Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

A DISCRETE-EVENT HEURISTIC FOR MAKESPAN OPTIMIZATION IN MULTI-SERVER FLOW-SHOP PROBLEMS WITH MACHINE RE-ENTERING

Angel A. Juan
Pedro Copado
Javier Panadero

IN3 – Computer Science Dept.
Universitat Oberta de Catalunya
Euncet Business School
Castelldefels, 08860, SPAIN

Christoph Laroque

Institute for Management and Information
University of Applied Sciences Zwickau
Kornmarkt 1
Zwickau, 08012, GERMANY

Rocio de la Torre

INARBE Institute
Public University of Navarre
Campus de Arrosadia
Pamplona, 31006, SPAIN

ABSTRACT

Modern Manufacturing, known as Industrial Internet or Industry 4.0, is more than ever determined by customer-specific products, that are to be manufactured and delivered in given lead times and due-dates. Many of these manufacturing systems can be modeled as flow-shops where some of the processes can handle jobs on parallel machines. In addition, complex manufacturing environments contain specific machine loops or re-entry cycles where jobs might re-enter specific processes at some point of the flow-shop chain. A specific server is assigned to a job the first time it visits a machine, and it is quite usual that this job has to be processed by exactly the same server if it re-visits the machine due to quality issues. With the goal of minimizing the makespan, this paper analyzes this complex flow-shop setting and proposes an original discrete-event heuristic for solving it in short computing times. Our algorithm combines biased (non-uniform) randomization strategies with the use of a discrete-event list, which iteratively processes as the simulation clock advances. A series of computational experiments contribute to illustrate the performance of our methodology.

1 INTRODUCTION

Manufacturing in the era of cyber-physical systems and Industry 4.0 is determined by a growing demand for customer-specific products, that are to be manufactured and delivered in given lead times and in-time. For the operative production scheduling process, questions like “When to release a specific lot at latest to deliver in-time?” are answered by experience or though the use of backward-oriented scheduling. In practice, uncertainty is generally not taken into account. In a modern industrial setting with mixed-model manufacturing lines with a high degree of automation, things get even more complex. Especially, in semiconductor manufacturing, the production system itself has a high complexity (re-entry cycles, several hundreds of process steps, etc.). A more sophisticated answer could be delivered by the use of simulation-optimization approaches. During planning, construction, and ramp-up, this methods are well established, but they are rarely adopted for operational support during the decision-making process.

In real-life applications of scheduling, it is possible that some jobs have to visit certain machines more than once. This repetitiveness may generate conflicts among jobs, at some machines and at different levels during the whole process. Moreover, later operations may also interfere with earlier required operations in the same machine on other jobs (Chen et al. 2008). When each machine processes a certain set of jobs and some of these need to be processed more than once, the classical permutation flow-shop scheduling problem is transformed into the so-called *re-entrant* permutation flow-shop scheduling problem or RPFSP (Graves et al. 1983). This paper analyzes an enriched version of the RPFSP in which some machines might also be able to process several jobs in parallel, i.e., they are multi-server machines. Despite being inspired in a real-life case from a German manufacturing industrial partner, this rich variant of the RPFSP with multi-server machines has not been frequently addressed in the existing literature.

Accordingly, the main contributions of this work can be stated as follows: *(i)* the modeling of a multi-server flow-shop problem with machine re-entrance (loops); *(ii)* an original biased-randomized optimization heuristic, which also integrates concepts from discrete-event simulation to deal with the complexity of the system in a natural way; *(iii)* a review of related work in the existing scientific literature; and *(iv)* a series of computational experiments that illustrate the application of the proposed methodology for solving such complex flow-shop scenarios. Biased-randomized optimization algorithms are extensively discussed in Grasas et al. (2017). In a nutshell, they make use of Monte Carlo simulation and skewed probability distributions to introduce a non-uniform random behavior into a constructive heuristic. Thus, the heuristic is transformed into a more powerful probabilistic algorithm, which can be run in virtually the same wall-clock time as the original heuristic if parallelization techniques are employed. Some applications of these algorithms include distributed-assembly flow-shop problems (Gonzalez-Neira et al. 2017), integrated routing and facility-location problems (Quintero-Araujo et al. 2019), arc routing problems (Gonzalez-Martin et al. 2012), and e-marketing problems (Marmol et al. 2020). Similarly, discrete-event heuristics – i.e., the combination of heuristics with concepts from discrete-event simulation – have been employed in Fikar et al. (2016) to deal with optimization problems with synchronization issues. To the best of our knowledge, however, this is the first time that both techniques are hybridized to solve a complex flow-shop problem as the one analyzed here.

The remainder of the paper is organized as follows. Section 2 provides a more detailed description of the scheduling problem studied in this paper. Section 3 reviews related work on similar flow-shop problems. Section 4 proposes a novel optimization heuristic that incorporates concepts from discrete-event simulation. Section 5 carries out a series of numerical experiments to test our methodology. Section 6 analyzes the obtained results. Section 7 discusses some preliminary managerial insights derived from our work. Finally, Section 8 summarizes the main findings of this paper and points out some future research lines.

