Research Article
Journal of Intelligent Systems: Theory and Applications 4(2) (2021) 86-93
DOI: 10.38016/jista.852099

Solution Representation in Proportionate Multiprocessor Open Shop

Zeynep Adak [1]

[1]Gebze Technical University, Department of Industrial Engineering, Kocaeli, Turkey

zeynepadak@gtu.edu.tr

Abstract
Proportionate multiprocessor open shop is considered in this study. It is a shop model where a set of jobs follow no predefined route to visit several stages with at least one having two or more parallel machines to carry out the same task. Proportionality means that processing times depend only on stages and are independent of jobs, hence is defined as stage-wise. The shop model has various application areas in industry but the literature on the field is still limited. In this study, a novel solution representation scheme is proposed for the proportionate multiprocessor open shop. The scheme is based on permutation of stages and encodes the cumulative number of job assignments to a stage. The proposed scheme is shown to generate higher quality random solutions compared to the common operation permutation representation for the shop model. The approach proposed in this study to design a solution representation for a scheduling problem is a new and favorable approach that takes into account the specific machine environment, job characteristics and objective function of the problem under consideration. This way of designing solution representation schemes would increase the solution quality or decrease the computational time required in solution algorithms for scheduling problems.

Keywords: Multiprocessor open shop, solution representation, implicit-stage representation, scheduling, proportionate

Orantılı Esnek Açık Atölye Tipinde Çözüm Gösterimi

Öz
Bu çalışmada orantılı esnek açık atölye tipi ele alınmıştır. Bu atölye modelinde yapılacak işlerin işlem istasyonlarını gezerken takip edecekleri bir rota bulunmaz ve bu istasyonların en az birinde aynı işlemi yapan iki veya daha fazla paralel makine bulunur. Orantılı ifadesi işlem sürelerinin istasyona bağlı olduğunu ve işten bağımsız olduğunu ifade eder. Böylece, orantılılık istasyon-bazlı olarak tanımlanmıştır. Bu atölye modelinin endüstride farklı uygulama alanları mevcuttur ancak bu alandaki literatür hala kısıtlıdır. Bu çalışmada, orantılı esnek açık atölye tipi için yeni bir çözüm gösterimi önerilmiştir. Önerilen gösterim istasyonların permutasyonuna dayanır ve bir istasyona yapılan kümülatif iş ataması sayısını şifreler. Önerilen bu yeni gösterim, bu atölye tipinde yaygın olarak kullanılan operasyon-permütasyonu gösteriminden daha iyi kalitede rastgele çözümler üretmiştir. Bu çalışmada çizelgeleme probleminde çözüm gösterimi tasarımında kullanılan yaklaşım yeni ve sonuçlar bakımından olumludur. Bu yaklaşım eldeki probleme özel makine ortamını, iş özelliklerini ve amaç fonksiyonunu dikkate alır. Çözüm gösterimi tasarlanmasında izlenen bu yol çizelgeleme problemlerinde kullanılan çözüm algoritmalarının daha yüksek kalitede çözüme ulaşmasını veya harcanan hesaplama zamanının kısaltılmasını sağlayacaktır.

Anahtar Kelimeler: Esnek açık atölye, çözüm gösterimi, istasyon gösterimi, çizelgeleme, orantılı

1. Introduction

Solution or schedule representation is a key element in dealing with scheduling of shop models. It is a way to represent a feasible schedule for the shop. Feasibility refers to the condition that a given schedule meets all model definitions in terms of machine environment and job characteristics and does not violate any constraints imposed in the model.

Scheduling problems are one large class of combinatorial optimization problems that are thoroughly studied in the literature. It is about creating a feasible schedule for a given shop model to minimize

* Corresponding Author.
E-mail: zeynepadak@gtu.edu.tr

Recieved : 01 January 2021
Revision : 23 March 2021
Accepted : 04 April 2021

a certain objective or multiple objectives. Since most of scheduling problems are NP-Hard in nature (Chen et al., 1998) they require unreasonable amount of computational time to find an optimal solution by enumerative algorithms. Instead, approximation algorithms and more recently search algorithms are used to find a good enough solution.

