *3*, 182-192.

Herrmann, J., Proth, J.-M., Sauer, N. (1997). Heuristics for unrelated machine scheduling with precedence constraints. *European Journal of Operational Research*, *102*(3), 528-537. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221796002470

https://doi.org/10.1016/S0377-2217(96)00247-0

Ibarra, O.H., & Kim, C.E. (1977, April). Heuristic algorithms for scheduling independent tasks on nonidentical processors. *J. ACM*, *24*(2), 280–289. Retrieved from https://doi.org/10.1145/322003.322011

10.1145/322003.322011

Izakian, H., Abraham, A., Snasel, V. (2009). Comparison of heuristics for scheduling independent tasks on heterogeneous distributed environments. *2009 international joint conference on computational sciences and optimization* (Vol. 1, p. 8-12). 10.1109/CSO.2009.487

Jaklinovic, K., Đurasevic, M., Jakobovic, D. (2021). Designing dispatching rules with genetic programming for the unrelated machines environment with constraints. *Expert Systems with Applications*, *172*, 114548. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417420311921

https://doi.org/10.1016/j.eswa.2020.114548

Jolai, F., Amalnick, M.S., Alinaghian, M., Shakhsi-Niaei, M., Omrani, H. (2009, July). A hybrid memetic algorithm for maximizing the weighted number of just-in-time jobs on unrelated parallel machines. *Journal of Intelligent Manufacturing*, *22*(2), 247–261. Retrieved from https://doi.org/10.1007/s10845-009-0285-7

10.1007/s10845-009-0285-7

Joo, C.M., & Kim, B.S. (2015). Hybrid genetic algorithms with dispatching rules for unrelated parallel machine scheduling with setup time and production availability. *Computers & Industrial Engineering*, *85*, 102-109. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835215001084

https://doi.org/10.1016/j.cie.2015.02.029

Joo, C.M., & Kim, B.S. (2017). Rule-based meta-heuristics for integrated scheduling of unrelated parallel machines, batches, and heterogeneous delivery trucks. *Applied Soft Computing*, *53*, 457-476. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494616306639

https://doi.org/10.1016/j.asoc.2016.12.038

Jou, C. (2005). A genetic algorithm with sub-indexed partitioning genes and its application to production scheduling of parallel machines. *Computers & Industrial Engineering*, *48*(1), 39-54. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835204001354 (Selected Papers from the 31st. International Conference on Computers and Industrial Engineering)

https://doi.org/10.1016/j.cie.2004.07.007

Jouhari, H., Lei, D., A. A. Al-qaness, M., Abd Elaziz, M., Ewees, A.A., Farouk, O. (2019). Sine-cosine algorithm to enhance simulated annealing for unrelated parallel machine scheduling with setup times. *Mathematics*, *7*(11). Retrieved from https://www.mdpi.com/2227-7390/7/11/1120

10.3390/math7111120

Jouhari, H., Lei, D., Al-qaness, M.A.A., Elaziz, M.A., Damaševičius, R., Korytkowski, M., Ewees, A.A. (2020). Modified harris hawks optimizer for solving machine scheduling problems. *Symmetry*, *12*(9). Retrieved from https://www.mdpi.com/2073-8994/12/9/1460

10.3390/sym12091460

Jovanovic, R., & Voß, S. (2021). Fixed set search application for minimizing the makespan on unrelated parallel machines with sequence-dependent setup times. *Applied Soft Computing*, *110*, 107521. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494621004440

https://doi.org/10.1016/j.asoc.2021.107521

Kayvanfar, V., & Teymourian, E. (2014). Hybrid intelligent water drops algorithm to unrelated parallel machines scheduling problem: a just-in-time approach. *International Journal of Production Research*, *52*(19), 5857-5879. Retrieved from https://doi.org/10.1080/00207543.2014.923124 https://arxiv.org/abs/ https://doi.org/10.1080/00207543.2014.923124 10.1080/00207543.2014.923124

Keskinturk, T., Yildirim, M.B., Barut, M. (2012). An ant colony optimization algorithm for load balancing in parallel machines with sequence-dependent setup times. *Computers & Operations Research*, *39*(6), 1225-1235. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054810002935 (Special Issue on Scheduling in Manufacturing Systems)

https://doi.org/10.1016/j.cor.2010.12.003

Kim, D.-W., Kim, K.-H., Jang, W., Frank Chen, F. (2002). Unrelated parallel machine scheduling with setup times using simulated annealing. *Robotics and Computer-Integrated Manufacturing*, *18*(3), 223-231. Retrieved from https://www.sciencedirect.com/science/article/pii/S0736584502000133 (11th International Conference on Flexible Automation and Intelligent Manufacturing)

https://doi.org/10.1016/S0736-5845(02)00013-3

Kim, D.-W., Na, D.-G., Frank Chen, F. (2003). Unrelated parallel machine scheduling with setup times and a total weighted tardiness objective. *Robotics and Computer-Integrated Manufacturing*, *19*(1), 173-181. Retrieved from https://www.sciencedirect.com/science/article/pii/S0736584502000777 (12th International Conference on Flexible Automation and Intellig ent Manufacturing)

https://doi.org/10.1016/S0736-5845(02)00077-7

Kim, S.-I., Choi, H.-S., Lee, D.-H. (2006). Tabu search heuristics for parallel machine scheduling with sequence-dependent setup and ready times. *Computational science and its applications - ICCSA 2006* (pp. 728–737). Springer Berlin Heidelberg. Retrieved from https://doi.org/10.1007/11751595_77 10.1007/11751595_77

Kim, S.-I., Choi, H.-S., Lee, D.-H. (2007). Scheduling algorithms for parallel machines with sequence-dependent set-up and distinct ready times: Minimizing total tardiness. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *221*(6), 1087-1096. Retrieved from https://doi.org/10.1243/09544054JEM779 https://arxiv.org/abs/https://doi.org/10.1243/09544054JEM779 10.1243/09544054JEM779

Klemmt, A., Weigert, G., Almeder, C., Mönch, L. (2009). A comparison of mip-based decomposition techniques and vns approaches for batch scheduling problems. *Proceedings of the 2009 winter simulation conference (wsc)* (p. 1686-1694). 10.1109/WSC.2009.5429173

Koza, J.R. (2010, May). Human-competitive results produced by genetic programming. *Genetic Programming and Evolvable Machines*, *11*(3-4), 251–284. Retrieved from https://doi.org/10.1007/s10710-010-9112-3

10.1007/s10710-010-9112-3

Kramer, A., & Subramanian, A. (2017, December). A unified heuristic and an annotated bibliography for a large class of earliness–tardiness scheduling problems. *Journal of Scheduling*, *22*(1), 21–57. Retrieved from https://doi.org/10.1007/s10951-017-0549-6

10.1007/s10951-017-0549-6

Kucukoglu, I., Dewil, R., Cattrysse, D. (2021). The electric vehicle routing problem and its variations: A literature review. *Computers & Industrial Engineering*, *161*, 107650.