2 A DETAILED DESCRIPTION OF THE PROBLEM

Figure 1 displays a simple but illustrative example of the type of flow-shop systems we are considering in this work. As it can be seen at the top part of the figure, jobs arrive at the system on the left side, are processed in machine $M1$, and then go to machine $M2$, where they are processed either by server $M2$-1 or by server $M2$-2, which are in parallel. After being processed by one of these servers, jobs reach machine $M3$, from which they come back to the previously visited server in machine $M2$ before returning to machine $M3$ – notice that once a job i is processed in a parallel server k of a machine j, any time i re-enters the same machine j it has to be processed by the same server k as before. After that, jobs leave the production system – since they have been completed already. Due to the existence of this *re-entrance loop*, each job has to be processed twice in one of the parallel servers of machine $M2$ as well as in machine $M3$. Actually, the fact that jobs are processed in parallel machines makes it possible for some jobs to overtake others during the parallel processing, i.e., the order in which jobs leave the system need not correspond to the order in which jobs entered it.

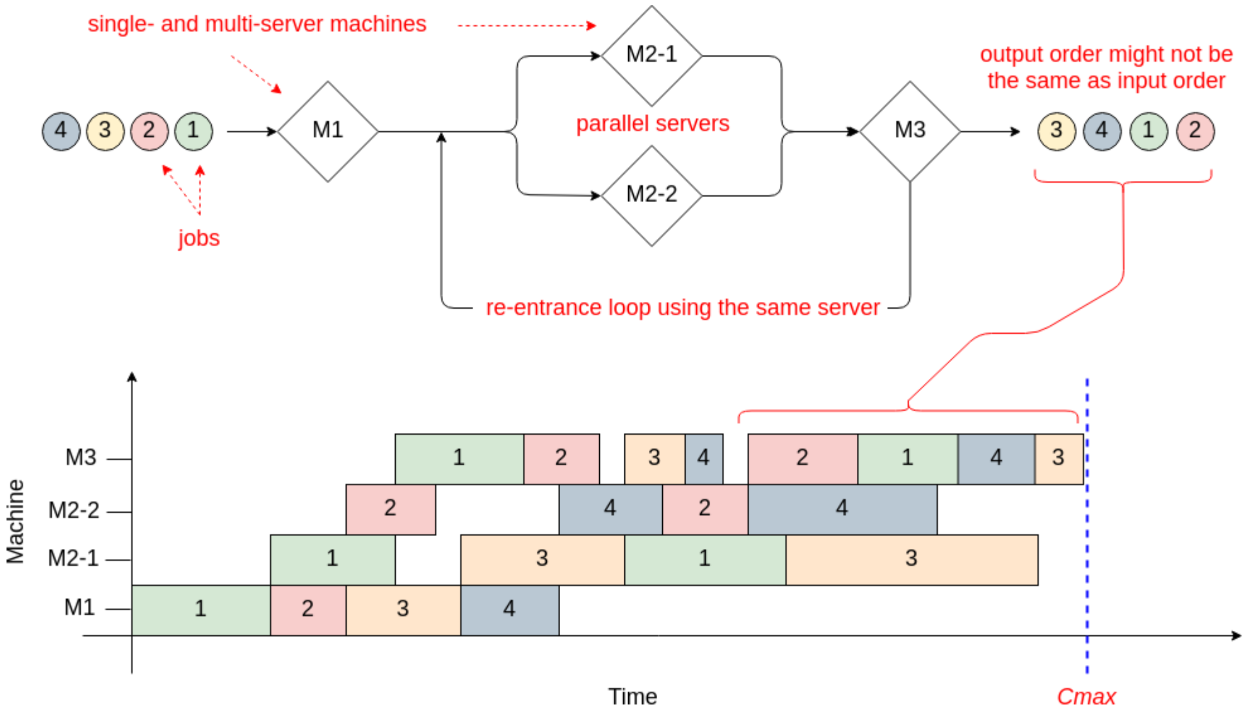
single- and multi-server machines
jobs
M1
M2-1
M2-2
parallel servers
M3
output order might not be the same as input order
re-entrance loop using the same server
Machine
Time
Cmax

Figure 1: A simple example illustrating the multi-server flow-shop problem with machine re-entering.

The bottom part of the figure contains a typical time-versus-machine representation of a flow-shop problem. Thus, for instance, job 1 starts in machine 1 at time 0. Then, it moves to the first available server in machine 2 and, after that, it is processed by machine 3. However, before re-entering its previously visited server in machine 2, it has to wait until job 3 is processed there. Once processed twice by the first server in machine 2, job 1 returns to machine 3 for a second service, but it has to wait until machine 3 is available since it is currently occupied by job 2, which overtook job 1 in the parallel section of the flow-shop due to using lower processing times in previous machines.

3 RELATED WORK

During the last decades, a wide range of approaches have been developed for addressing the RPFSP, from exact methods to heuristic and meta-heuristic algorithms (Danping and Lee 2011). Regarding exact methods, they have been typically formulated as integer programming problems. However, being a *NP-hard* problem, the use of exact methods is limited to small- and medium-sized instances (Kaihara et al. 2010; Lin et al. 2013). Due to the complexity of the RPFSP, some studies have focused on improving the computational times of an optimization algorithm, while others have paid attention to the development of an efficient heuristic (Pan and Chen 2003; Kim and Lee 2009; Yu and Pinedo 2020). For example, Gao et al. (2015) present a three-step methodology based on the enhancement of an initial solution. The main goals considered in these studies are: *(i)* to minimize the weighted sum of key device shortages; *(ii)* to maximize the weighted sum of processed lots; *(iii)* to minimize the number of required machines; and *(iv)* to minimize the makespan.

