Journal of Turkish

Operations Management

Two new parameter proposals for variable neighborhood descent algorithm to minimize C*max* in machine scheduling problem

Günay KILIÇ[1*], Arzu ORGAN[2]
[1]Pamukkale Üniversitesi Rektörlük, Denizli, Turkiye
gkilic@pau.edu.tr, ORCID No: http://orcid.org/0000-0003-2236-7535
[2] Pamukkale Üniversitesi İİBF, Denizli, Turkiye
aorgan@pau.edu.tr, ORCID No: http://orcid.org/0000-0002-2400-4343
*Sorumlu Yazar

Article Info

Article History:

Received: 12.07.2024
Revised: 10.02.2025
Accepted: 11.02.2025

Keywords:

Variable Neighborhood Descent,
Metaheuristic,
Unrelated Parallel Machine
Scheduling Problem with Sequence-
Dependent Setup Times

Abstract

Scheduling can be defined as the assignment of jobs to machines that will be processed under certain constraints and measurements. Different scheduling problems arise when creating schedules and assigning jobs to machines. The problem of interest in this study is the Unrelated Parallel Machine Scheduling Problem with Sequence-Dependent Setup Times (UPMSPST), which is classified as NP-hard. The objective of the problem is to minimize the makespan (Cmax). In UPMSPST, machines have different processing times for jobs and there are machine-dependent setup times between jobs. Exact solution methods are not sufficient for solving the UPMSPST and many metaheuristics have been proposed by researchers to find approximate solutions.
The aim of this paper is to propose two new parameters for solving the UPMSPST with the Variable Neighborhood Descent (VND) algorithm, which is a single-solution metaheuristic algorithm, in order to find better solutions. The proposed parameters are tested in four different scenarios and the results of the best parameter configuration are given. The results show that the proposed new parameters are effective in improving the existing solution.

1. Introduction

Scheduling is a highly significant process that affects the efficiency of the production process in many manufacturing problems and must be well-planned by businesses. This process varies from one business to another. These differences lead to variations in machines and tasks and result in different scheduling problems. Changes in workshops, machines, and tasks have led to the emergence of many scheduling problems. The goal of scheduling is to optimize one or more objectives when assigning a resource to a task. If the resources are the machines in a workshop, the tasks can be described as the jobs in a production process. Scheduling problems can be classified based on the arrival of jobs and the number of machines processing the jobs. Deterministic problems in the classification based on the arrival of jobs are delay-free situations where no machine stops. There is no change in the number of jobs or machines throughout the scheduling process (Pinedo, 2008). When classified by the number of machines, scheduling problems can be divided into single-machine models and multiple-machine models. In environments with multiple machines, there are three fundamental models: parallel systems, serial systems, and job shop systems (Baker and Trietsch, 2018). Unrelated machines in the parallel machine scheduling category refer to machines that perform the same function but have different capabilities and capacities. The use of an appropriate number of machines with different capabilities and capacities increases flexibility in businesses, making it widely used in many sectors such as textiles, chemicals, and electronics manufacturing (Lin et al., 2011). Unrelated Parallel Machine Scheduling Problem with Sequence-Dependent Setup Time (UPMSPSDST), the completion times of each job on each machine differ. In this problem, machines operate with different capacities according to the jobs. Each job also requires a setup time on the machine before processing. The setup times for jobs vary depending on the last job processed on the machine or the first job processed on that machine. The problem addressed in this study involves unrelated multiple machines in the UPMSPSDST. The processing times of machines for jobs are different. Additionally, each job has different setup times depending on the machines and the jobs assigned previously. The simpler version of this problem, the identical parallel machine scheduling problem without setup times, is an NP-hard problem when the number of machines is 2 (Arnaout, 2020). Therefore,

this problem is also NP-hard. The goal of this problem is to minimize the maximum completion time (Cmax) resulting from the scheduling. Exact solution methods are insufficient for solving NP-hard problems. Metaheuristic methods are employed to solve problems in this class. Metaheuristic methods aim to improve a candidate solution iteratively based on given criteria. Although they do not guarantee an exact solution, metaheuristic algorithms can solve optimization problems that other optimization techniques cannot effectively solve (Al-Salem, 2004). The aim of this study is to enhance the Variable Neighborhood Descent (VND) algorithm, a single-solution metaheuristic algorithm, and test two newly proposed parameters under four different scenarios to find higher quality solutions for the UPMSPSDST. The results of the proposed new parameters have been provided according to the scenarios, and it was found that the proposed new parameters are effective in improving the solution.