10.1016/j.cie.2021.107650

Lee, C.-H., Liao, C.-J., Chao, C.-W. (2014). Unrelated parallel machine scheduling with dedicated machines and common deadline. *Computers & Industrial Engineering*, *74*, 161-168. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835214001570

https://doi.org/10.1016/j.cie.2014.05.012

Lee, J.-H., Yu, J.-M., Lee, D.-H. (2013, July). A tabu search algorithm for unrelated parallel machine scheduling with sequence- and machine-dependent setups: minimizing total tardiness. *The International Journal of Advanced Manufacturing Technology*, *69*(9-12), 2081–2089. Retrieved from https://doi.org/10.1007/s00170-013-5192-6

10.1007/s00170-013-5192-6

Lei, D., & Cai, J. (2020). Multi-population meta-heuristics for production scheduling: A survey. *Swarm and Evolutionary Computation*, *58*, 100739. Retrieved from https://www.sciencedirect.com/science/article/pii/S2210650220303928

https://doi.org/10.1016/j.swevo.2020.100739

Lei, D., & Liu, M. (2020). An artificial bee colony with division for distributed unrelated parallel machine scheduling with preventive maintenance. *Computers & Industrial Engineering*, *141*, 106320. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835220300541

https://doi.org/10.1016/j.cie.2020.106320

Lei, D., Yuan, Y., Cai, J. (2020). An improved artificial bee colony for multi-objective distributed unrelated parallel machine scheduling. *International Journal of Production Research*, *0*(0), 1-13. Retrieved from https://doi.org/10.1080/00207543.2020.1775911 https://arxiv.org/abs/https://doi.org/10.1080/00207543.2020.1775911
10.1080/00207543.2020.1775911

Lei, D., Yuan, Y., Cai, J., Bai, D. (2020). An imperialist competitive algorithm with memory for distributed unrelated parallel machines scheduling. *International Journal of Production Research*, *58*(2), 597-614. Retrieved from https://doi.org/10.1080/00207543.2019.1598596 https://arxiv.org/abs/https://doi.org/10.1080/00207543.2019.1598596
10.1080/00207543.2019.1598596

Lensen, A., Xue, B., Zhang, M. (2021, 8). Genetic Programming for Manifold Learning: Preserving Local Topology.

10.25455/wgtn.16416828.v1

Lenstra, J.K., Shmoys, D.B., Tardos, É. (1990, January). Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, *46*(1-3), 259–271. Retrieved from https://doi.org/10.1007/bf01585745

10.1007/bf01585745

Leung, J.Y.-T. (2004). *Handbook of scheduling : algorithms, models, and performance analysis*. Boca Raton, Fla.: Chapman & Hall/CRC.

Li, X., Huang, Y., Tan, Q., Chen, H. (2013). Scheduling unrelated parallel batch processing machines with non-identical job sizes. *Computers & Operations Research*, *40*(12), 2983-2990. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054813001731

https://doi.org/10.1016/j.cor.2013.06.016

Li, Z., Yang, H., Zhang, S., Liu, G. (2015, August). Unrelated parallel machine scheduling problem with energy and tardiness cost. *The International Journal of Advanced Manufacturing Technology*, *84*(1-4), 213–226. Retrieved from https://doi.org/10.1007/s00170-015-7657-2

10.1007/s00170-015-7657-2

Liang, P., dong Yang, H., sheng Liu, G., hua Guo, J. (2015). An ant optimization model for unrelated parallel machine scheduling with energy consumption and total tardiness. *Mathematical Problems in Engineering*, *2015*, 1–8. Retrieved from https://doi.org/10.1155/2015/907034

10.1155/2015/907034

Liao, C.-J., Lee, C.-H., Tsai, H.-T. (2016). Scheduling with multi-attribute set-up times on unrelated parallel machines. *International Journal of Production Research*, *54*(16), 4839-4853. Retrieved from https://doi.org/10.1080/00207543.2015.1118574 https://arxiv.org/abs/ https://doi.org/10.1080/00207543.2015.1118574

10.1080/00207543.2015.1118574

Liao, T., Chang, P., Kuo, R., Liao, C.-J. (2014). A comparison of five hybrid metaheuristic algorithms for unrelated parallel-machine scheduling and inbound trucks sequencing in multi-door cross docking systems. *Applied Soft Computing*, *21*, 180-193. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494614001070

https://doi.org/10.1016/j.asoc.2014.02.026

Lin, C.-W., Lin, Y.-K., Hsieh, H.-T. (2013, February). Ant colony optimization for unrelated parallel machine scheduling. *The International Journal of Advanced Manufacturing Technology*, *67*(1-4), 35–45. Retrieved from https://doi.org/10.1007/s00170-013-4766-7

10.1007/s00170-013-4766-7

Lin, D.-Y., & Huang, T.-Y. (2021). A hybrid metaheuristic for the unrelated parallel machine scheduling problem. *Mathematics*, *9*(7). Retrieved from https://www.mdpi.com/2227-7390/9/7/768

10.3390/math9070768

Lin, S.-W., Lu, C.-C., Ying, K.-C. (2010, July). Minimization of total tardiness on unrelated parallel machines with sequence- and machine-dependent setup times under due date constraints. *The International Journal of Advanced Manufacturing Technology*, *53*(1-4), 353–361. Retrieved from https://doi.org/10.1007/s00170-010-2824-y

10.1007/s00170-010-2824-y

Lin, S.-W., & Ying, K.-C. (2014). Abc-based manufacturing scheduling for unrelated parallel machines with machine-dependent and job sequence-dependent setup times. *Computers & Operations Research*, *51*, 172-181. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054814001427

https://doi.org/10.1016/j.cor.2014.05.013

Lin, S.-W., & Ying, K.-C. (2015). A multi-point simulated annealing heuristic for solving multiple objective unrelated parallel machine scheduling problems. *International Journal of Production Research*, *53*(4), 1065-1076. Retrieved from https://doi.org/10.1080/00207543.2014.942011 https://arxiv.org/abs/https://doi.org/10.1080/00207543.2014.942011

10.1080/00207543.2014.942011

Lin, S.-W., Ying, K.-C., Wu, W.-J., Chiang, Y.-I. (2016). Multi-objective unrelated parallel machine scheduling: a tabu-enhanced iterated pareto greedy algorithm. *International Journal of Production Research*, *54*(4), 1110-1121. Retrieved from https://doi.org/10.1080/00207543.2015.1047981 https://arxiv.org/abs/https://doi.org/10.1080/00207543.2015.1047981

10.1080/00207543.2015.1047981

Lin, Y., Pfund, M., Fowler, J. (2011). Heuristics for minimizing regular performance measures in unrelated parallel machine scheduling problems. *Computers & Operations Research*, *38*(6), 901-916. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054810001887

https://doi.org/10.1016/j.cor.2010.08.018

Lin, Y.-K. (2013). Particle swarm optimization algorithm for unrelated parallel machine scheduling with release dates. *Mathematical Problems in Engineering*, *2013*, 1–9. Retrieved from https://doi.org/10.1155/2013/409486