Search algorithms are named as metaheuristics and they mainly make an "educated" search in the large solution space of the problem. The search is educated since it uses problem knowledge and previous search experience. The search routine works around numerous solutions until a predefined termination criterion is met. Hence, solution (schedule) representation is an initial important step in those algorithms. Since the algorithm subroutines, neighborhood structures for instance, are defined over the solution representation, choosing a proper and an efficient one may even increase the solution quality of the algorithm and decrease the computational time required.

In this study, proportionate multiprocessor open shop (MPOS) with makespan minimization criterion is considered, and a novel efficient solution representation is proposed for the shop model. Inefficiencies due to the conventional representation in the literature are pointed out and the proposed representation is introduced in detail with its encoding and decoding procedures. Efficiency of the proposed representation is presented by making a basic random search for the solution of a set of benchmark instances. The results are compared with a random search by the conventional representation and it is shown that the solution quality reached with the proposed representation is remarkably higher.

The paper is organized as follows. The proportionate MPOS is defined in Section 2 and the literature is reviewed in terms of solution representations for the problem. Section 3 presents the conventional solution representation for the problem in the literature and states the inefficiencies it causes. The proposed solution representation is introduced in Section 4. Computational experiments and comparisons are presented in Section 5. Section 6 discusses the proposed representation and the results. Conclusive remarks are given in Section 7.

2. Proportionate Multiprocessor Open Shop

Multiprocessor open shop (MPOS) is a machine environment with a set of stages (machine centers), $S = \{1, 2, \dots, s\}$, where stage i has $m_i \geq 1$ machines in parallel and at least for one stage $m_i > 1$. Every stage carries out a different task and the machines of a stage are assumed to be identical in this study. There are a number of jobs, $J = \{1,2, \dots, n\}$, to be processed in the stages, but there is no route for the jobs to visit the stages and it is what makes the shop an *open* shop. A job is processed on a single machine and a machine processes a single job at a time. Operation O_{ji} is defined as the processing of job j on machine i. In this study, it is assumed that all jobs are processed in all stages and the objective function is to minimize the makespan, the time all operations of the shop are completed.

Proportionate MPOS is considered in this study. Proportionate property for MPOS is defined by Matta (2009) and it refers to processing times being based on stages and not both on stages and jobs. More precisely, a stage i processes any job in the same amount of p_i time, hence $p_{ji} = p_i \;\; j \in J, \;\; i \in S$, where p_{ji} is the processing time of job j in stage i. Thus, the scheduling problem considered in this study is represented in three-field notation as $O(P)|\, p_{ji} = p_i | C_{max}$.

The proportionate property definition for MPOS is in contrast with the widely accepted definition for proportionate flow, job, and open shops. Proportionality was introduced as job-wise for flow shops (Ow, 1985), defining $p_{ji} = p_j$. This job-wise proportionality definition has also been accepted in job shops and open shops (Pinedo, 2016). Proportionality was taken as machine-wise in rare studies in flow, job and open shops. However, it is always considered as stage-wise, $p_{ji} = p_i$, in MPOS literature due to the real applications of proportionate MPOS, as given next.

MPOS is seen in several industrial settings such as medical testing facilities in health care, auto repair and maintenance centers, electronics manufacturing, and inspection and quality control operations. The proportionate case, on the other hand, is more common in healthcare medical testing where it takes the same amount of time to carry out a test, independent of patients.

Gonzalez and Sahni (1976) showed that $O_m||C_{max}$ is NP-Complete for $m > 2$. The result applies for MPOS, $O(P)||C_{max}$, which is the generalized version of the classical open shop. Further, Mao (1995) showed that $O_2(P_k)||C_{max}$ is NP-Complete even when there are $k = 2$ machines at each stage. However, the complexity status of the proportionate case, $O(P)|\, p_{ji} = p_i | C_{max}$, has not been studied in the literature yet.

It should be noted that despite the various industrial applications of the shop model, the literature on MPOS is still very limited. Since the current study is about solution representation in proportionate MPOS, different approaches used in the literature to represent solutions in MPOS are reviewed here. A solution for the problem should supply the job routes to visit the stages as well as the machine sequences at every stage. The most common scheme used in representing schedules of MPOS is the operation permutation representation. It was first proposed by Liaw (2000) for the classical open shop and adapted to MPOS by Matta (2009). Later, it was also used by other researchers studying in MPOS (Naderi et al., 2011, Goldansaz et al., 2013, Azadeh et al., 2014, Bai et al., 2016). Definition of the representation is given in the next section.