2. Mathematical Model

The scheduling problem is a combinatorial problem and it can be modeled as an integer programming problem (Toragay ve Pouya, 2023). In addition to the heuristic solutions developed for the solution of the UPMSPSDST problem, an integer programming model is also presented (Anagnostopoulos & Rabadi, 2002). The steps and formulas of the presented model are given below.

$$Minimize \quad C_{max} \tag{1}$$

subject to

$$\sum_{\substack{i=0 \\ i\neq j}}^{n}\sum_{k=1}^{m} X_{i,j,k} = 1 \qquad \forall j = 1,\dots,n \tag{2}$$

$$\sum_{\substack{i=0 \\ i\neq h}}^{n} X_{i,j,k} - \sum_{\substack{j=0 \\ j\neq h}}^{n} x_{h,j,k} = 0 \qquad \forall h = 1,\dots,n, \forall k = 1,\dots,m \tag{3}$$

$$C_j \geq C_i + \sum_{k=1}^{m} X_{h,j,k}\left(S_{i,j,k} + P_{j,k}\right) + HV\left(\sum_{k=1}^{m} X_{i,j,k} - 1\right) \qquad \forall h = 1,\dots,n, \forall i = 0,\dots,n, \forall j = 1,\dots,n \tag{4}$$

$$C_j \leq C_{max} \;\; \forall j = 1,\dots,n \tag{5}$$

$$\sum_{j=0}^{n} X_{0,j,k} = 1 \quad \forall k = 1,\dots,m \tag{6}$$

$$C_j \geq 0 \;\; \forall j = 1,\dots,n \tag{7}$$

$$C_0 = 0 \tag{8}$$

$$X_{i,j,k} \in \{0,1\} \;\; \forall i = 0,\dots,n, \forall j = 0,\dots,n, \quad \forall k = 1,\dots,m \tag{9}$$

Where,

n: Number of jobs

m: Number of machines

C_j: Completion time of job j

$P_{j,k}$: processing time of job j on machine k

$S_{i,j,k}$: Setup time if job j is scheduled directly after job i on machine k

$S_{0,j,k}$: Setup time if job j is scheduled to go first on machine k

$X_{i,j,k}$: 1 if job j is scheduled directly after job i on machine k and 0 otherwise

$X_{0,j,k}$: 1 if job j is scheduled first on machine k and 0 otherwise

$X_{j,0,k}$: 1 if job j is scheduled last on machine k and 0 otherwise

HV: a large positive integer

The goal (1) is to minimize the makespan. Constraints (2) ensure that each job is assigned to only one machine and scheduled once. Constraints (3) make sure that each job has a predecessor and a successor. Constraints (4) are responsible for calculating the completion times and prevent any job from being both the predecessor and the successor of the same job. Constraints (5) define the makespan. Constraints (6) ensure that no job is scheduled as the first job more than once. There's no need for additional constraints to guarantee only one job is scheduled last,

as this is already covered by constraints (6) and (3) together. Constraints (7) and (8) ensure that completion times are non-negative and that the dummy job's completion time is zero, respectively. Finally, constraints (9) state that the decision variable x is binary across all domains.

3. Literature Review

The scheduling problem is encountered in many production areas. The differences in these problems have led to the formation of various objective functions. One of the common objective functions is to minimize Cmax. Table 1 shows other academic studies that use same benchmark dataset and used algorithms.

Table 1. Previous Studies used same benchmark

Study No	Study	Algorithm	Compared Study
1	Al-Salem (2004)	PH (Partitioning Heuristic)	-
2	Rabadi et al. (2006)	Meta-RaPS	1
3	Helal et al. (2006)	TS	1
4	Arnaout et al. (2010)	Ant Colony Optimization	1,2,3
5	Chang and Chen (2011)	SA and GA	1
6	Ying et al. (2012)	Restricted TS	1,2,3,4
7	Fleszar et al. (2012)	Multi-Start VND	2,4
8	Lin ve Ying (2014)	Artificial Bee Colony Hybrid Artificial Bee Colony	1,2,3,4,5
9	Arnaout et al. (2014)	Ant Colony Optimization	1,2,3,4
10	Yılmaz Eroğlu et al. (2014)	GA+ Local Search	4,5
11	Cota et al. (2017)	Automata - Adaptive Large Neighborhood Search	4,9
12	De Abreu, L. R., and De Athayde (2020)	GA+VND+SA (GIVP)	4,5,10
13	Arnaout (2020)	Worm Optimization Algorithm	3,6,8,9,10

As seen in Table 1, the tested work has attracted attention from many researchers and different solutions have been developed with different algorithms. These developed solutions have been compared with each other. For solving this problem, single-solution methods such as Tabu Search and Simulated Annealing have been used, as well as population-based metaheuristic algorithms such as Ant Colony Optimization, Worm Optimization Algorithm, Genetic Algorithm, and Artificial Bee Colony Algorithm. Additionally, hybrid algorithms that combine both single-solution and population-based algorithms have also been employed. The methods developed by researchers have achieved successful results in the test benchmark.