10.1155/2013/409486

Lin, Y.-K., Fowler, J.W., Pfund, M.E. (2013). Multiple-objective heuristics for scheduling unrelated parallel machines. *European Journal of Operational Research*, *227*(2), 239-253. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221712007357

https://doi.org/10.1016/j.ejor.2012.10.008

Lin, Y.-K., & Hsieh, F.-Y. (2014). Unrelated parallel machine scheduling with setup times and ready times. *International Journal of Production Research*, *52*(4), 1200-1214. Retrieved from https://doi.org/10.1080/00207543.2013.848305 https://arxiv.org/abs/ https://doi.org/10.1080/00207543.2013.848305

10.1080/00207543.2013.848305

Lin, Y.-K., & Lin, H.-C. (2015). Bicriteria scheduling problem for unrelated parallel machines with release dates. *Computers & Operations Research*, *64*, 28-39. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054815001203

https://doi.org/10.1016/j.cor.2015.04.025

Liu, C. (2013). A hybrid genetic algorithm to minimize total tardiness for unrelated parallel machine scheduling with precedence constraints. *Mathematical Problems in Engineering*, *2013*, 1–11. Retrieved from https://doi.org/10.1155/2013/537127

10.1155/2013/537127

Liu, C., & Yang, S. (2011, June). A heuristic serial schedule algorithm for unrelated parallel machine scheduling with precedence constraints. *Journal of Software*, *6*(6). Retrieved from https://doi.org/10.4304/jsw.6.6.1146-1153

10.4304/jsw.6.6.1146-1153

Logendran, R., McDonell, B., Smucker, B. (2007, 11). Scheduling unrelated parallel machines with sequence-dependent setups. *Computers &*

Operations Research, *34*, 3420-3438.

10.1016/j.cor.2006.02.006

LOGENDRAN, R., & SUBUR, F. (2004). Unrelated parallel machine scheduling with job splitting. *IIE Transactions*, *36*(4), 359-372. Retrieved from https://doi.org/10.1080/07408170490279598 https://arxiv.org/abs/https://doi.org/10.1080/07408170490279598
10.1080/07408170490279598

long Zheng, X., & Wang, L. (2016). A two-stage adaptive fruit fly optimization algorithm for unrelated parallel machine scheduling problem with additional resource constraints. *Expert Systems with Applications*, *65*, 28-39. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417416304389

https://doi.org/10.1016/j.eswa.2016.08.039

Low, C., Li, R.-K., Wu, G.-H. (2013). Ant colony optimization algorithms for unrelated parallel machine scheduling with controllable processing times and eligibility constraints. *Proceedings of the institute of industrial engineers asian conference 2013* (pp. 79–87). Springer Singapore. Retrieved from https://doi.org/10.1007/978-981-4451-98-7_10 10 .1007/978-981-4451-98-7_10

Low, C., & Wu, G.-H. (2016, February). Unrelated parallel-machine scheduling with controllable processing times and eligibility constraints to minimize the makespan. *Journal of Industrial and Production Engineering*, *33*(4), 286–293. Retrieved from https://doi.org/10.1080/21681015.2016.1139005

10.1080/21681015.2016.1139005

Lu, H., & Qiao, F. (2017). An improved genetic algorithm for a parallel machine scheduling problem with energy consideration. *2017 13th ieee conference on automation science and engineering (case)* (p. 1487-1492). 10.1109/COASE.2017.8256314

Lu, S., Liu, X., Pei, J., T. Thai, M., M. Pardalos, P. (2018). A hybrid abc-ts algorithm for the unrelated parallel-batching machines scheduling problem with deteriorating jobs and maintenance activity. *Applied Soft Computing*, *66*, 168-182. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494618300772

https://doi.org/10.1016/j.asoc.2018.02.018

Luo, P., Lü, K., Shi, Z. (2007). A revisit of fast greedy heuristics for mapping a class of independent tasks onto heterogeneous computing systems. *Journal of Parallel and Distributed Computing*, *67*(6), 695-714. Retrieved from https://www.sciencedirect.com/science/article/pii/S0743731507000433

https://doi.org/10.1016/j.jpdc.2007.03.003

Maheswaran, M., Ali, S., Siegel, H.J., Hensgen, D., Freund, R.F. (1999). Dynamic mapping of a class of independent tasks onto heterogeneous computing systems. *Journal of Parallel and Distributed Computing*, *59*(2), 107-131. Retrieved from https://www.sciencedirect.com/science/article/pii/S0743731599915812

https://doi.org/10.1006/jpdc.1999.1581

MANUPATI, V.K., RAJYALAKSHMI, G., CHAN, F.T.S., THAKKAR, J.J. (2017, February). A hybrid multi-objective evolutionary algorithm approach for handling sequence- and machine-dependent set-up times in unrelated parallel machine scheduling problem. *Sādhanā*, *42*(3), 391–403. Retrieved from https://doi.org/10.1007/s12046-017-0611-2

10.1007/s12046-017-0611-2

Marinho Diana, R.O., & de Souza, S.R. (2020). Analysis of variable neighborhood descent as a local search operator for total weighted tardiness problem on unrelated parallel machines. *Computers & Operations Research*, *117*, 104886. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054820300034

https://doi.org/10.1016/j.cor.2020.104886

Mehravaran, Y., & Logendran, R. (2011). Bicriteria supply chain scheduling on unrelated-parallel machines. *Journal of the Chinese Institute of Industrial Engineers*, *28*(2), 91-101. Retrieved from https://doi.org/10.1080/10170669.2010.546165 https://arxiv.org/abs/ https://doi.org/10.1080/10170669.2010.546165
10.1080/10170669.2010.546165

Min, L., & Cheng, W. (2006). Genetic algorithms for the optimal common due date assignment and the optimal scheduling policy in

parallel machine earliness/tardiness scheduling problems. *Robotics and Computer-Integrated Manufacturing*, *22*(4), 279-287. Retrieved from https://www.sciencedirect.com/science/article/pii/S073658450500044X

https://doi.org/10.1016/j.rcim.2004.12.005

Moghdani, R., Salimifard, K., Demir, E., Benyettou, A. (2021). The green vehicle routing problem: A systematic literature review. *Journal of Cleaner Production*, *279*, 123691.

10.1016/j.jclepro.2020.123691

Mokotoff, E. (2001, 11). Parallel machine scheduling problems: A survey. *Asia-Pacific Journal of Operational Research*, *18*.

Morton, T.E., & Pentico, D.W. (1993). *Heuristic scheduling systems: With applications to production systems and project management*. New York: Wiley.