There were also a few other schemes used for solution representation in MPOS scheduling problem. Zhang et al. (2019) used a two-level hierarchical scheme where the first level supplied the information about job routes and the second level specified the machine to be used for the corresponding job and in the corresponding stage given by the first level. Abdelmaguid et al. (2014) used a disjunctive graph representation where every O_{ji} was represented by a separate node in the graph with a weight equal to p_{ji}. Stage visiting route for job j was represented by solid arcs between $O_{ji}, i \in S$, and processing sequences of machines at stage i were represented by dashed arcs between $O_{ji}, j \in J$. Abdelmaguid (2020) used two sets of vectors to represent a single solution of MPOS. One set of vectors had n vectors to define the stage visiting route for every job explicitly, and the other set of vectors had a total of $\sum_{i \in S} m_i$ vectors to define the processing sequence in every machine at every stage.

Interested reader is referred to Adak et al. (2020) for a detailed review of the literature on MPOS scheduling problem.

3. Operation Permutation and Its Inefficiencies in Proportionate MPOS

Operation permutation is a permutation of operations O_{ji}, as its name implies. A sample operation permutation is as the following for the sample 3-stage proportionate MPOS problem given in Table 1.

Table 1. Sample proportionate MPOS problem

$n = 4$			
	Stage 1	Stage 2	Stage 3
p_i	5	2	2
m_i	2	1	1

$$O_{31}O_{21}O_{32}O_{12}O_{41}O_{23}O_{22}O_{42}O_{33}O_{13}O_{11}O_{43} \quad (1)$$

The permutation in (1) is decoded by reading it from left to right and assigning the respective operation to the first available machine at the respective stage. Accordingly, the schedule represented by (1) is given in Figure 1.

Several problems due to using operation permutation representation in proportionate MPOS are stated through the schedule encoded by the sample permutation. However, these problems do not apply only to the current sample permutation, but they rather represent the general nature of the schedules resulting from using the operation permutation scheme for the proportionate MPOS.

The schedule in Figure 1 has idle times in machines which causes a great increase in the makespan. Indeed, idle machine times are part of most optimum schedules. However, the idle times in the schedule in Figure 1 are undesirable (unnecessary) since there are jobs which can be started earlier and finished earlier in a machine without delaying start time of any job. This type of schedules is called inactive. However, an optimum schedule is known to be an active schedule.

The idle times are resulting from the job routes and machine sequences imposed by the permutation. The permutation encodes the following job routes and machine sequences:

Job routes to visit the stages: Job 1: [2-3-1]; Job 2: [1-3-2]; Job 3: [1-2-3]; Job 4: [1-2-3].

Job sequences in machines: Stage 1, machine 1: [3-4-1]; Stage 1, machine 2: [2]; Stage 2, machine 1: [3-1-2-4]; Stage 3, machine 1: [2-3-1-4]. Note that jobs can be interchanged in machines of stage 1, as long as the job order is preserved.

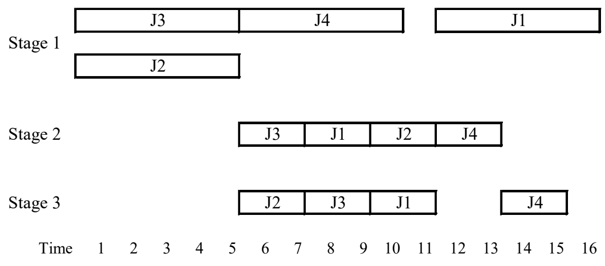

Figure 1. Schedule encoded by the sample operation permutation

The problem with the resulting schedule can be addressed by one of two approaches. First, it can be converted to an active schedule by moving the jobs to earlier times on machines, though not delaying any job's start time of processing on any machine. Applying this post-processing on the schedule, the enhanced schedule in Figure 2 is obtained. A 6 time-unit decrease in the makespan is achieved and an optimal schedule is received since stage 1 is working with full capacity until the completion time of the schedule. However, this post-processing causes inefficiencies in using the representation for the problem.

One inefficiency is due to the computational time required to convert an inactive schedule to an active one. Considering the thousands of schedules visited by search algorithms, performing such an additional time-consuming activity for every schedule generated is not favorable. Another inefficiency of a possible post-processing is that the ultimate schedule reached would be different from what the permutation encodes. This inconsistency between the encoded information and the decoded result poses a challenge in particularly memory-based algorithms. Those algorithms keep good solution characteristics -defined over solution components represented by the permutation- in their memory and recall them later to build good quality solutions. Almost all metaheuristics use some kind of memory as part of their solution methodology.