The unrelated parallel machine scheduling problem has also been the subject of academic studies on different objective functions and different data sets. Some of recent studies are as follows.

In the study by Zhang et al. (2021), a new model for the UPMSPST problem was proposed by incorporating limited workforce and the learning effects of individuals. To solve this proposed model, a combinatorial evolutionary algorithm (CEA) was presented, which includes list scheduling (LS), the shortest setup time (SST) priority rule, and the earliest completion time (ECT) priority rule. For testing the algorithm, 72 instances were created, and the Taguchi method was used to determine the best parameter combinations. The proposed algorithm was compared with other algorithms and found to be successful.

In the study by Berthier et al. (2022), they focused on a real industrial problem in a textile factory. Two additional features were incorporated into the parallel machine scheduling problem with sequence-dependent setup times: machine eligibility and limited resources. The study aimed to balance the workload by addressing the dynamic layout between jobs in the workshop and determining machine groups. A mathematical model of the problem was developed, and solutions were obtained using a genetic algorithm.

In the study by Sanati et al. (2023), the authors address the unrelated parallel machine energy-efficient scheduling problem with sequence-dependent setup times, considering various energy consumption tariffs. The study evaluates setup times in two different ways: separately and jointly with processing times, developing mixed-integer linear programming models for both scenarios. In the models where setup times are separate, solutions were obtained for problems involving up to 16 machines and 45 jobs, while for the joint setup time models, solutions were achieved for up to 20 machines and 40 jobs. Additionally, a fix-and-relax heuristic was developed for large-scale problems, enabling solutions for instances ranging from 20 to 100 jobs.

In their study, Saraç and Özçelik (2023) focus on a multi-objective problem in the unrelated parallel machine scheduling problem, aiming to minimize both total completion time and total tardiness. To achieve this, they

propose an alternative metaheuristic algorithm to the Augmented ε-constraint (AEC) algorithm. The authors emphasize that the proposed algorithm outperforms the AEC algorithm.
In the study, Ramos-Figueroa et al (2023) they aimed to solve the unrelated parallel machine scheduling problem using a Group Genetic Algorithm, an extension of the standard Genetic Algorithm. They achieved a 52% improvement over the original mutation operator by introducing a new mutation operator called 2-Items.
In their study, Elyasi et al.(2024) focus on an unrelated parallel machine scheduling problem related to Vestel. This problem aims to minimize both early and tardy completion times in a production environment with unrelated parallel machines. To achieve this, they employed the Imperial Competitive Algorithm, and the results showed that their algorithm outperformed the current state of the art by 39%, as well as surpassing the best algorithms in the literature by 23% and 12%.
In their study, Fonseca et al. (2024) proposed a "fix and optimize" algorithm for the unrelated parallel machine scheduling problem. They aimed to solve subproblems grouped by subsets of machines. To address these subproblems, they introduced a mathematical program called MPA, which was used to find solutions. Computational experiments demonstrated that the proposed approach provided significantly higher-quality solutions compared to the standalone exact algorithm. When compared to the best heuristic approaches in the literature, the proposed algorithm performed better in instances with a high number of jobs per machine, but exhibited lower performance in instances with fewer jobs per machine.
In their 2024 study, Muñoz-Díaz et al. proposed an unrelated parallel machine problem inspired by a station in the wind tower production process, which includes setup times for the machines. They also introduced a new feature involving supportive machines that assist those with capacity shortages. For this problem, a mixed-integer linear programming model was formulated, and solutions were proposed using three different algorithms: a constructive heuristic, simulated annealing, and tabu search. The results of these algorithms were compared to evaluate their effectiveness.

The aim of this study is to develop a proposed VND algorithm (VND_KMP_SP), which is a single-solution metaheuristic, with two new parameters, and to test it on a commonly used problem, instead of proposing an algorithm to be compared with hybrid or population-based algorithms.