Munir, E.U., Li, J., Shi, S., Zou, Z., Yang, D. (2008, May). MaxStd: A task scheduling heuristic for heterogeneous computing environment. *Information Technology Journal*, *7*(4), 679–683. Retrieved from https://doi.org/10.3923/itj.2008.679.683

10.3923/itj.2008.679.683

Na, D.-G., Kim, D.-W., Jang, W., Chen, F.F. (2006). Scheduling unrelated parallel machines to minimize total weighted tardiness. *2006 ieee international conference on service operations and logistics, and informatics* (p. 758-763). 10.1109/SOLI.2006.329085

Nanthapodej, R., Liu, C.-H., Nitisiri, K., Pattanapairoj, S. (2021a). Hybrid differential evolution algorithm and adaptive large neighborhood search to solve parallel machine scheduling to minimize energy consumption in consideration of machine-load balance problems. *Sustainability*, *13*(10). Retrieved from https://www.mdpi.com/2071-1050/13/10/5470

10.3390/su13105470

Nanthapodej, R., Liu, C.-H., Nitisiri, K., Pattanapairoj, S. (2021b). Variable neighborhood strategy adaptive search to solve parallel-machine scheduling to minimize energy consumption while considering job priority and control makespan. *Applied Sciences*, *11*(11). Retrieved from https://www.mdpi.com/2076-3417/11/11/5311

10.3390/app11115311

Nawaz, M., Enscore, E.E., Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, *11*(1), 91-95. Retrieved from https://www.sciencedirect.com/science/article/pii/0305048383900889

https://doi.org/10.1016/0305-0483(83)90088-9

Nguyen, S., Mei, Y., Zhang, M. (2017, February). Genetic programming for production scheduling: a survey with a unified framework. *Complex & Intelligent Systems*, *3*(1), 41–66. Retrieved from https://doi.org/10.1007/s40747-017-0036-x

10.1007/s40747-017-0036-x

Niu, Q., Zhou, F., Zhou, T.J. (2011, February). An adaptive clonal selection algorithm with stage mutation operation for unrelated parallel machine scheduling problem with sequence-dependent setup times. *Key Engineering Materials*, *467-469*, 1967–1972. Retrieved from https://doi.org/10.4028/www.scientific.net/kem.467-469.1967

10.4028/www.scientific.net/kem.467-469.1967

Nohra Haddad, M., Perdigão Cota, L., Jamilson Freitas Souza, M., Maculan, N. (2014). Aiv: A heuristic algorithm based on iterated local search and variable neighborhood descent for solving the unrelated parallel machine scheduling problem with setup times. *Proceedings of the 16th international conference on enterprise information systems - volume 1* (p. 376–383). Setubal, PRT: SCITEPRESS - Science and Technology Publications, Lda. Retrieved from https://doi.org/10.5220/0004884603760383 10.5220/0004884603760383

Orts, F., Ortega, G., Puertas, A.M., García, I., Garzón, E.M. (2020, January). On solving the unrelated parallel machine scheduling problem: active microrheology as a case study. *The Journal of Supercomputing*, *76*(11), 8494–8509. Retrieved from https://doi.org/10.1007/s11227-019-03121-z

10.1007/s11227-019-03121-z

Ouelhadj, D., & Petrovic, S. (2008, October). A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling*, *12*(4), 417–431. Retrieved from https://doi.org/10.1007/s10951-008-0090-8

10.1007/s10951-008-0090-8

Pan, Z., Lei, D., Wang, L. (2020). A knowledge-based two-population optimization algorithm for distributed energy-efficient parallel machines scheduling. *IEEE Transactions on Cybernetics*, 1-13.

10.1109/TCYB.2020.3026571

Pan, Z., Lei, D., Zhang, Q. (2018). A new imperialist competitive algorithm for multiobjective low carbon parallel machines scheduling. *Mathematical Problems in Engineering*, *2018*, 1–13. Retrieved from https://doi.org/10.1155/2018/5914360

10.1155/2018/5914360

Panwalkar, S.S., & Iskander, W. (1977, 2021/07/04/). A survey of scheduling rules. *Operations Research*, *25*(1), 45-61. Retrieved from http://www.jstor.org/stable/169546 (Full publication date: Jan. - Feb., 1977)

Peng, J., & Liu, B. (2004). Parallel machine scheduling models with fuzzy processing times. *Information Sciences*, *166*(1), 49-66. Retrieved from https://www.sciencedirect.com/science/article/pii/S0020025503004067

https://doi.org/10.1016/j.ins.2003.05.012

Perez-Gonzalez, P., Fernandez-Viagas, V., Zamora García, M., Framinan, J.M. (2019). Constructive heuristics for the unrelated parallel machines scheduling problem with machine eligibility and setup times. *Computers & Industrial Engineering*, *131*, 131-145. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835219301706

https://doi.org/10.1016/j.cie.2019.03.034

Piersma, N., & van Dijk, W. (1996). A local search heuristic for unrelated parallel machine scheduling with efficient neighborhood search. *Mathematical and Computer Modelling*, *24*(9), 11-19. Retrieved from https://www.sciencedirect.com/science/article/pii/0895717796001501

https://doi.org/10.1016/0895-7177(96)00150-1

Pinedo, M.L. (2008). *Scheduling: Theory, algorithms, and systems* (3rd ed.). Springer Publishing Company, Incorporated.

Pinheiro, J.C.S.N., Arroyo, J.E.C., Fialho, L.B. (2020). Scheduling unrelated parallel machines with family setups and resource constraints

to minimize total tardiness. *Proceedings of the 2020 genetic and evolutionary computation conference companion* (p. 1409–1417). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/3377929.3398150 10.1145/3377929.3398150

Planinić, L., Backović, H., Đurasević, M., Jakobović, D. (2022). A comparative study of dispatching rule representations in evolutionary algorithms for the dynamic unrelated machines environment. *IEEE Access*, *10*, 22886-22901.

10.1109/ACCESS.2022.3151346

Planinić, L., Đurasević, M., Jakobović, D. (2021). Towards interpretable dispatching rules: Application of expression simplification methods. *2021 ieee symposium series on computational intelligence (ssci)* (p. 01-08). 10.1109/SSCI50451.2021.9659842

Planinić, L., Đurasević, M., Jakobović, D. (2021). On the application of ϵ-lexicase selection in the generation of dispatching rules. *2021 ieee congress on evolutionary computation (cec)* (p. 2125-2132). 10.1109/CEC45853.2021.9504982

Polyakovskiy, S., & M'Hallah, R. (2014). A multi-agent system for the weighted earliness tardiness parallel machine problem. *Computers & Operations Research*, *44*, 115-136. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054813003080

https://doi.org/10.1016/j.cor.2013.10.013

Priore, P., Gómez, A., Pino, R., Rosillo, R. (2014). Dynamic scheduling of manufacturing systems using machine learning: An updated review. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *28*(1), 83–97.

