ESAIM 2024 – 2nd European Symposium on Artificial Intelligence in Manufacturing

LOS data set: A Large Scale Online Scheduling Benchmark for Flexible Job Shop Problems with Setup and Transportation Times

Katharina Hengel[1,*], Achim Wagner[1], and Martin Ruskowski[1,2]

[1] German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany
[2] Technologie-Initiative SmartFactory KL e.V., Kaiserslautern, Germany

[*] Corresponding author. Tel.: +49-631205757246;. E-mail: katharina.hengel@dfki.de

Abstract. With an increased flexibility in the production new scheduling techniques are necessary to accommodate this change. Though there have already been published many scheduling algorithms fostering this demand for flexibility, there is no common ground on a benchmark data set to compare these approaches against each other. Therefore, this paper aims at the generation of a benchmark data set for the flexible job shop problem (FJSP) with setup and transportation times on which different scheduling algorithms can be evaluated. The data set is specified by several key parameters from which FJSP are created. The use and advantage of LOS is exemplified by its application on a Reinforcement Learning online scheduling algorithm and dispatching rules. Furthermore, backward compatibility is established with the former FJSP notation.

Keywords: Benchmark · Scheduling · Flexible Job Shop · Data Set

1 Introduction

Industry 4.0 increases the flexibility of the production in the shop floor. Hence, new approaches for scheduling are required to accommodate for the accompanying changes introduced in Industry 4.0. On the hardware side the flexibility is often enabled by the use of modular production systems. In this connection, the resulting scheduling problem is formally defined by the flexible job shop problem (FJSP) with setup and transportation times. Though there have already been published many scheduling algorithms fostering this demand for an increased flexibility, there is no common ground on a benchmark data set to compare these approaches against each other.

Towards this end, the data sets which are used are either not published and explained in detail, small scale or only have few instances. This makes a broad adaptability difficult. Hence, a comparison between the algorithms as well as the evaluation of its advantages and disadvantages is not possible. Therefore, we present a benchmark data set for the FJSP with setup and transportation times on which different scheduling algorithms can be evaluated.

The data set is specified by several key parameters from which FJSPs are created. A new notation is introduced for better extensibility and readability reasons. The use and advantage of LOS is exemplified by its application on a Reinforcement Learning online scheduling algorithm and dispatching rules. Furthermore, backward compatibility is established with the former FJSP notation. As example the existing FJSP data set published by [12] is converted to our notation. This allows the comparison of the results between previous and newly published algorithms using either of these notations.

The remainder of this paper is organized as follows. An introduction to the FJSP with setup and transportation times as well as an overview for existing scheduling data sets is given in section 2. Afterwards the method for the creation of the FJSP data set is introduced in section 3. At last a RL scheduling algorithm and dispatching rules are applied on the data sets. The results are demonstrated in section 4 and discussed in section 5.

2 State-of-the-art

2.1 Flexible Job Shop Problem

[14] gives an overview of the taxonomy of scheduling problems. To handle a high degree of flexibility in the shop floor we focus on online scheduling for FJSP with setup and transportation times.

Given are n jobs $J = \{j_1, j_2, ..., j_n\}$ each one being composed of k operations $O_i = \{o^i_1, o^i_2, ..., o^i_k\}$. The operations must be processed in a given order defined by precedence constraints. Every operation must be processed on one of m machines $M = \{m_1, m_2, ..., m_m\}$. A machine can only process one operation at a time. While an operation o^i_j in the general job shop problem (JSP) must be processed on one specific machine, the FJSP relaxes this condition and an operation can be process by a specified subset $M_{ij} \subset M$ of all machines. The processing time of an operation o^i_j on machine l is given by p^{ij}_l. Between the processing of two different jobs on a machine, a setup time might occur. The setup time is denoted by s^{ij}_l with l being the machine id and i and j being the operation indexes. Furthermore, t_{lk} denotes the transportation time of a job between the machines l and k.

2.2 Benchmarks

A collection of FJSP instances is provided by [12]. This collection contains in total 313 selected benchmark instances from [5], [9], [7] and [3]. The FJSP instances date from the years 1993-1997. All instances use the standard FJSP notation syntax. In the first line the number of jobs and the number of machines is specified. It is followed by a line for each job. Each job specification starts with the number of operations of the job. It is followed by a sequence for each operation specifying the number of machines which are able to process the operation and the machine ids and the processing times of them.