4. Variable Neighborhood Descent Algorithm

The Variable Neighborhood Search (VNS) algorithm is based on the idea of systematically changing predefined neighborhood structures to search for solutions (Mladenović and Hansen, 1997). In this study, the proposed algorithm with new parameters is the Variable Neighborhood Descent (VND) algorithm, which is the deterministic version of the VNS algorithm and does not include a local search phase (Hansen and Mladenovic, 2003). The VND algorithm has been used as the local search phase in many hybrid metaheuristic algorithms, including scheduling problems (Gao et al., 2008; Haupt and Haupt, 2004). The basic form of the VND algorithm is presented below.

Algorithm VND

input: x = initial solution, neighborhood structures N_k, $k = 1, \ldots, k_{max}$

output: *x:* Final Scheduled Jobs

1. **procedure** VND (x, N_k)
2. **repeat** until stopping criterion is met:
3. $k = 1$
4. $k = k_{max}$ repeat until:
5. $x' = N_k(x)$
6. **if** ($x'$is better then x):
7. $x = x'$, $k = 1$
8. **else**:
9. $k=k+1$;
10. **return** x
11. **finish**.

VND algorithm aims to search for a better solution by swapping jobs within neighborhood structures. In solving scheduling problems, typically five different neighborhood structures are used. Three of these involve intra-machine swaps, while the other two involve inter-machine job swaps. The intra-machine swaps consist of swapping two jobs on the same machine (K1), inserting one job after another job on the same machine (K2), and reversing the order of two jobs on the same machine (K3). The inter-machine job swaps involve swapping jobs between two different machines (K4) and inserting a job from one machine after a job on another machine (K5) (Wang et al., 2016).

5.Proposed adapted VND Algorithm

VND algorithm requires an initial solution and the predefinition of neighborhood structures. In this study, five different neighborhood structures, as outlined in Section 3, have been utilized, and a balanced-random initial algorithm has been proposed for generating the initial solution.

5.1 Initial Solution (Balanced- Random)

The proposed balanced-random initialization algorithm aims to distribute jobs across machines such that all machines are assigned an equal number of jobs. When a randomly selected job is assigned to a machine, the machine with the fewest jobs is chosen from the available machines. If there is more than one machine with the minimum load, the selection among these machines is made randomly.

Algorithm Balanced- Random

input: *N*: All Jobs, *M*: All machines *UM*: Available machines

output: *x:* Final scheduling

```
1. procedure balaced-random (N, M, UM)
2. N, M, UM ← M, m=j=L=Ø
3.   repeat until N is empty:
4.       if UM is empty:
5.           UM ←M
6.       j=random.choise (N)
7.       m= random.choise (UM)
8.       x.append(m,j)
9.       N.remove(j)
10.      UM.remove(m)
11. return x
12. finish.
```

The balanced-random algorithm creates an initial solution for testing the parameters proposed to the VND algorithm. The initial solution is the same for each scenario. In this way, the proposed parameters were tested with the same initial solutions in different scenarios.

5.2 New parameters for VND

In this study, two new parameters are proposed for the VND algorithm. These are the critical machine selection parameter and the constraint parameter.

5.2.1 Critical machine selection parameter

In scheduling problems where the objective function is Cmax, the machine with the highest load determines the Cmax value. This machine can be referred to as the critical machine. When selecting two machines randomly during neighborhood changes, if one of these machines is not the critical machine, the Cmax value does not decrease even if the workloads of the selected machines change when the neighborhood structure is applied. For this reason, a critical machine parameter (kmp) has been added to the neighborhood structures found in the literature. When the kmp parameter is 1, if the number of affected machines is one in the neighborhood structure

to be applied, it is ensured that this machine is the critical machine. If the number of affected machines is two, one of these machines is ensured to be the critical machine. This parameter aims to reduce the Cmax value as soon as possible in the neighborhood structure. The kmp parameter aims to more effectively explore the solution space.

5.2.2 Constraint parameter

The study proposes a constraint parameter (sp) to constrain the tasks that can change in neighborhood structures. This parameter can take values in the range sp = [0- N/2) where N is the total number of jobs. During neighborhood changes, the restriction parameter sp can limit the change to all tasks in the list or only to those on the critical machine. If only the tasks on the critical machine are limited to change, the sp can take values in the range sp = [0- nkm /2), where nkm is the number of tasks on the critical machine. When sp=0, it is possible to select all jobs during the neighborhood change. For each increment of 1 in the sp parameter, the two consecutive jobs with the lowest processing and setup times among the tasks assigned to the machines are calculated. These jobs are then fixed on their assigned machine and prevented from changing.