10.1017/S0890060413000516

Rabadi, G., Moraga, R.J., Al-Salem, A. (2006, February). Heuristics for the unrelated parallel machine scheduling problem with setup times. *Journal of Intelligent Manufacturing*, *17*(1), 85–97. Retrieved from https://doi.org/10.1007/s10845-005-5514-0

10.1007/s10845-005-5514-0

Rafsanjani, M.K., & Bardsiri, A.K. (2012). A new heuristic approach for scheduling independent tasks on heterogeneous computing systems. *International Journal of Machine Learning and Computing*, 371–376.

Retrieved from https://doi.org/10.7763/ijmlc.2012.v2.147

10.7763/ijmlc.2012.v2.147

Raja, K., Arumugam, C., Selladurai, V. (2008). Non-identical parallel-machine scheduling using genetic algorithm and fuzzy logic approach. *International Journal of Services and Operations Management*, *4*(1), 72. Retrieved from https://doi.org/10.1504/ijsom.2008.015941

10.1504/ijsom.2008.015941

Rambod, M., & Rezaeian, J. (2014). Robust meta-heuristics implementation for unrelated parallel machines scheduling problem with rework processes and machine eligibility restrictions. *Computers & Industrial Engineering*, *77*, 15-28. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835214002733

https://doi.org/10.1016/j.cie.2014.09.006

Ramezanian, R., & Saidi-Mehrabad, M. (2012). Multi-product unrelated parallel machines scheduling problem with rework processes. *Scientia Iranica*, *19*(6), 1887-1893. Retrieved from https://www.sciencedirect.com/science/article/pii/S102630981200154X

https://doi.org/10.1016/j.scient.2012.05.004

Randhawa, S.U., & Kuo, C.-H. (1997). Evaluating scheduling heuristics for non-identical parallel processors. *International Journal of Production Research*, *35*(4), 969-981. Retrieved from https://doi.org/10.1080/002075497195489 https://arxiv.org/abs/ https://doi.org/10.1080/002075497195489
10.1080/002075497195489

RANDHAWA, S.U., & SMITH, T.A. (1995). An experimental investigation of scheduling non-identical, parallel processors with sequence-dependent set-up times and due dates. *International Journal of Production Research*, *33*(1), 59-69. Retrieved from https://doi.org/10.1080/00207549508930137 https://arxiv.org/abs/ https://doi.org/10.1080/00207549508930137
10.1080/00207549508930137

Ravetti, M.G., Mateus, G.R., Rocha, P.L., Pardalos, P.M. (2007). A scheduling problem with unrelated parallel machines and sequence dependent setups. *International Journal of Operational Research*, *2*(4),

380. Retrieved from https://doi.org/10.1504/ijor.2007.014169

10.1504/ijor.2007.014169

Rezaeian Zeidi, J., Zarei, M., Shokoufi, K. (2017). Pareto-based multi-criteria evolutionary algorithm for parallel machines scheduling problem with sequence-dependent setup times. *International Journal of Engineering*, *30*(12), 1863-1869. Retrieved from http://www.ije.ir/article_73076.html https://arxiv.org/abs/http://www.ije.ir/article_73076_0206e099bd9ad1d54af1a53028ae7e18.pdf

Ritchie, G., & Levine, J. (2003). A fast, effective local search for scheduling independent jobs in heterogeneous computing environments..

Rodriguez, F.J., Blum, C., García-Martínez, C., Lozano, M. (2012, June). GRASP with path-relinking for the non-identical parallel machine scheduling problem with minimising total weighted completion times. *Annals of Operations Research*, *201*(1), 383–401. Retrieved from https://doi.org/10.1007/s10479-012-1164-8

10.1007/s10479-012-1164-8

Rodriguez, F.J., García-Martínez, C., Blum, C., Lozano, M. (2012). An artificial bee colony algorithm for the unrelated parallel machines scheduling problem. *Lecture notes in computer science* (pp. 143–152). Springer Berlin Heidelberg. Retrieved from https://doi.org/10.1007/978-3-642-32964-7_15 10.1007/978-3-642-32964-7_15

Rodriguez, F.J., Lozano, M., Blum, C., García-Martínez, C. (2013). An iterated greedy algorithm for the large-scale unrelated parallel machines scheduling problem. *Computers & Operations Research*, *40*(7), 1829-1841. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054813000269

https://doi.org/10.1016/j.cor.2013.01.018

Rojanasoonthon, S., & Bard, J. (2005, February). A GRASP for parallel machine scheduling with time windows. *INFORMS Journal on Computing*, *17*(1), 32–51. Retrieved from https://doi.org/10.1287/ijoc.1030.0048

10.1287/ijoc.1030.0048

Ruiz, R., & Andrés-Romano, C. (2011, May). Scheduling unrelated parallel machines with resource-assignable sequence-dependent setup times. *The International Journal of Advanced Manufacturing Technology*, *57*(5-8),

777–794. Retrieved from https://doi.org/10.1007/s00170-011-3318-2

10.1007/s00170-011-3318-2

Salehi Mir, M.S., & Rezaeian, J. (2016). A robust hybrid approach based on particle swarm optimization and genetic algorithm to minimize the total machine load on unrelated parallel machines. *Applied Soft Computing*, *41*, 488-504. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494615008145

https://doi.org/10.1016/j.asoc.2015.12.035

Santos, H.G., Toffolo, T.A., Silva, C.L., Berghe, G.V. (2016, June). Analysis of stochastic local search methods for the unrelated parallel machine scheduling problem. *International Transactions in Operational Research*, *26*(2), 707–724. Retrieved from https://doi.org/10.1111/itor.12316

10.1111/itor.12316

Sels, V., Coelho, J., Manuel Dias, A., Vanhoucke, M. (2015). Hybrid tabu search and a truncated branch-and-bound for the unrelated parallel machine scheduling problem. *Computers & Operations Research*, *53*, 107-117. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054814002123

https://doi.org/10.1016/j.cor.2014.08.002

Shahidi-Zadeh, B., Tavakkoli-Moghaddam, R., Taheri-Moghadam, A., Rastgar, I. (2017). Solving a bi-objective unrelated parallel batch processing machines scheduling problem: A comparison study. *Computers & Operations Research*, *88*, 71-90. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054817301570

https://doi.org/10.1016/j.cor.2017.06.019

Shahvari, O., & Logendran, R. (2015, 06). Bi-criteria batch scheduling on unrelated-parallel machines..