Regarding heuristic approaches, the more popular ones have been the constructive heuristics, as well as those based on dispatching rules. For example, Choi and Kim (2007) designed and implemented a local search algorithm based on simulated annealing with the main objective of minimizing the makespan.

There is a wide variety of heuristics applied to small-sized problems, most of them focused on minimizing the makespan (Jing et al. 2008; Jing et al. 2011). Similarly, Wu et al. (2018) developed four heuristics, based on commonly used local search methods, in order to construct an approximate approach for solving the RPFSP. In addition, the solutions generated by these heuristics are also employed as starting solutions inside a simulated annealing metaheuristics.

Actually, metaheuristics have been frequently employed to solve the RPFSP. Among them, genetic algorithms (GA) and tabu search algorithms, as in Chen et al. (2008), where all jobs are assumed to follow the same path – which includes re-entrance in some machines. In a similar context, Mousavi et al. (2018) extend the previous paper by including the total tardiness along with the makespan, thus solving a bi-objective optimization problem. Amin Naseri et al. (2015) designed a mathematical model for solving the RPFS with no-waiting re-entrant operations. Three different methods were presented: *(i)* a simulated annealing; *(ii)* a GA; and *(iii)* a bottleneck-based heuristic. Comparing the solutions obtained by the aforementioned algorithms (considering both small- and large-sized instances), the authors concluded that their simulated annealing algorithm offered a better performance.

Regarding existing work on systems with parallel machines, Kim and Lee (2009) suggested and tested two heuristics for solving the re-entrant flow-shop problem with unrelated parallel machines. The main objective is to minimize the makespan while considering the total tardiness as a key negotiation tool with the customer. Likewise, Jia et al. (2013) presented a real-time closed-loop control dispatching heuristic for parallel machines with incompatible job families, limited waiting time constraints, re-entrant flow and dynamic arrivals. The scheduling is carried out by batches. If a batch family is performing a job on one machine, it cannot be interrupted until the entire batch ends. Moreover, the re-entrant order is subjected to the waiting time: if the waiting time exceeds a certain threshold, the batch must go through the previous machine again (thus repeating the sequence). In that case, the dispatching heuristic method is transformed into a GA to minimize the total weighted tardiness. Eskandari and Hosseinzadeh (2014) deal with a hybrid flow-shop scheduling problem with re-work. For solving this problem, some heuristic methods based on dispatching rules and a variable neighborhood search are proposed. Although it is considered to include re-entrant flow, the jobs returned to the machines are only those with defects. Moreover, among the assumptions of the problem is that the machines are always available.

4 A DISCRETE-EVENT HEURISTIC

Figure 2 shows a conceptual schema of the solving approach. It relies on a multi-start framework that iteratively calls a biased-randomized and adapted version of the well-known NEH heuristic for the classical permutation flow-shop scheduling problem (Nawaz et al. 1983). Biased-randomization (BR) techniques make use of skewed probability distributions to introduce non-uniform randomization into a constructive heuristic such as the NEH, thus transforming it into a probabilistic algorithm able to generate many 'good-quality' solutions to the problem in short computational times. Among other applications, these techniques have been recently applied in solving different flow-shop scheduling problems (Ferrer et al. 2016), vehicle routing problems (Belloso et al. 2019; Dominguez et al. 2016), capacitated location routing problems (Quintero-Araujo et al. 2017), and even to extend and enhance traditional metaheuristic frameworks (Ferone et al. 2019).

The NEH heuristic, however, is designed to minimize the makespan in a flow-shop scenario with single-server machines and without any re-entering point. Hence, one of the main challenges in this work was to adapt the way this heuristic computes makespan to a new multi-sever flow-shop scenario with re-entering machines. Notice that this computation is not straightforward due to the fact that some jobs must have to wait for the proper server to be available. This issue is even more troublesome if we force the job to repeat the previously visited server each time it re-enters a given multi-server machine. Hence, in order to compute the makespan associated with a given solution (permutation of jobs) generated by the constructive BR heuristic, we employ a discrete-event deterministic simulation: two types of events are considered, the starting of a job i in server k of a machine j at time t, denoted as $(t,i,0,j,k)$, and the ending