Another approach to get rid of the unnecessary idle times on machines is to schedule a job as early as possible during decoding even it is in a later position in the permutation. This results in a non-delay schedule. However, the approach was shown (Naderi et al., 2011) not to increase the solution quality, even decrease it.

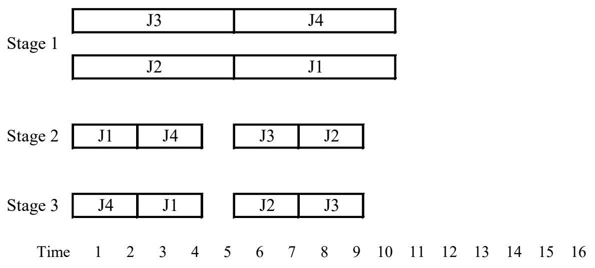

Figure 2. Enhanced schedule by post-processing

A second important problem with using operation permutation to represent proportionate MPOS schedules is the job restrictions present in the permutation. That is, the permutation forces a certain job to be scheduled for the respective stage. This reduces the flexibility present in the shop model and may lead to schedules with increased makespan values. To make the statement clearer, consider Figure 3 which contains 6 different schedules with the same makespan value of 10 for the sample problem. Even more schedules can be generated similarly. A stage processes any job in the same amount of time, and scheduling different jobs make no difference in the makespan as long as the schedule template is the same. This notion of templates for schedules are discussed further in the upcoming paragraphs.

Lastly, one implication drawn out by analyzing proportionate MPOS schedules is presented next, which constituted an important element in constructing the proposed representation.

Consider the dense schedule given in Figure 4 for a 2-stage 33-job problem. There are 23 machines at stage 1 and 10 machines at stage 2. Processing times are 10, and 4 time-units for stages 1 and 2, respectively. A dense schedule leaves no machine idle if there is any job waiting to be processed. Allocating 23 jobs to stage 1 at the beginning of the schedule makes machines of stage 2 wait idle until time 10 and to process 23 jobs after that time, which causes a makespan of 22. Instead, if the schedule in Figure 4 is enhanced as in Figure 5, stage 2 would be able to process another 10-job until time 10 and this will decrease the makespan by 2 time-units, leading to optimal makespan of 20.