Shahvari, O., & Logendran, R. (2017a). A bi-objective batch processing problem with dual-resources on unrelated-parallel machines. *Applied Soft Computing*, *61*, 174-192. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494617304969

https://doi.org/10.1016/j.asoc.2017.08.014

Shahvari, O., & Logendran, R. (2017b, January). An enhanced tabu search algorithm to minimize a bi-criteria objective in batching and scheduling problems on unrelated-parallel machines with desired lower bounds on batch sizes. *Comput. Oper. Res.*, *77*(C), 154–176. Retrieved from https://doi.org/10.1016/j.cor.2016.07.021

10.1016/j.cor.2016.07.021

Siepak, M., & Józefczyk, J. (2014, February). Solution algorithms for unrelated machines minmax regret scheduling problem with interval processing times and the total flow time criterion. *Annals of Operations Research*, *222*(1), 517–533. Retrieved from https://doi.org/10.1007/s10479-014-1538-1

10.1007/s10479-014-1538-1

Silva, C., & Magalhaes, J.M. (2006). Heuristic lot size scheduling on unrelated parallel machines with applications in the textile industry. *Computers & Industrial Engineering*, *50*(1), 76-89. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835206000040

https://doi.org/10.1016/j.cie.2006.01.001

Slotnick, S.A. (2011). Order acceptance and scheduling: A taxonomy and review. *European Journal of Operational Research*, *212*(1), 1-11. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221710006405

https://doi.org/10.1016/j.ejor.2010.09.042

Soleimani, H., Ghaderi, H., Tsai, P.-W., Zarbakhshnia, N., Maleki, M. (2020). Scheduling of unrelated parallel machines considering sequence-related setup time, start time-dependent deterioration, position-dependent learning and power consumption minimization. *Journal of Cleaner Production*, *249*, 119428. Retrieved from https://www.sciencedirect.com/science/article/pii/S0959652619342982

https://doi.org/10.1016/j.jclepro.2019.119428

Sörensen, K. (2013, February). Metaheuristics-the metaphor exposed. *International Transactions in Operational Research*, *22*(1), 3–18.

Retrieved from https://doi.org/10.1111/itor.12001

10.1111/itor.12001

Srivastava, B. (1998, August). An effective heuristic for minimising makespan on unrelated parallel machines. *The Journal of the Operational Research Society*, *49*(8), 886. Retrieved from https://doi.org/10.2307/3009970

10.2307/3009970

Strohhecker, J., Hamann, M., Thun, J.-H. (2016). Loading and sequencing heuristics for job scheduling on two unrelated parallel machines with long, sequence-dependent set-up times. *International Journal of Production Research*, *54*(22), 6747-6767. Retrieved from https://doi.org/10.1080/00207543.2016.1173248 https://arxiv.org/abs/https://doi.org/10.1080/00207543.2016.1173248
10.1080/00207543.2016.1173248

Suresh, V., & Chaudhuri, D. (1994). Minimizing maximum tardiness for unrelated parallel machines. *International Journal of Production Economics*, *34*(2), 223-229. Retrieved from https://www.sciencedirect.com/science/article/pii/0925527394900388

https://doi.org/10.1016/0925-5273(94)90038-8

Suresh, V., & Chaudhuri, D. (1996). Bicriteria scheduling problem for unrelated parallel machines. *Computers & Industrial Engineering*, *30*(1), 77-82. Retrieved from https://www.sciencedirect.com/science/article/pii/0360835295000283

https://doi.org/10.1016/0360-8352(95)00028-3

SURESH, V., & GHAUDHURI, D. (1996). Scheduling of unrelated parallel machines when machine availability is specified. *Production Planning & Control*, *7*(4), 393-400. Retrieved from https://doi.org/10.1080/09537289608930367 https://arxiv.org/abs/https://doi.org/10.1080/09537289608930367
10.1080/09537289608930367

Taboada, H.A., & Coit, D.W. (2008). Multi-objective scheduling problems: Determination of pruned pareto sets. *IIE Transactions*, *40*(5), 552-564. Retrieved from https://doi.org/10.1080/07408170701781951 https://arxiv.org/abs/https://doi.org/10.1080/07408170701781951
10.1080/07408170701781951

Tamaki, H., Hasegawa, Y., Kozasa, J., Araki, M. (1993). Application of search methods to scheduling problem in plastics forming plant: a binary

representation approach. *Proceedings of 32nd ieee conference on decision and control* (p. 3845-3850 vol.4). 10.1109/CDC.1993.325943

Tavakkoli-Moghaddam, R., Taheri, F., Bazzazi, M., Izadi, M., Sassani, F. (2009). Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints. *Computers & Operations Research*, *36*(12), 3224-3230. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054809000549 (New developments on hub location)

https://doi.org/10.1016/j.cor.2009.02.012

Terzi, M., Arbaoui, T., Yalaoui, F., Benatchba, K. (2020). Solving the unrelated parallel machine scheduling problem with setups using late acceptance hill climbing. N.T. Nguyen, K. Jearanaitanakij, A. Selamat, B. Trawiński, & S. Chittayasothorn (Eds.), *Intelligent information and database systems* (pp. 249–258). Cham: Springer International Publishing.

Torabi, S., Sahebjamnia, N., Mansouri, S., Bajestani, M.A. (2013). A particle swarm optimization for a fuzzy multi-objective unrelated parallel machines scheduling problem. *Applied Soft Computing*, *13*(12), 4750-4762. Retrieved from https://www.sciencedirect.com/science/article/pii/S156849461300286X

https://doi.org/10.1016/j.asoc.2013.07.029

Tozzo, E., Cotrim, S.L., Galdamez, E.V.C., Leal, G.C.L. (2018, September). A genetic algorithm and variable neighborhood search for the unrelated parallel machine scheduling problem with sequence dependent setup time. *Acta Scientiarum. Technology*, *40*(1), 36607. Retrieved from https://doi.org/10.4025/actascitechnol.v40i1.36607

10.4025/actascitechnol.v40i1.36607

Tseng, L.-Y., Chin, Y.-H., Wang, S.-C. (2009). A minimized makespan scheduler with multiple factors for grid computing systems. *Expert Systems with Applications*, *36*(8), 11118-11130. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417409002401

https://doi.org/10.1016/j.eswa.2009.02.071

Ulaga, L., Đurasević, M., Jakobović, D. (2022). Local search based methods for scheduling in the unrelated parallel machines environment. *Expert Systems with Applications*, 116909. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417422003463

https://doi.org/10.1016/j.eswa.2022.116909

Đurasevic, M., & Jakobovic, D. (2016). Comparison of solution representations for scheduling in the unrelated machines environment. *2016 39th international convention on information and communication technology, electronics and microelectronics (mipro)* (p. 1336-1342). 10.1109/MIPRO.2016.7522347

Đurasević, M., & Jakobović, D. (2017a, April). Comparison of ensemble learning methods for creating ensembles of dispatching rules for the unrelated machines environment. *Genetic Programming and Evolvable Machines*, *19*(1-2), 53–92. Retrieved from https://doi.org/10.1007/s10710-017-9302-3

10.1007/s10710-017-9302-3

Đurasević, M., & Jakobović, D. (2017b, September). Evolving dispatching rules for optimising many-objective criteria in the unrelated machines environment. *Genetic Programming and Evolvable Machines*, *19*(1-2), 9–51. Retrieved from https://doi.org/10.1007/s10710-017-9310-3

10.1007/s10710-017-9310-3

Đurasević, M., & Jakobović, D. (2019, May). Creating dispatching rules by simple ensemble combination. *Journal of Heuristics*, *25*(6), 959–1013. Retrieved from https://doi.org/10.1007/s10732-019-09416-x