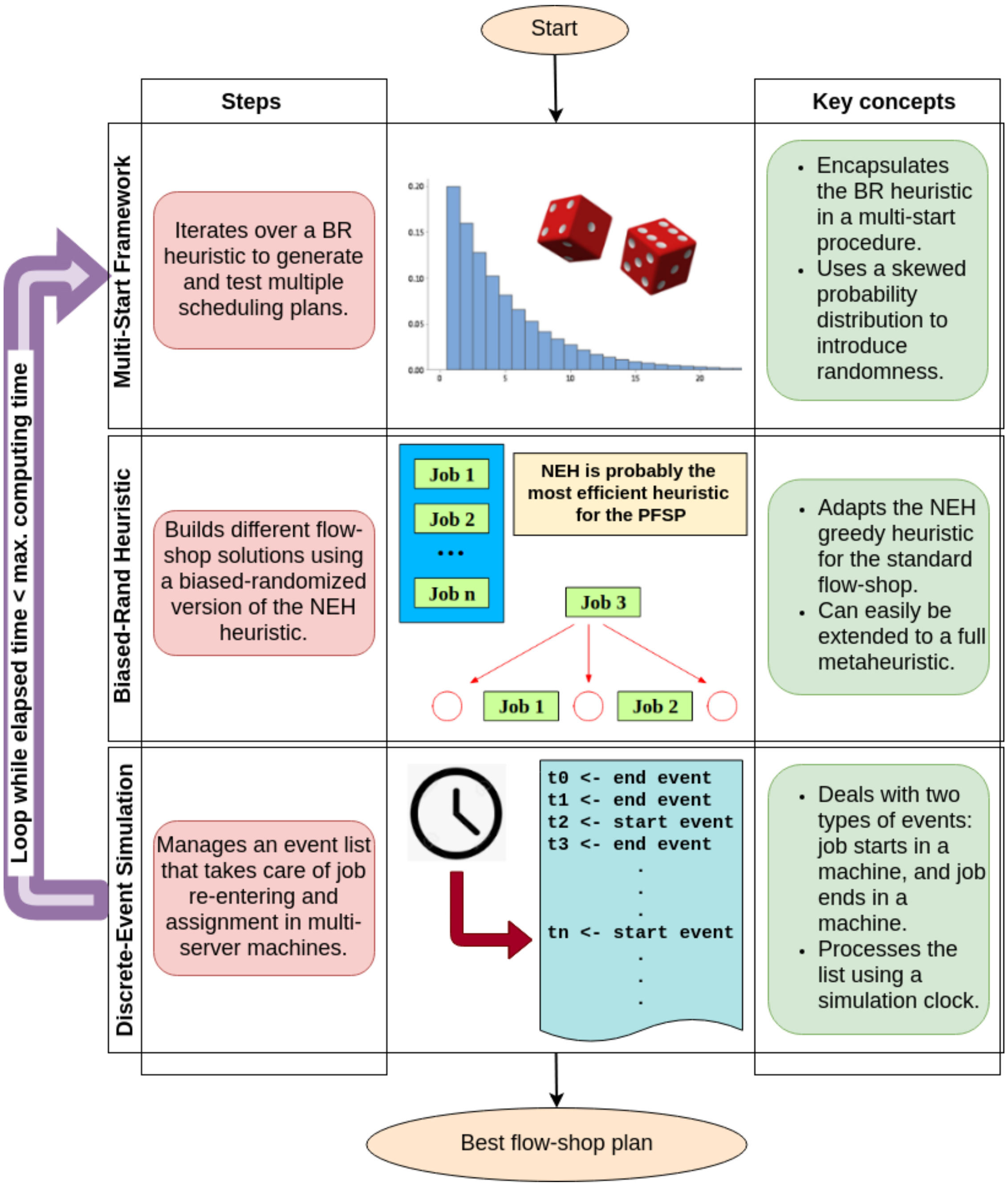
Start
Steps
Key concepts
Multi-Start Framework
Iterates over a BR heuristic to generate and test multiple scheduling plans.
0.20
0.15
0.10
0.05
0.00
0
5
10
15
20
• Encapsulates the BR heuristic in a multi-start procedure.
• Uses a skewed probability distribution to introduce randomness.
Biased-Rand Heuristic
Builds different flow-shop solutions using a biased-randomized version of the NEH heuristic.
Job 1
Job 2
• • •
Job n
NEH is probably the most efficient heuristic for the PFSP
Job 3
Job 1
Job 2
• Adapts the NEH greedy heuristic for the standard flow-shop.
• Can easily be extended to a full metaheuristic.
Discrete-Event Simulation
Manages an event list that takes care of job re-entering and assignment in multi-server machines.
t0 <- end event
t1 <- end event
t2 <- start event
t3 <- end event
tn <- start event
• Deals with two types of events: job starts in a machine, and job ends in a machine.
• Processes the list using a simulation clock.
Loop while elapsed time < max. computing time
Best flow-shop plan

Figure 2: Conceptual schema of the solving approach.

of a job i in server k of a machine j at time t, denoted as $(t,i,1,j,k)$. The discrete-event list is initialized by considering the ending events generated by the list of jobs $(1,2,\ldots,n)$ that enter in machine $M1$. Then, the simulation clock is started and the next event in the list is extracted from it and processed. On the one hand, ending events might free the corresponding server / machine and also schedule new starting events for the associated jobs in case they have not completed the whole sequence of machines yet (including re-entering points). On the other hand, starting events will block the required servers or machines and will also generate new ending events associated with the starting ones. This generation of new events continues until the last job is processed by the last server / machine, which defines the solution makespan.

The entire process describe above is repeated in a multi-start framework as far as the elapsed time does not exceed the maximum computational time allowed by the manager (or any other finishing criterion). Because of the BR effect, each iteration is likely to provide a different solution. Once this maximum time is reached, the best-found solution (the one with the lowest makespan) is returned by the algorithm.