5.2.3 Test Scenarios

By introducing new parameters to the VND algorithm, a new adapted VND algorithm called VND_KMP_SP has been proposed. These parameters have been tested with initial solutions generated using the balanced-random initialization algorithm in four different scenarios. The scenarios and parameter values, including the critical machine selection (kmp) from Section 3.2.1 and the constraint parameter (sp) from Section 3.2.2, are presented in Table 2.

Table 2. Test scenarios ve parameter values

Scenario	*kmp*	*sp*
scenario -1	0	0
scenario -2	1	0
scenario -3	1	*deg_sp*
scenario -4	1	*km_sp*

The explanations of the scenarios and their parameters given in Table 2 are as follows.

scenario-1: In this scenario, machine and job selection is kept free. All machines and jobs can be selected in the neighborhood change. This scenario represents the basic VND algorithm introduced in section 3.

scenario -2: In this scenario, one of the selected machines must be a critical machine. There is no constraint in job selection, all jobs can be chosen.

scenario -3: In machine selection, one of the selected machines must be a critical machine. In job change, change of some of the jobs is not allowed according to the sp parameter. The sp parameter varies depending on the number of iterations and the number of jobs to be prevented from changing also varies.

scenario -4: In machine selection, one of the selected machines must be a critical machine. In job change, only some of the jobs on the critical machine are not allowed to change according to the sp parameter.

A new adapted VND algorithm is proposed with the scenarios and parameters introduced above (VND_KM_SP). The parameters of the VND _KMP_SP algorithm are as follows.

Algorithm VND_KMP_SP

input: Neighborhood structures: N_k, $k = 1, \ldots, k_{max}$, stopping count: (n = 50). x = initial solution with balanced-rondom algorithm. scenario = (scenario -1, scenario -2, scenario -3, scenario -4). i =iteration number is 0 at beginig

output: *x:* Final scheduling

1. **procedure** VND_KMP_SP ($N_{k,}$ *x,* scenario, n):
2. **while** (i <= n):
3. **if** (scenario = scenario -1) **then** *sp*=0, *kmp*=0

   if (scenario = scenario -2) **then** *sp*=0, *kmp* =1

   if (scenario = scenario -3) **then** *maxsp* = n/2, *kmp* =1

   if (scenario = scenario -4) **then** *maxsp* = job count on critical machine/2, *kmp* =1

4. k =1
5. while ($k = k_{max}$):
   if (scenario = scenario -3 **or** scenario = scenario -4) **then** *sp*=mod (*i*, *maxsp*)
6. $x' = N_k(x)$
7. if (x'is better then x):
8. $x = x'$, $k = 1$
9. else:
10. $k=k+1$;
11. $i=i+1$
12. **return** x
13. **finish**.

The difference between scenario -1 and scenario -2 lies in the kmp parameter. These two scenarios allow for measuring the effectiveness of the kmp parameter without any constraints on job swaps. In scenario-3 and scenario-4, the constraint parameter (sp) starts at 0 and is modulated by the number of iterations relative to the maximum value of sp (maxsp). In this case, the selection of all jobs is random, allowing for a broader search. As the sp parameter approaches its maximum value, the swapping of jobs on the critical machine becomes more restricted to the least costly jobs, enabling a narrower, more refined search. When the number of iterations reaches the maxsp parameter, only the two most costly jobs are swapped.

6. Application

6.1 Benchmark dataset

In this study, the parameters proposed for the VND_KMP_SP algorithm were tested using the dataset created by Al-Salem (2004). This large dataset consists of 540 data files, with balanced setup and processing times. The data files involve scheduling N jobs {20, 40, 60, 80, 100, and 120} on M machines {2, 4, 6, 8, 10, and 12}. The small dataset involves scheduling N jobs {6, 7, 8, 9, 10, and 11} on M machines {2, 4, 6, and 8}.

The problem and dataset of interest have been widely recognized in academic studies, with many researchers proposing solutions using various metaheuristic methods (Arnaout, 2020; Rabadi et al., 2006; Helal et al., 2006; Arnaout et al., 2010; Chang and Chen, 2011; Ying et al., 2012; Fleszar et al., 2012; Lin and Ying, 2014; Arnaout et al., 2014; Yilmaz Eroglu et al., 2014; Cota et al., 2017). The dataset and the results of other researchers can be accessed through (SchedulingResearch, 2022).