10.1007/s10732-019-09416-x

Đurasević, M., & Jakobović, D. (2020, August). Automatic design of dispatching rules for static scheduling conditions. *Neural Computing and Applications*, *33*(10), 5043–5068. Retrieved from https://doi.org/10.1007/s00521-020-05292-w

10.1007/s00521-020-05292-w

Đurasevic, M., & Jakobovic, D. (2020). Comparison of schedule generation schemes for designing dispatching rules with genetic programming in the unrelated machines environment. *Applied Soft Computing*, *96*, 106637. Retrieved from

https://www.sciencedirect.com/science/article/pii/S1568494620305755

https://doi.org/10.1016/j.asoc.2020.106637

Đurasevic, M., Jakobovic, D., Knezevic, K. (2016). Adaptive scheduling on unrelated machines with genetic programming. *Applied Soft Computing*, *48*, 419-430. Retrieved from https://www.sciencedirect.com/science/article/pii/S1568494616303519

https://doi.org/10.1016/j.asoc.2016.07.025

Vallada, E., & Ruiz, R. (2011). A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *European Journal of Operational Research*, *211*(3), 612-622. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221711000142

https://doi.org/10.1016/j.ejor.2011.01.011

Vallada, E., Villa, F., Fanjul-Peyro, L. (2019). Enriched metaheuristics for the resource constrained unrelated parallel machine scheduling problem. *Computers & Operations Research*, *111*, 415-424. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054819301923

https://doi.org/10.1016/j.cor.2019.07.016

Van, B.K., & Hop, N.V. (2021). Genetic algorithm with initial sequence for parallel machines scheduling with sequence dependent setup times based on earliness- tardiness. *Journal of Industrial and Production Engineering*, *38*(1), 18-28. Retrieved from https://doi.org/10.1080/21681015.2020.1829111 https://arxiv.org/abs/https://doi.org/10.1080/21681015.2020.1829111
10.1080/21681015.2020.1829111

Van Hop, N., & Nagarur, N.N. (2004). The scheduling problem of pcbs for multiple non-identical parallel machines. *European Journal of Operational Research*, *158*(3), 577-594. Retrieved from https://www.sciencedirect.com/science/article/pii/S037722170300376X

https://doi.org/10.1016/S0377-2217(03)00376-X

Villa, F., Vallada, E., Fanjul-Peyro, L. (2018). Heuristic algorithms for the unrelated parallel machine scheduling problem with one scarce additional resource. *Expert Systems with Applications*, *93*, 28-38. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417417306607

https://doi.org/10.1016/j.eswa.2017.09.054

Vlašic, I., Đurasevic, M., Jakobovic, D. (2019). Improving genetic algorithm performance by population initialisation with dispatching rules. *Computers & Industrial Engineering*, *137*, 106030. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835219304899

https://doi.org/10.1016/j.cie.2019.106030

Vlašic, I., Đurasevic, M., Jakobovic, D. (2020). A comparative study of solution representations for the unrelated machines environment. *Computers & Operations Research*, *123*, 105005. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054820301222

https://doi.org/10.1016/j.cor.2020.105005

Vredeveld, T., & Hurkens, C. (2002, May). Experimental comparison of approximation algorithms for scheduling unrelated parallel machines. *INFORMS Journal on Computing*, *14*(2), 175–189. Retrieved from https://doi.org/10.1287/ijoc.14.2.175.119

10.1287/ijoc.14.2.175.119

Wang, H., & Alidaee, B. (2019). Effective heuristic for large-scale unrelated parallel machines scheduling problems. *Omega*, *83*, 261-274. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305048318302081

https://doi.org/10.1016/j.omega.2018.07.005

Wang, I.-L., Wang, Y.-C., Chen, C.-W. (2012, May). Scheduling unrelated parallel machines in semiconductor manufacturing by problem reduction and local search heuristics. *Flexible Services and Manufacturing Journal*, *25*(3), 343–366. Retrieved from https://doi.org/10.1007/s10696-012-9150-7

10.1007/s10696-012-9150-7

Wang, L., Wang, S., Zheng, X. (2016). A hybrid estimation of distribution algorithm for unrelated parallel machine scheduling with sequence-dependent setup times. *IEEE/CAA Journal of Automatica Sinica*, *3*(3), 235-246.

10.1109/JAS.2016.7508797

Wang, M., & Pan, G. (2019). A novel imperialist competitive algorithm with multi-elite individuals guidance for multi-object unrelated parallel machine scheduling problem. *IEEE Access*, *7*, 121223-121235.

10.1109/ACCESS.2019.2937747

Wang, M.-Z., Zhang, L.-L., Choi, T.-M. (2020). Bi-objective optimal scheduling with raw material's shelf-life constraints in unrelated parallel machines production. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *50*(11), 4598-4610.

10.1109/TSMC.2018.2855700

Wang, W.-L., Wang, H.-Y., Zhao, Y.-W., Zhang, L.-P., Xu, X.-L. (2013). Parallel machine scheduling with splitting jobs by a hybrid differential evolution algorithm. *Computers & Operations Research*, *40*(5), 1196-1206. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305054812002766

https://doi.org/10.1016/j.cor.2012.12.007

Wang, X., Li, Z., Chen, Q., Mao, N. (2020). Meta-heuristics for unrelated parallel machines scheduling with random rework to minimize expected total weighted tardiness. *Computers & Industrial Engineering*, *145*, 106505. Retrieved from https://www.sciencedirect.com/science/article/pii/S0360835220302394

https://doi.org/10.1016/j.cie.2020.106505

Weng, M.X., Lu, J., Ren, H. (2001). Unrelated parallel machine scheduling with setup consideration and a total weighted completion time objective. *International Journal of Production Economics*, *70*(3), 215-226. Retrieved from https://www.sciencedirect.com/science/article/pii/S0925527300000669

https://doi.org/10.1016/S0925-5273(00)00066-9

Wolpert, D., & Macready, W. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67-82.

10.1109/4235.585893

Wotzlaw, A. (2007). *Scheduling unrelated parallel machines- algorithms, complexity, and performance*. Saarbrücken, DEU: VDM Verlag.

Wu, L., & Wang, S. (2018). Exact and heuristic methods to solve the parallel machine scheduling problem with multi-processor tasks. *International Journal of Production Economics*, *201*, 26-40. Retrieved from https://www.sciencedirect.com/science/article/pii/S0925527318301683

https://doi.org/10.1016/j.ijpe.2018.04.013

Wu, M.-Y., & Shu, W. (2001). A high-performance mapping algorithm for heterogeneous computing systems. *Proceedings 15th international parallel and distributed processing symposium. ipdps 2001* (p. 6 pp.-). 10.1109/IPDPS.2001.925020

Wu, X., & Che, A. (2019). A memetic differential evolution algorithm for energy-efficient parallel machine scheduling. *Omega*, *82*, 155-165. Retrieved from https://www.sciencedirect.com/science/article/pii/S0305048317307922

https://doi.org/10.1016/j.omega.2018.01.001

Xhafa, F., Barolli, L., Durresi, A. (2007, 01). Batch mode scheduling in grid systems. *IJWGS*, *3*, 19-37.