5 COMPUTATIONAL EXPERIMENTS

The proposed BR algorithm has been implemented using Python 3.7 and tested on a workstation with a multi-core processor Intel Xeon E5-2650 v4 with 32GB of RAM. To the best of the authors knowledge, there are no public instances for the multi-server flow show-shop problem with machine re-entering. Hence, we have adapted some of the well-known instances proposed by Taillard (1993) for the classical permutation flow-shop problem, which are available from Taillard (2020). The extension incorporates both multi-server machines as well as machine re-entering. In particular, in an instance with m machines, we have modified machine $m-2$ so that it is a 2-server machine (i.e., it can process two jobs in parallel). Also, we have added a re-entering point (loop) just after machine $m-1$, so the first time any job finishes being processed by machine $m-1$ it has to re-visit the same server as before in machine $m-2$.

Table 1 displays the obtained results as follows: the first column shows the original name of the instance, which also indicates the number of jobs and machines (e.g., instance m_i_j has i jobs and j machines). The next block of columns belong to the classical version of the problem, with single-server machines and without re-entrance. This scenario is considered to show that the solutions provided by the methodology (OBS) are relatively close to the best-known solutions from the literature (BKS), which allows to validate the performance of the described algorithm. Obviously, there is gap between OBS and BKS, since the methodology is not specifically designed to solve the classical version of the problem. Hence, it does not make use of the traditional local search operators or the fast makespan computations – which are invalid for scenarios with multi-sever machines and loops. The remaining three blocks are the most interesting ones, since they consider the above mentioned real-world problem; the results obtained for the three extended scenarios show: *(i)* multi-server without re-entrance loops; *(ii)* single-server with re-entrance machines; and *(iii)* multi-server with re-entrance machines.

6 ANALYSIS OF RESULTS

Figure 3 summarizes the results provided in Table 1. Notice that, despite being designed for the general case with multi-server (parallel) machines and re-entrance points (loops), the discrete-event heuristic performs reasonably well when applied to the classical single-sever flow-shop without loops (series scenario). For the tested instances, the discrete-event heuristic provides an average makespan of $1,392.7$, while the best-known solution in the literature – obtained with state-of-the-art metaheuristics – is $1,336.3$. This result contributes to validate the approach, specially taking into account that metaheuristics for the classical flow-shop problem make use of analytical expressions to compute the makespan, together with local search operators and data structures that allow for a more efficient exploration of the solution space. Unfortunately, these analytical expressions, operators, and data structures cannot be employed in the general scenario with multi-server machines and loops.

Table 1: Results across different scenarios.

Taillard Inst.	Single-server without loop				Multi-server without loop			Single-server with loop				Multi-server with loop			
	BKS [1]	**OBS [2]**	**time (s)**	**gap (%) [1]-[2]**	**OBS [3]**	**time (s)**	**gap (%) [2] - [3]**	**OBS [4]**	**time (s)**	**gap (%) [2] - [4]**	**gap(%) [4] - [5]**	**OBS [5]**	**time (s)**	**gap(%) [2] - [5]**	**gap(%) [3] - [5]**
4_20_5	1293	1358	10.3	5.03	1352	0.3	-0.44	2428	8.6	78.79	-11.07	2186	64.3	37.88	-38.15
5_20_5	1235	1305	19.7	5.67	1250	103.9	-4.40	2108	27.7	61.53	-3.23	2042	104.8	36.09	-38.79
6_20_5	1195	1227	93.5	2.68	1209	58.3	-1.49	2403	9.0	95.84	-4.12	2308	43.1	46.84	-47.62
7_20_5	1234	1251	84.5	1.38	1234	55.1	-1.38	2381	12.2	90.33	0.00	2381	35.7	47.46	-48.17
8_20_5	1206	1226	197.5	1.66	1214	30.9	-0.99	2367	90.4	93.07	-9.63	2159	79.2	43.21	-43.77
9_20_5	1230	1267	105.8	3.01	1254	99.2	-1.04	2394	90.4	88.95	-7.26	2232	15.1	43.23	-43.82
10_20_5	1108	1131	115.7	2.08	1104	155.5	-2.45	2103	5.3	85.94	0.00	2103	0.7	46.22	-47.50
11_20_10	1582	1676	129.5	5.94	1663	127.4	-0.78	2673	3.841	59.49	0.00	2673	6.612	37.30	-37.79
12_20_10	1659	1747	177.5	5.30	1730	144.7	-0.98	2714	13.097	55.35	-11.09	2443	106.084	28.49	-29.19
13_20_10	1496	1561	281.7	4.34	1539	280.5	-1.43	2374	184.777	52.08	-3.17	2301	65.117	32.16	-33.12
14_20_10	1378	1461	238.1	6.02	1422	200.2	-2.74	2264	0	54.96	0.00	2264	38.218	35.47	-37.19
15_20_10	1419	1502	262.0	5.85	1493	37.4	-0.60	2559	18.3	70.30	-22.21	2094	81.611	28.27	-28.70
Avg	**1336.25**	**1392.67**	**142.98**	**4.08**	**1372.00**	**107.8**	**-1.56**	**2397.33**	**38.64**	**73.89**	**-5.98**	**2265.50**	**53.39**	**38.55**	**-39.48**

The boxplot also shows that considering the multi-sever (parallel machines) option always improves the makespan when compared with the associate series scenario, e.g.: while the series-without-loop scenario gives an average makespan of 1,392.7, the substitution of one single-server machine by a two-server machine reduces this value to 1,372.0. Likewise, while the single-server-with-loop scenario provides an average makespan of 2,397.3, its parallel counterpart reduces this value to 2,265.5. Finally, notice that scenarios with a loop always provide a higher makespan than the equivalent scenarios without re-entrance points, which is coherent with the authors expectations, since the jobs are processed multiple-times on the server.