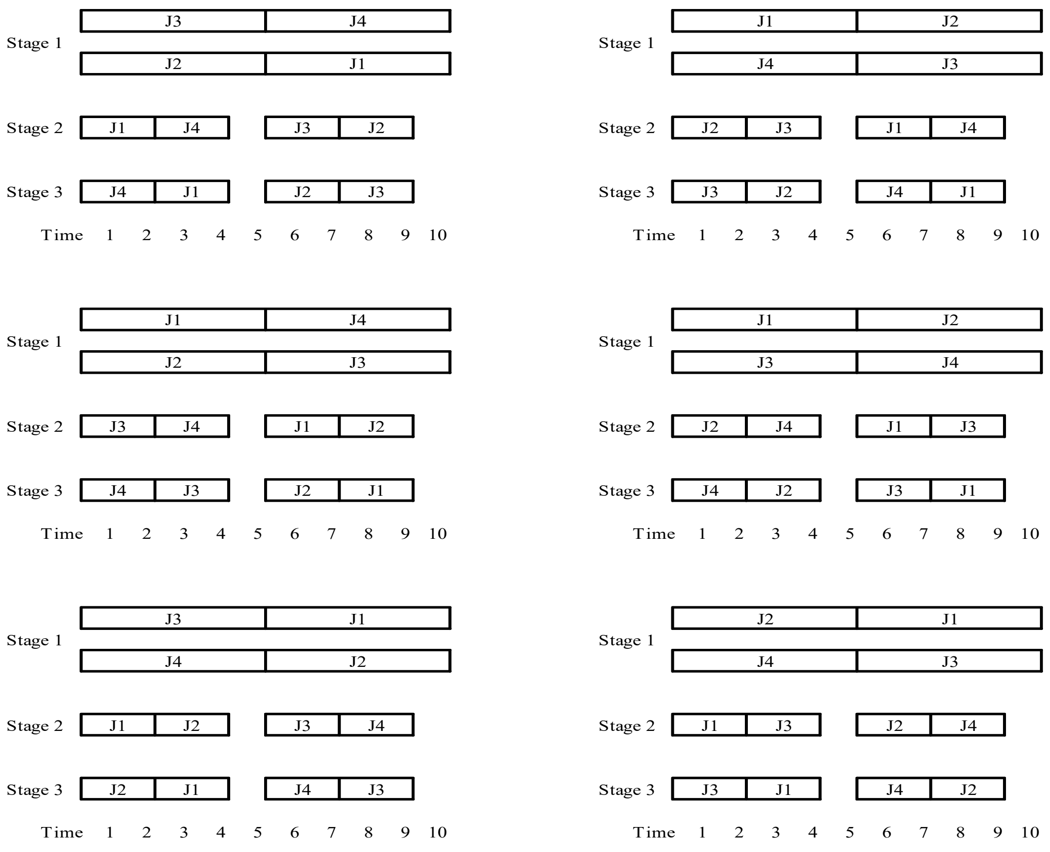

Figure 3. Different schedules with same makespan

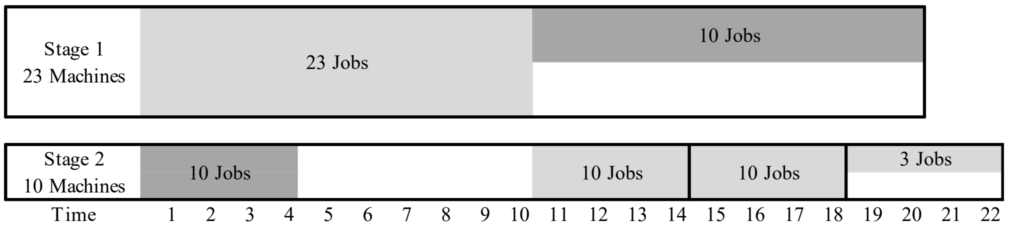

Figure 4. Dense schedule

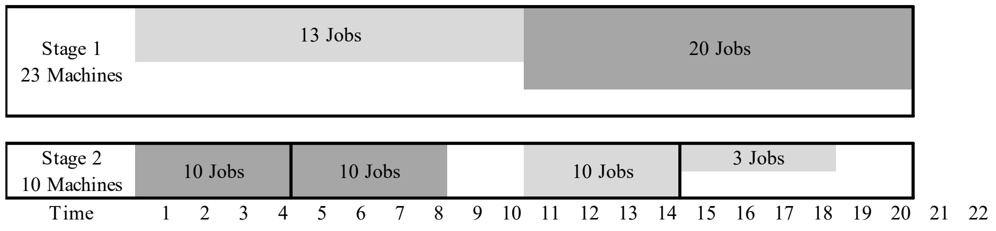

Figure 5. Enhanced schedule

An important implication to draw out from this analysis in schedules is the following. What makes a schedule better than another in terms of makespan objective is the number of jobs allocated to a stage in a block, where block is a p_i-time period at a stage. Thus, to differentiate between good and bad schedule characteristics one should hold this information in the solution representation. That would allow to correctly keep good solution characteristics in the memory of a search algorithm. Again, operation permutation fails to encode this information while representing a proportionate MPOS solution.

3.1. Schedule templates

The schematic schedule representations in Figures 4 and 5 are constructed as schedule templates which do not have explicit calls to job identities. Instead, there exists blocks of job bundles. Numerous different schedules can be generated using a schedule template by assigning different jobs to blocks. This template consideration for schedules is again a novel approach, and it constitutes an important base for the proposed solution representation. The schedule template approach is further discussed in the discussion section of the paper.

4. Novel solution representation: Implicit-stage permutation

To eliminate the inefficiencies caused by using operation permutation to represent a solution of a proportionate MPOS, a novel solution representation scheme is proposed here. This new representation is also able to encode good solution characteristics more effectively and allow for better memory models.

The proposed solution representation for the proportionate MPOS is based on a *stage permutation*. A stage permutation is introduced as a permutation of s stages where each stage repeats n times. Remind that it is assumed in this study that every job is processed at every stage. However, it is straightforward to modify the representation to allow for some jobs not to be processed at some stages. Simply, these stages would repeat less than n times in the permutation.