6.2 Testing VND_KMP_SP

6.2.1 Testing VND_KMP_SP on small dataset

The proposed VND_KMP_SP algorithm was run 50 times for each data file using a balanced-random initial algorithm on a small dataset. The Cmax values obtained under different scenarios and initial solutions are grouped by the number of machines and jobs. The results are presented in Table 3.

Table 3. Average Cmax Values for VND_KMP_SP Algorithm on Small Dataset Grouped by Number of Machines and Jobs

machines	jobs	Balanced-random	scenario -1 (VND)	scenario -2	scenario -3	scenario -4
2	6	467.676	419.968	418.880	**416.512**	418.460
2	7	599.843	528.648	523.541	**522.727**	523.440
2	8	615.835	558.229	555.761	**552.149**	552.896
2	9	743.924	654.793	650.467	**645.248**	647.107
2	10	770.911	698.211	693.653	**688.008**	690.609
2	11	890.765	783.955	777.759	**769.104**	774.276
4	6	315.395	281.159	**267.207**	273.863	267.813
4	7	322.011	288.125	**276.287**	279.723	277.176
4	8	327.483	299.732	289.931	296.623	**289.591**
4	9	446.104	389.524	**373.972**	378.749	374.844
4	10	469.175	412.669	398.072	400.733	**397.380**

4	11	479.261	426.984	412.529	**410.853**	411.833
6	8	313.833	278.704	**259.276**	268.360	259.725
6	9	322.355	287.687	269.043	273.920	**268.793**
6	10	329.653	296.924	279.063	285.601	**278.844**
6	11	332.701	301.708	**285.552**	289.047	285.792
8	10	317.083	278.535	254.501	265.771	**254.341**
8	11	323.837	288.816	**265.243**	271.511	265.281

According to Table 3, The use of the proposed parameters results in average best outcomes compared to the basic VND algorithm. When the number of machines is 2, Scenario 3 is the most successful. However, as the number of machines increases, the effectiveness of Scenario 2 and Scenario 4 increases.

The proposed VND_KMP_SP algorithm has been compared with previously used algorithms on the same small dataset. For each compared algorithm, the known best results for all instances (Cmax_alg) were compared with the best results from the test scenarios of the proposed algorithm (Cmax_ scenario). The error percentages were calculated using Equation 1.

$$error\ percentages = \frac{Cmax\_alg\ - Cmax\_scenario}{Cmax\_alg} x100 \quad (10)$$

The averages of the values obtained with Equation 1 have been calculated for all instances. The average error percentages are provided in Table 4.

Table 4. Averages of Initial Solution Improvement Percentages by Number of Machines with VND_KMP_SP Algorithm

Algorithm	scenario-1	scenario-2	scenario -3	scenario -4
Meta-RaPS (Rabadi et al. (2006))	4.093111	1.738846	2.079439	1.618032
Tabu Search (Helal et al. (2006))	1.600028	**-0.66695**	**-0.3482**	**-0.78759**
Ant Colony Optimization (Arnaout et al. (2010))	4.087209	1.733748	2.074086	1.61281
Ant Colony Optimization (Arnaout et al. (2010))	1.817143	2.145203	1.675349	1.672795

When examining Table 4, it is observed that the VND_KMP_SP algorithm, run 50 times, produces better results than the commonly used Tabu Search algorithm. Additionally, it achieves results close to those of population-based algorithms such as Ant Colony Optimization. It is evident that the effectiveness of the VND_KMP_SP algorithm increases when its parameters are adjusted in the scenarios.

While VND_KMP_SP algorithm performs better than the metaheuristic Tabu Search algorithm in solving problems, it does not outperform the population-based Ant Colony Optimization algorithm. The goal of this study is not necessarily to develop a superior algorithm but to enhance the effectiveness of the basic VND algorithm with the proposed parameters.

6.2.2 Testing VND_KMP_SP on big dataset

The proposed VND _KMP_SP algorithm was run 50 times on a large dataset using a balanced-random initial algorithm for each data file. The improvement percentages of the initial solution Cmax values (Cmax_bas) compared to the Cmax values from the tested scenarios (Cmax_ scenario) were calculated using Equation 2.

$$improvement\ percentages = \frac{Cmax\_bas\ - Cmax\_scenario}{Cmax\_bas} x100 \quad (11)$$

The averages of the development percentages were calculated and grouped according to the number of jobs and machines, and the results in Table 5 and Table 6 were obtained. The best results in the tables are indicated in bold.