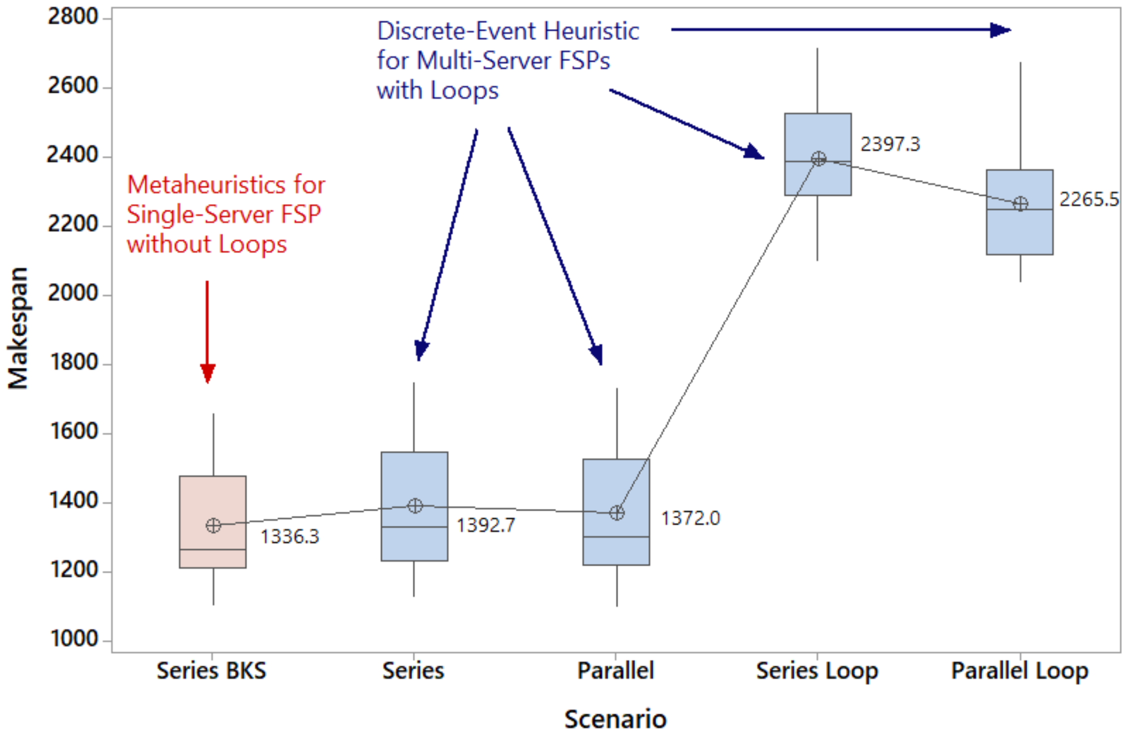

Figure 3: Makespan comparison among different scenarios and solving approaches.

7 MANAGERIAL INSIGHTS

While the flow-shop literature is vast, real-life systems are diverse and they generate rich scheduling problems with special characteristics that, when considered altogether, impose new challenges that have been seldom analyzed in previous works. That is the case, for instance, of flow-shops with multi-server machines and re-entering points. In these complex scenarios, even traditional metaheuristic algorithms might find difficulties to generate reasonably good solutions due to the unpredictable time dependencies generated by the combination of loops and parallel servers. However, discrete-event heuristics – like the one proposed here – can deal with these synchronization issues in a quite natural way. Also, by incorporating simulation concepts inside a heuristic optimization framework, these simulation-optimization algorithms go beyond classical simulation-alone methods, which lack the capacity to efficiently explore the solution space.

All in all, these simulation-based heuristics constitute a new managerial tool that can support decision making in the manufacturing and process industries. Due to the (in comparison to mathematical optimization) lower computation times for the heuristic approach, an real-life application for operational decision support can be regarded as realistic. Additionally, with the use of stochastic input factors, uncertainty known from the real-world setting can be considered in operational decision making. Here, the simulation-based approach might especially prove beneficial.

8 CONCLUSIONS AND FUTURE WORK

This paper has addressed a challenging permutation flow-shop problem with multi-server machines and re-entrance points. Despite its relevance for some of the project's industrial partners, this problem has been rarely analyzed in the existing scheduling literature. The main complexity in solving the problem lies in the computation of the makespan, since the combination of parallel servers and loops introduce subtle time dependencies in the order in which jobs are processed by the machines.

Accordingly, in order to compute the makespan in such a system, a discrete-event simulation is run. This simulation uses two types of events: the starting of a job in a machine, and the end of a job in a machine. The simulation is then encapsulated into a biased-randomized heuristic, which allows us to test different promising configurations of jobs. Computational results show that simulation-optimization scheduling approaches – as the one proposed here – can be beneficial as a decision-support technology. This is specially the case for operational decisions in complex manufacturing settings as the ones that can be found in modern, highly automated, industry 4.0 environments.