Following is a sample stage permutation for the sample problem in Table 1.

$$3\ 1\ 3\ 2\ 2\ 1\ 1\ 3\ 2\ 1\ 2\ 3 \qquad (2)$$

The stage permutation is decoded from left to right and *a job* is assigned to the respective stage. The job is selected from an eligible job set consisting of the jobs still waiting to be processed at that stage and are not currently processed at another stage. If there is more than one job in the eligible set, then the job with the lowest desirability across stages is selected. This way, it is aimed to allow for non-empty future eligible job sets as much as possible. If job desirability values are equal among the eligible set, then the jobs are assigned in numerical order. The selected job is assigned for the respective stage to earliest available machine.

The introduced stage permutation scheme for proportionate MPOS solves the flexibility issues mentioned before about the operation permutation as it includes no explicit job calls. This also enables active schedules, importance of which is also emphasized earlier in the paper. However, stage permutation does not supply the information about the number of jobs to be allocated to a stage block, as it is shown to be a distinguishing characteristic of a schedule. Further, it includes repetition of elements, which makes it hard for memory models to extract patterns of good characteristics. Thus, the stage permutation scheme introduced here is proposed to be represented in a higher-level form where a number in permutation encodes the cumulative number of job assignments for the respective stage.

The stage permutation in (2) is converted to the representation in (3) using the encoding given in Table 2. This resulting form of solution representation is named as *implicit stage permutation*. It is implicit because a number in the permutation implicitly refers to two things: 1) the stage where a job is to be assigned, and 2) cumulative number of jobs assigned for that stage.

$$9 - 1 - 10 - 5 - 6 - 2 - 3 - 11 - 7 - 4 - 8 - 12 \qquad (3)$$

The permutation is decoded as follows: 9: assign a *first* job to stage 3, 1: assign a *first* job to stage 1, 10: assign a *second* job to stage 3, 5: assign a *first* job to stage 2, and so on. The schedule given by the permutation is shown in Figure 6.

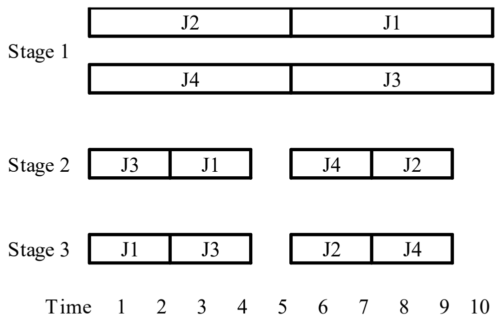

Figure 6. Schedule of the sample implicit stage permutation

Random solution generation

To have random implicit stage permutations, it would not be feasible to generate random numbers from 1 to $n \times s$. Because there is an ordering between cumulative number of assignments to a stage. That is, representation for a second assignment, for instance, should not precede the representation for the first assignment. Otherwise, the implicit stage permutation would not serve its purpose and become meaningless.

To generate random solutions, first a random stage permutation should be created and then it should be converted to an implicit stage permutation using Table 2. Creating a random stage permutation is straightforward as it is a permutation from 1 to s with every element repeating n times.

Table 2. Encoding to construct implicit stage permutation from a stage permutation

	1st assignment	2nd assignment	…	nth assignment
Stage 1	1	2	…	n
Stage 2	$n+1$	$n+2$	…	$2n$
Stage 3	$2n+1$	$2n+2$	…	$3n$
⋮	⋮	⋮	…	⋮
Stage i	$n(i-1)+1$	$n(i-1)+2$	…	$n \times i$
⋮	⋮	⋮	…	⋮
Stage s	$n(s-1)+1$	$n(s-1)+2$	…	$n \times s$

5. Computational Tests

The implicit stage permutation proposed in this study is aimed to be a solution representation for proportionate MPOS, particularly within a solution algorithm for the problem. For an effective ant colony optimization algorithm using the proposed solution representation see (Adak, 2020).

Developing a solution algorithm for the problem is out of the scope of this study. Rather, the favorable results due to the proposed solution representation are shown by random schedules. Random solutions are simple, quick but mostly inferior quality solutions for combinatorial optimization problems. It is highly unlikely to pick a good quality solution randomly from a very huge solution space. However, random solutions are used to initiate a search process in many algorithms. Starting with higher quality initial solutions may lead to increased performance or at least it decreases the computational time required to reach the best objective value.