10.1504/IJWGS.2007.012635

Xu, S., & Bean, J.C. (2007). A genetic algorithm for scheduling parallel non-identical batch processing machines. *2007 ieee symposium on computational intelligence in scheduling* (p. 143-150). 10.1109/SCIS .2007.367682

Xu, X., Ma, Y., Zhou, Z., Zhao, Y. (2015). Customer order scheduling on unrelated parallel machines to minimize total completion time. *IEEE Transactions on Automation Science and Engineering*, *12*(1), 244-257.

10.1109/TASE.2013.2291899

Xue, Y., Jiang, P., Neri, F., Liang, J. (2021, July). A multi-objective evolutionary approach based on graph-in-graph for neural

architecture search of convolutional neural networks. *International Journal of Neural Systems*, *31*(09), 2150035. Retrieved from https://doi.org/10.1142/s0129065721500350

10.1142/s0129065721500350

Xue, Y., Tang, Y., Xu, X., Liang, J., Neri, F. (2022). Multi-objective feature selection with missing data in classification. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *6*(2), 355-364.

10.1109/TETCI.2021.3074147

Xue, Y., Wang, Y., Liang, J., Slowik, A. (2021). A self-adaptive mutation neural architecture search algorithm based on blocks. *IEEE Computational Intelligence Magazine*, *16*(3), 67-78.

10.1109/MCI.2021.3084435

Yang-Kuei, L., & Chi-Wei, L. (2013, February). Dispatching rules for unrelated parallel machine scheduling with release dates. *The International Journal of Advanced Manufacturing Technology*, *67*(1-4), 269–279. Retrieved from https://doi.org/10.1007/s00170-013-4773-8

10.1007/s00170-013-4773-8

Yepes-Borrero, J.C., Perea, F., Ruiz, R., Villa, F. (2021). Bi-objective parallel machine scheduling with additional resources during setups. *European Journal of Operational Research*, *292*(2), 443-455. Retrieved from https://www.sciencedirect.com/science/article/pii/S0377221720309450

https://doi.org/10.1016/j.ejor.2020.10.052

Yepes-Borrero, J.C., Villa, F., Perea, F., Caballero-Villalobos, J.P. (2020). Grasp algorithm for the unrelated parallel machine scheduling problem with setup times and additional resources. *Expert Systems with Applications*, *141*, 112959. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417419306773

https://doi.org/10.1016/j.eswa.2019.112959

Yildirim, M., Duman, E., Krishnan, K., Senniappan, K. (2007, 01). Parallel machine scheduling with load balancing and sequence dependent setups. *International Journal of Operations Research (Taichung)*, *1*.

Ying, K.-C., Lee, Z.-J., Lin, S.-W. (2010, November). Makespan minimization for scheduling unrelated parallel machines with setup times. *Journal of Intelligent Manufacturing*, *23*(5), 1795–1803. Retrieved from https://doi.org/10.1007/s10845-010-0483-3

10.1007/s10845-010-0483-3

Ying, K.-C., & Lin, S.-W. (2012, 05). Unrelated parallel machine scheduling with sequence-and machine-dependent setup times and due date constraints. *International Journal of Innovative Computing, Information and Control*, *8*, 3279-3297.

Yu, L., Shih, H.M., Pfund, M., Carlyle, W.M., Fowler, J.W. (2002). *IIE Transactions*, *34*(11), 921–931. Retrieved from https://doi.org/10.1023/a:1016185412209

10.1023/a:1016185412209

Zarook, Yaser, Rezaeian, Javad, Mahdavi, Iraj, Yaghini, Masoud. (2021). Efficient algorithms to minimize makespan of the unrelated parallel batch-processing machines scheduling problem with unequal job ready times. *RAIRO-Oper. Res.*, *55*(3), 1501-1522. Retrieved from https://doi.org/10.1051/ro/2021062

10.1051/ro/2021062

Zeidi, J.R., & MohammadHosseini, S. (2015, May). Scheduling unrelated parallel machines with sequence-dependent setup times. *The International Journal of Advanced Manufacturing Technology*, *81*(9-12), 1487–1496. Retrieved from https://doi.org/10.1007/s00170-015-7215-y

10.1007/s00170-015-7215-y

Zhang, L., Deng, Q., Lin, R., Gong, G., Han, W. (2021). A combinatorial evolutionary algorithm for unrelated parallel machine scheduling problem with sequence and machine-dependent setup times, limited worker resources and learning effect. *Expert Systems with Applications*, *175*, 114843. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417421002840

https://doi.org/10.1016/j.eswa.2021.114843

Zhang, Z., Zheng, L., Weng, M.X. (2006, August). Dynamic parallel machine scheduling with mean weighted tardiness objective by q-learning. *The*

International Journal of Advanced Manufacturing Technology, *34*(9-10), 968–980. Retrieved from https://doi.org/10.1007/s00170-006-0662-8

10.1007/s00170-006-0662-8

Zheng, X.-L., & Wang, L. (2018). A collaborative multiobjective fruit fly optimization algorithm for the resource constrained unrelated parallel machine green scheduling problem. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *48*(5), 790-800.

10.1109/TSMC.2016.2616347

Zhou, H., Li, Z., Wu, X. (2007). Scheduling unrelated parallel machine to minimize total weighted tardiness using ant colony optimization. *2007 ieee international conference on automation and logistics* (p. 132-136). 10.1109/ICAL.2007.4338544

Zhou, S., Xie, J., Du, N., Pang, Y. (2018). A random-keys genetic algorithm for scheduling unrelated parallel batch processing machines with different capacities and arbitrary job sizes. *Applied Mathematics and Computation*, *334*, 254-268. Retrieved from https://www.sciencedirect.com/science/article/pii/S0096300318303424

https://doi.org/10.1016/j.amc.2018.04.024

Đurasevic, M., & Jakobovic, D. (2018). A survey of dispatching rules for the dynamic unrelated machines environment. *Expert Systems with Applications*, *113*, 555-569. Retrieved from https://www.sciencedirect.com/science/article/pii/S0957417418304159

https://doi.org/10.1016/j.eswa.2018.06.053

Özpeynirci, S., Gökgür, B., Hnich, B. (2016). Parallel machine scheduling with tool loading. *Applied Mathematical Modelling*, *40*(9), 5660-5671. Retrieved from https://www.sciencedirect.com/science/article/pii/S0307904X16000093

https://doi.org/10.1016/j.apm.2016.01.006

Đurasević, M., & Jakobović, D. (2022). Selection of dispatching rules evolved by genetic programming in dynamic unrelated machines scheduling based on problem characteristics. *Journal of Computational Science*, *61*, 101649. Retrieved from

https://www.sciencedirect.com/science/article/pii/S1877750322000667

https://doi.org/10.1016/j.jocs.2022.101649