As future research lines, the following ones can be highlighted: *(i)* to test the algorithm in a real-life manufacturing environment and quantify how it can improve the results obtained by current industrial practices; *(ii)* to consider optimization objectives other than makespan, e.g., to maximize rewards for partially completed jobs, etc.; *(iii)* to consider backward-oriented approaches, where simulation-based scheduling generates a specific production plan for a given order set with fixed due-dates under the consideration of stochastic behavior; and *(iv)* to extend the algorithm into a full simheuristic one (Juan et al. 2018), so it can also consider stochastic processing times.

ACKNOWLEDGMENTS

This work has been partially funded by the IoF2020 project of the European Union (grant agreement no. 731884), the Spanish Ministry of Science, Innovation, and Universities (RED2018-102642-T), and the Erasmus+ Program (2019-I-ES01-KA103-062602). Likewise, this work has been partially funded by the European project iDEV40. The iDev40 project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 783163. The JU receives support from the European Union's Horizon 2020 research and innovation programme. It is co-funded by the consortium members, grants from Austria, Germany, Belgium, Italy, Spain, and Romania.

REFERENCES

Amin Naseri, M., S. Tasouji Hassanpour, and N. Nahavandi. 2015. "Solving re-entrant no-wait flow shop scheduling problem". *International Journal of Engineering* 28(6):903–912.

Belloso, J., A. A. Juan, and J. Faulin. 2019. "An iterative biased-randomized heuristic for the fleet size and mix vehicle-routing problem with backhauls". *International Transactions in Operational Research* 26(1):289–301.

Chen, J.-S., J. C.-H. Pan, and C.-K. Wu. 2008. "Hybrid tabu search for re-entrant permutation flow-shop scheduling problem". *Expert Systems with Applications* 34(3):1924–1930.

Choi, S.-W., and Y.-D. Kim. 2007. "Minimizing makespan on a two-machine re-entrant flowshop". *Journal of the Operational Research Society* 58(7):972–981.

Danping, L., and C. K. Lee. 2011. "A review of the research methodology for the re-entrant scheduling problem". *International Journal of Production Research* 49(8):2221–2242.

Dominguez, O., A. A. Juan, I. A. de la Nuez, and D. Ouelhadj. 2016. "An ILS-biased randomization algorithm for the two-dimensional loading HFVRP with sequential loading and items rotation". *Journal of the Operational Research Society* 67(1):37–53.

Eskandari, H., and A. Hosseinzadeh. 2014. "A variable neighbourhood search for hybrid flow-shop scheduling problem with rework and set-up times". *Journal of the Operational Research Society* 65(8):1221–1231.

Ferone, D., A. Gruler, P. Festa, and A. A. Juan. 2019. "Enhancing and extending the classical GRASP framework with biased randomisation and simulation". *Journal of the Operational Research Society* 70(8):1362–1375.

Ferrer, A., D. Guimarans, H. Ramalhinho, and A. A. Juan. 2016. "A BRILS metaheuristic for non-smooth flow-shop problems with failure-risk costs". *Expert Systems with Applications* 44:177–186.

Fikar, C., A. A. Juan, E. Martinez, and P. Hirsch. 2016. "A discrete-event driven metaheuristic for dynamic home service routing with synchronised trip sharing". *European Journal of Industrial Engineering* 10(3):323–340.

Gao, Z., J. F. Bard, R. Chacon, and J. Stuber. 2015. "An assignment-sequencing methodology for scheduling assembly and test operations with multi-pass requirements". *IIE Transactions* 47(2):153–172.

Gonzalez-Martin, S., A. A. Juan, D. Riera, Q. Castellà, R. Muñoz, and A. Pérez. 2012. "Development and assessment of the SHARP and RandSHARP algorithms for the arc routing problem". *AI Communications* 25(2):173–189.

Gonzalez-Neira, E. M., D. Ferone, S. Hatami, and A. A. Juan. 2017. "A biased-randomized simheuristic for the distributed assembly permutation flowshop problem with stochastic processing times". *Simulation Modelling Practice and Theory* 79:23–36.

Grasas, A., A. A. Juan, J. Faulin, J. de Armas, and H. Ramalhinho. 2017. "Biased randomization of heuristics using skewed probability distributions: a survey and some applications". *Computers & Industrial Engineering* 110:216–228.

Graves, S. C., H. C. Meal, D. Stefek, and A. H. Zeghmi. 1983. "Scheduling of re-entrant flow shops". *Journal of operations management* 3(4):197–207.

Jia, W., Z. Jiang, and Y. Li. 2013. "Closed loop control-based real-time dispatching heuristic on parallel batch machines with incompatible job families and dynamic arrivals". *International Journal of Production Research* 51(15):4570–4584.

Jing, C., W. Huang, and G. Tang. 2011. "Minimizing total completion time for re-entrant flow shop scheduling problems". *Theoretical Computer Science* 412(48):6712–6719.

Jing, C., G. Tang, and X. Qian. 2008. "Heuristic algorithms for two machine re-entrant flow shop". *Theoretical Computer Science* 400(1-3):137–143.