In computational tests, quality of random solutions by the proposed solution representation are compared with the ones by the operation permutation representation. Table 3 reports the results of the computational tests. 12 test instances from a benchmark testbed (Matta, 2009) are used in the tests. 10 random solutions are generated for each of the instances, and minimum, average, and maximum of the 10 solutions are given in the table. Optimum makespan values for the instances were reported in literature (Abdelmaguid, 2020) and are presented in the table. The table also gives the percent improvement in average makespan deviation from the optimum makespan, achieved by the proposed representation.

The results given in Table 3 suggests the ability of the proposed representation in generating higher quality solutions even in a random search. This is due to two features of the representation: 1) it does not enforce job identities in the permutation, thus does not restrict the flexibility already present in a proportionate shop model, and 2) it creates active schedules and avoids unnecessary idle times in the schedule.

Table 3. Comparison of quality of random solutions by operation permutation and by implicit stage permutation

					Operation-permutation (10 random permutations)			Implicit-stage permutation (10 random permutations)			
	Number of stages	Number of jobs	Max. number of machines in a stage	Optimum Makespan	Min. Makespan	Avr. Makespan	Max. Makespan	Min. Makespan	Avr. Makespan	Max. Makespan	% Improvement in deviation (Avr. makespan)
S4-P3	4	63	22	48	75	81	90	**51**	**54.3**	**57**	80.91
S4-P4	4	38	14	27	42	45.6	51	**36**	**36.3**	**39**	50.00
S4-P6	4	60	20	25	37	41.2	46	**26**	**27.8**	**28**	82.72
S4-P7	4	53	17	36	48	53.5	63	**36**	**36.5**	**38**	97.14
S8-P14	8	72	14	84	136	146	153	**86**	**90.2**	**96**	90.00
S8-P17	8	100	24	60	97	102.7	108	**60**	**62.1**	**64**	95.08
S8-P18	8	106	22	56	96	99.4	104	**58**	**61.2**	**66**	88.02
S8-P20	8	105	22	49	77	82	85	**51**	**52.8**	**58**	88.48
S16-P9	16	101	10	121	221	228.3	239	**126**	**133.9**	**143**	87.98
S16-P13	16	112	10	180	336	352.8	372	**195**	**207.6**	**213**	84.03
S16-P14	16	97	10	84	156	162.3	176	**90**	**94.7**	**101**	86.33
S16-P18	16	102	10	110	216	226.2	240	**122**	**125.6**	**130**	86.57

6. Discussion

The proposed representation for proportionate MPOS is an easy-to-use approach with straightforward encoding and decoding procedures. Complex representations of feasible solutions in attempting to solve a combinatorial problem makes it even harder and may render useless most of the time. Further, it may require additional computational time to decode a complicated representation and calculate the objective value, which is disadvantageous since search algorithms require to go over many solutions and assess the quality of every single solution. The simplicity of the proposed solution representation is very favorable in that aspect.

Instead of creating exact schedules where positions of jobs are explicitly stated, the proposed representation works through schedule templates. It states how many jobs would be allocated to distinct time slots, creates groups of jobs, and determines position of the groups at every stage in the final schedule. Different schedules having same template can be generated by using different rules of job selection from the eligible set during decoding the permutation. The template generating approach proposed in this study is particularly useful as it deals with solution families with same makespan value instead of single solutions. This decreases the problem size and facilitates searching the solution space.

It should be emphasized that the representation in this paper is proposed specifically for the proportionate MPOS with makespan criterion. The proposed representation encodes cumulative number of job assignments to a stage, since this information has a decisive role in makespan value.

However, a different objective function would require different considerations about the properties a solution representation should encode. Again, because of the proportionality of the stages the representation does not encode job identities. But if a non-proportionate shop is considered, then a solution representation should also encode job identities.

Since the proportionality definition is mostly jobwise in the literature for other shop models, the proposed solution representation is not directly applicable in that problems. However, the proposed approach can be reconsidered for other proportionate models and can be adapted accordingly. The general idea in proposed solution representation is valuable in dealing with solutions of scheduling problems. It is based on constructing a representation that holds significant information about the machine environment, job characteristics and objective function. However, in scheduling literature, the common approach in selecting a scheme to represent solutions is based on generating complete, valid, and feasible solutions. The way of handling solutions as proposed in this study leads to a problem-specific approach for representation and results in more effective solution algorithms for scheduling problems.