Table 5. Average of initial solution improvement percentages according to the number of jobs with the VND_KMP_SP algorithm

Machines	scenario -1	scenario -2	scenario -3	scenario -4
2	8.594	9.521	**11.039**	10.359
4	7.62	10.106	**11.78**	10.869
6	8.591	11.707	**12.941**	12.295
8	8.186	11.465	**12.507**	11.929

10	6.884	10.115	**10.678**	10.469
12	8.397	12.302	**12.69**	12.537

When Table 5 is examined, it can be seen that the most successful scenario in developing the balanced-random starting solution according to the number of machines is scenario-3, where one of the machines is forced to choose a critical machine and the constraint parameter is variable.

Table 6. Average of initial solution improvement percentages according to the number of jobs with the VND_KMP_SP algorithm

Jobs	scenario -1	scenario -2	scenario -3	scenario -4
20	9.533	13.87	13.501	**13.939**
40	8.809	11.978	**12.915**	12.486
60	7.834	10.429	**11.739**	11.03
80	7.663	10.062	**11.496**	10.697
100	7.794	10.114	**11.675**	10.853
120	6.638	8.763	**10.309**	9.452

When Table 6 is examined, it is seen that similar results are obtained when grouped according to the number of jobs and scenario-3 is the most successful scenario. It has been observed that when the number of jobs is low, restricting only the jobs on the critical machine (scenario-4) gives better results.

7. Conclusion

In this study, a newly adapted VND algorithm is proposed to solve the UPMSPSDST. The proposed new algorithm tests the effectiveness of two new parameters in four different scenarios. The proposed new VND_KMP_SP algorithm suggests an effective critical machine selection parameter for scheduling problems with the objective function C*max*. The proposed critical machine selection parameter has added efficiency to the machine selection and has been effective in reducing the C*max* value. The other proposed constraint parameter has been found to be beneficial in reducing the C*max* value by preventing job changes in scheduling problems with sequence-dependent setup times. The proposed VND_KMP_SP algorithm can solve the UPMSPSDST and other scheduling problems more effectively by being used in hybrid form with other metaheuristic algorithms. The effectiveness of the parameters can be investigated more extensively by adding new neighborhood structures. Also the proposed algorithm may use with local searches in the VNS algorithm. The effectiveness of the proposed algorithm can be investigated with local searches in the VNS algorithm.

Author contribution statement

In the study carried out, Günay Kılıç was involved in the formation of the idea, making the design, literature review and obtaining the results; Arzu Organ contributed to the evaluation of the results, spelling and checking the article for content.

Conflict of Interest

No conflict of interest is declared by the author.

References

Al-Salem, A. "Scheduling to minimize makespan on unrelated parallel machines with sequence dependent setup times." *Engineering Journal of the University of Qatar* 17.1 (2004): 177-187. https://scholar.google.com/scholar_lookup?title=Scheduling%20to%20minimize%20makespan%20on%20unrelated%20parallel%20machines%20with%20sequence%20dependent%20setup%20times&publication_year=2004&author=A.%20Al-Salem

Anagnostopoulos, G. C., & Rabadi, G. (2002, June). A simulated annealing algorithm for the unrelated parallel machine scheduling problem. *In Proceedings of the 5th Biannual world automation congress* (Vol. 14, pp. 115-120). IEEE. https://doi.org/10.1109/wac.2002.1049430

Arnaout, J. P. (2020). "A worm optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times." *Annals of Operations Research*, 285(1), 273-293. https://doi.org/10.1007/s10479-019-03138-w

Arnaout, J. P., Musa, R., & Rabadi, G. (2014). "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. https://doi.org/10.1007/s10845-012-0672-3

Arnaout, J. P., Rabadi, G., & Musa, R. (2010). "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. https://doi.org/10.1007/s10845-009-0246-1

Baker, K. R., & Trietsch, D. (2018). *Principles of sequencing and scheduling*. John Wiley & Sons. https://doi.org/10.1002/9781119262602

Berthier, A., Yalaoui, A., Chehade, H., Yalaoui, F., Amodeo, L., & Bouillot, C. (2022). "Unrelated parallel machines scheduling with dependent setup times in textile industry." *Computers & Industrial Engineering*, 174, 108736. https://doi.org/10.1016/j.cie.2022.108736

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

Cota, L. P., Guimarães, F. G., de Oliveira, F. B., & Souza, M. J. F. (2017, June). "An adaptive large neighborhood search with learning automata for the unrelated parallel machine scheduling problem." In 2017 *IEEE Congress on Evolutionary Computation (CEC)* (pp. 185-192). IEEE. https://doi.org/10.1109/cec.2017.7969312