Juan, A. A., W. D. Kelton, C. S. Currie, and J. Faulin. 2018. "Simheuristics applications: dealing with uncertainty in logistics, transportation, and other supply chain areas". In *Proceedings of the 2018 Winter Simulation Conference*, 3048–3059. IEEE.

Kaihara, T., N. Fujii, A. Tsujibe, and Y. Nonaka. 2010. "Proactive maintenance scheduling in a re-entrant flow shop using Lagrangian decomposition coordination method". *CIRP annals* 59(1):453–456.

Kim, H., and D.-H. Lee. 2009. "Heuristic algorithms for re-entrant hybrid flow shop scheduling with unrelated parallel machines". *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 223(4):433–442.

Lin, D., C. Lee, and W. Ho. 2013. "Multi-level genetic algorithm for the resource-constrained re-entrant scheduling problem in the flow shop". *Engineering Applications of Artificial Intelligence* 26(4):1282–1290.

Marmol, M., L. d. C. Martins, S. Hatami, A. A. Juan, and V. Fernandez. 2020. "Using biased-randomized algorithms for the multi-period product display problem with dynamic attractiveness". *Algorithms* 13(2):34.

Mousavi, S., I. Mahdavi, J. Rezaeian, and M. Zandieh. 2018. "An efficient bi-objective algorithm to solve re-entrant hybrid flow shop scheduling with learning effect and setup times". *Operational Research* 18(1):123–158.

Nawaz, M., E. E. Enscore Jr, and I. Ham. 1983. "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem". *Omega* 11(1):91–95.

Pan, J.-H., and J.-S. Chen. 2003. "Minimizing makespan in re-entrant permutation flow-shops". *Journal of the operational Research Society* 54(6):642–653.

Quintero-Araujo, C. L., J. P. Caballero-Villalobos, A. A. Juan, and J. R. Montoya-Torres. 2017. "A biased-randomized metaheuristic for the capacitated location routing problem". *International Transactions in Operational Research* 24(5):1079–1098.

Quintero-Araujo, C. L., A. Gruler, A. A. Juan, and J. Faulin. 2019. "Using horizontal cooperation concepts in integrated routing and facility-location decisions". *International Transactions in Operational Research* 26(2):551–576.

Taillard 2020. "Metaheuristics and Innovative Solution Techniques". http://http://mistic.heig-vd.ch/taillard/. [Online; accessed September-8-2020].

Taillard, E. 1993. "Benchmarks for basic scheduling problems". *European Journal of Operational Research* 64(2):278 – 285.

Wu, C.-C., S.-C. Liu, T. Cheng, Y. Cheng, S.-Y. Liu, and W.-C. Lin. 2018. "Re-entrant flowshop scheduling with learning considerations to minimize the makespan". *Iranian Journal of Science and Technology, Transactions A: Science* 42(2):727–744.

Yu, T.-S., and M. Pinedo. 2020. "Flow shops with reentry: Reversibility properties and makespan optimal schedules". *European Journal of Operational Research* 282(2):478–490.

AUTHOR BIOGRAPHIES

ANGEL A. JUAN is a Full Professor of Operations Research & Industrial Engineering in the Computer Science Dept. at the Universitat Oberta de Catalunya (Barcelona, Spain). He is also the Director of the ICSO research group at the Internet Interdisciplinary Institute and Lecturer at the Euncet Business School. Dr. Juan holds a Ph.D. in Industrial Engineering and an M.Sc. in Mathematics. He completed a predoctoral internship at Harvard University and postdoctoral internships at

Massachusetts Institute of Technology and Georgia Institute of Technology. His main research interests include applications of simheuristics and learnheuristics in computational logistics and transportation, as well as computational finance. He has published about 100 articles in JCR-indexed journals and more than 215 papers indexed in Scopus. His website address is http://ajuanp.wordpress.com and his email address is ajuanp@uoc.edu.

CHRISTOPH LAROQUE studied business computing at the University of Paderborn, Germany. Since 2013 he is Professor of Business Computing at the University of Applied Sciences Zwickau, Germany. He is mainly interested in the application of simulation-based decision support techniques for operational production and project management. His email address is Christoph.laroque@fh-zwickau.de.

PEDRO J. COPADO-MENDEZ is a Post-doctoral researcher in the ICSO group at the IN3 – Universitat Oberta de Catalunya. He completed his PhD at the University Rovira i Virgili (Spain). His main research interests include metaheuristics, hybrid heuristics and their applications. His email address is pcopadom@uoc.edu.

JAVIER PANADERO is an Assistant Professor of Simulation and High Performance Computing in the Computer Science Dept. at the Universitat Oberta de Catalunya. He is also a Lecturer at the Euncet Business School. He holds a Ph.D. and a M.S. in Computer Science. His major research areas are high performance computing and simheuristics. He has co-authored more than 40 scientific articles. His website address is http://www.javierpanadero.com and his email address is jpanaderom@uoc.edu.

ROCIO DE LA TORRE is an Assistant Professor in the Department of Business Management at the Public University of Navarre (Spain). She is also a researcher from INARBE-Institute for Advanced Research in Business and Economics. She holds a Ph.D. and a Bachelor Degree in Industrial Engineering from the Universitat Politécnica de Catalunya. Her major research areas are: mathematical programming for strategic planning decisions and the supply chain design. Her email address is rocio.delatorre@unavarra.es.