7. Conclusion

A novel solution representation is proposed in this study for the proportionate multiprocessor open shop scheduling problem with makespan criterion. The representation, named implicit stage permutation, encodes the cumulative number of job assignments to a stage. It is a problem-specific scheme that focuses on the shop characteristics and the objective function. This feature of the proposed scheme makes it a powerful approach in representing solutions of the problem.

The performance of the proposed representation was compared with the conventional operation permutation representation for multiprocessor open shop scheduling problems by generating random solutions for benchmark instances from the literature. The proposed representation produced random solutions with a 11% deviation from the optimum in average while it was 77% in operation permutation, leading to an 84%-improvement in solutions. Its performance in random solutions is an indicator of its efficiency as a solution representation scheme for proportionate multiprocessor open shop problem.

An important conclusion to draw out from this study is that it suggests a new way of dealing with solutions in scheduling problems. More effective representation schemes can be constructed by considering the machine environment, job characteristics and the objective function of the scheduling problem at hand. Future studies should consider this approach in designing solution algorithms for various scheduling problems.

In future research, the proposed representation can be used to improve previous results on proportionate multiprocessor open shop problem with makespan criterion. Further, it can be adapted to other scheduling problems where machine-wise proportionality is considered.

References

Abdelmaguid, T. F. 2020. Scatter search with path relinking for multiprocessor open shop scheduling. *Computers & Industrial Engineering,* 141**,** 106292.

Abdelmaguid, T. F., Shalaby, M. A. & Awwad, M. A. 2014. A tabu search approach for proportionate multiprocessor open shop scheduling. *Computational Optimization and Applications,* 58**,** 187-203.

Adak, Z. 2020. *An ant colony optimization approach for the proportionate multiprocessor open shop.* PhD Doctoral thesis, Marmara University.

Adak, Z., Arıoğlu Akan, M. Ö. & Bulkan, S. 2020. Multiprocessor open shop problem: literature review and future directions. *Journal of Combinatorial Optimization,* 40**,** 547-569.

Azadeh, A., Hosseinabadi Farahani, M., Torabzadeh, S. & Baghersad, M. 2014. Scheduling prioritized patients in emergency department laboratories. *Computer Methods and Programs in Biomedicine,* 117**,** 61-70.

Bai, D., Zhang, Z.-H. & Zhang, Q. 2016. Flexible open shop scheduling problem to minimize makespan. *Computers & Operations Research,* 67**,** 207-215.

Chen, B., Potts, C. N. & Woeginger, G. J. 1998. A Review of Machine Scheduling: Complexity, Algorithms and Approximability. *In:* DU, D.-Z. & PARDALOS, P. M. (eds.) *Handbook of Combinatorial Optimization: Volume1–3.* Boston, MA: Springer US.

Goldansaz, S. M., Jolai, F. & Zahedi Anaraki, A. H. 2013. A hybrid imperialist competitive algorithm for minimizing makespan in a multi-processor open shop. *Applied Mathematical Modelling,* 37**,** 9603-9616.

Gonzalez, T. & Sahni, S. 1976. Open Shop Scheduling to Minimize Finish Time. *J. ACM,* 23**,** 665–679.

Liaw, C.-F. 2000. A hybrid genetic algorithm for the open shop scheduling problem. *European Journal of Operational Research,* 124**,** 28-42.

Mao, W. Multi-operation multi-machine scheduling. HPCN-Europe 1995, 1995 Berlin, Heidelberg. Springer, Berlin, Heidelberg, 33-38.

Matta, M. E. 2009. A genetic algorithm for the proportionate multiprocessor open shop. *Computers & Operations Research,* 36**,** 2601-2618.

Naderi, B., Fatemi Ghomi, S. M. T., Aminnayeri, M. & Zandieh, M. 2011. Scheduling open shops with parallel machines to minimize total completion time. *Journal of Computational and Applied Mathematics,* 235**,** 1275-1287.

Ow, P. S. 1985. Focused Scheduling in Proportionate Flowshops. *Management Science,* 31**,** 852-869.

Pinedo, M. L. 2016. *Scheduling: Theory, Algorithms, and Systems,* Cham, Springer International Publishing.

Zhang, J., Wang, L. & Xing, L. 2019. Large-scale medical examination scheduling technology based on intelligent optimization. *Journal of Combinatorial Optimization,* 37**,** 385-404.