Elyasi, M., Selcuk, Y. S., Özener, O. Ö., & Coban, E. (2024). "Imperialist competitive algorithm for unrelated parallel machine scheduling with sequence-and-machine-dependent setups and compatibility and workload constraints." *Computers & Industrial Engineering*, 190, 110086. https://doi.org/10.1016/j.cie.2024.110086

Fleszar, K., Charalambous, C., & Hindi, K. S. (2012). "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. https://doi.org/10.1007/s10845-011-0522-8

Fonseca, G. H., Figueiroa, G. B., & Toffolo, T. A. (2024). "A fix-and-optimize heuristic for the Unrelated Parallel Machine Scheduling Problem." *Computers & Operations Research*, 163, 106504. https://doi.org/10.1016/j.cor.2023.106504

Gao, J., Sun, L., & Gen, M. (2008). "A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems." *Computers & Operations Research*, 35(9), 2892-2907. https://doi.org/10.1016/j.cor.2007.01.001

Hansen, P., & Mladenovic, N. (2003). "A tutorial on variable neighborhood search." *Les Cahiers du GERAD* ISSN, 711, 2440. https://scispace.com/pdf/a-tutorial-on-variable-neighborhood-search-lq7nrkn4zn.pdf

Haupt RL, Haupt SE. *Practical Genetic Algorithms*. 2nd ed. New York, USA, Wiley, 2004. https://doi.org/10.1002/0471671746

Helal, M., Rabadi, G., & Al-Salem, A. (2006). "A tabu search algorithm to minimize the makespan for the unrelated parallel machines scheduling problem with setup times." *International Journal of Operations Research*, 3(3), 182-192. https://www.researchgate.net/publication/228621816_A_tabu_search_algorithm_to_minimize_the_makespan_for_the_unrelated_parallel_machines_scheduling_problem_with_setup_times

Lin, S. W., Lu, C. C., & Ying, K. C. (2011). "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. https://doi.org/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. https://doi.org/10.1016/j.cor.2014.05.013

Mladenović, N., & Hansen, P. (1997). "Variable neighborhood search." *Computers & operations research*, 24(11), 1097-1100. https://doi.org/10.1016/s0305-0548(97)00031-2

Muñoz-Díaz, M. L., Escudero-Santana, A., & Lorenzo-Espejo, A. (2024). "Solving an Unrelated Parallel Machines Scheduling Problem with machine-and job-dependent setups and precedence constraints considering Support Machines." *Computers & Operations Research*, 163, 106511. https://doi.org/10.1016/j.cor.2023.106511

Pinedo, M. L (2018). Scheduling: theory, algorithms, and systems. *Springer*. https://link.springer.com/book/10.1007/978-0-387-78935-4

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

Ramos-Figueroa, O., Quiroz-Castellanos, M., Mezura-Montes, E., & Cruz-Ramírez, N. (2023). "An experimental study of grouping mutation operators for the unrelated parallel-machine scheduling problem". *Mathematical and Computational Applications*, 28(1), 6. https://doi.org/10.3390/mca28010006

Sanati, H., Moslehi, G., & Reisi-Nafchi, M. (2023). "Unrelated parallel machine energy-efficient scheduling considering sequence-dependent setup times and time-of-use electricity tariffs." *EURO Journal on Computational Optimization*, 11, 100052. https://doi.org/10.1016/j.ejco.2022.100052

Saraç, T., & Özçelik, F. (2023). "A matheuristic algorithm for multi-objective unrelated parallel machine scheduling problem Çok amaçli ilişkisiz paralel makine çizelgeleme problemi için bir matsezgisel algoritma." *Journal of the Faculty of Engineering and Architecture of Gazi University*, 38(3). https://doi.org/10.17341/gazimmfd.873295

SchedulingResearch. (2022), Accesed 01.09.2022 from www.schedulingresearch.com.

Toragay, O., & Pouya, S. (2023). "A Monte Carlo simulation approach to the gap-time relationship in solving the job shop scheduling problem." *Journal of Turkish Operations Management* (JTOM), 7(1). https://doi.org/10.56554/jtom.1286288

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. https://doi.org/10.1109/jas.2016.7508797

Yilmaz Eroğlu, D., 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. https://doi.org/10.1080/00207543.2014.920966

Ying, K. C., Lee, Z. J., & Lin, S. W. (2012). "Makespan minimization for scheduling unrelated parallel machines with setup times." *Journal of Intelligent Manufacturing*, 23, 1795-1803. https://doi.org/10.1007/s10845-010-0483-3

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