[5] defined the following parameters:

- number of jobs
- number of machines
- minimum and maximum number of operations per job
- maximum number of equivalent machines per operation
- minimum and maximum processing time per operation

The data set was then generated randomly using a uniform distribution to select the parameters within the given limit.

In [9] selected job shop instances from [8] and [1] were adapted to the problem of job shop scheduling with multi-purpose machines. It differs to the FJSP therein that the processing time of an operation on a machine is not dependent of the machine which has been chosen for processing the operation. The data set consists of four subsets: sdata, edata, rdata and vdata. The original instances from [8] and [1] build the sdata set. Here $|M_{ij}| = 1$ holds. In the three remaining data sets M_{ij} is enlarged and limited by the average and maximal cardinality of M_{ij}, i.e. $\operatorname{avg}_{i,j} |M_{ij}|$ and $\max_{i,j} |M_{ij}|$. Hereby, the edata represents the instances where only few operations can be processed by different machines. In the rdata most operations can be assigned to a few number of different machines and in the vdata each operation can be assigned to many different machines. The employed parameter selection for the data is depicted in Table 1.

Table 1. Parameter assignment for the FJSP instances by [9]

	$\operatorname{avg}_{i,j} \lvert M_{ij} \rvert$	$\max_{i,j} \lvert M_{ij} \rvert$
edata	1.15	2, if $m \leq 6$ 3, otherwise
rdata	2	3
vdata	0.5 m	0.8 m

For the generated FJSP in [7] the number of machines and jobs is fixed. Furthermore the limits for the number of operations per job is given. The processing time is defined by the average processing time given as range and its maximal deviation from it. The parameters were selected randomly between the given limits using a uniform distribution. A probability P is set for each instance. P is the probability of a machine l being in M_{ij}. In the case M_{ij} would be empty a machine $\bar{l}$ is selected randomly and $M_{ij} = \{\bar{l}\}$.

The data set of [3] extends the MT10 instance of [8] and the instances LA24 and LA40 of [10]. These instances have been replicated in the following way:

- the machine requiring the greatest cumulative processing time (CPT) is replicated once
- the machine requiring the greatest CPT is replicated twice
- the machine requiring the greatest CPT is replicated three times
- the machines requiring the greatest and second-greatest CPTs are replicated once each
- the machines requiring the greatest, second-greatest, and third-greatest CPTs are replicated once each
- the machine with the greatest number of critical operations is replicated once
- the machines with the greatest and second-greatest number of critical operations are replicated once each

Since the FJSP is a generalization of the JSP, every flexible job shop scheduling algorithm can also be applied to a JSP. A large data set of JSPs is contained in the OR-library [4]. It includes JSP by [16], [1], [8], [10], [2], [15] and [11]. Since the OR-library's JSP instance are rather small, [6] created a large scale set of JSP. In their first approach they hereby extended the benchmark set of [16].

3 Method

To foster scalability of the FJSP instances and the inclusion of further parameters like the deadline of a job or the energy consumption of a machine processing an operation, we want to shift the view point of the notation away from the product to the resource and process. We follow the understanding of skills and tasks and its relation to product, process and resources explained in [13]. While the term skill here relates to the resources and processes, tasks are related to the products. Hence, instead of defining the processing time for an operation, we define the processing time of a skill. Furthermore, we define the setup time of a machine between skills and a job as sequence of skills. In this way, we can scale the number of jobs upwards while limiting the skills without a blow-up of the instance specification.

For the generation of our benchmark data set we used approaches from [5] and [9]. We defined the following parameters:

- maximal and minimal number of machines
- maximal and minimal number of jobs
- number of skills
- maximal and minimal number of skills per job
- average number of machines per skill, i.e $\operatorname*{avg}_{i,j} |M_{ij}|$
- maximal number of machines per skill, i.e $\max_{i,j} |M_{ij}|$
- maximal and minimal processing time
- maximal and minimal setup time

– scaling factor for transportation times

Based in these parameters the instance is constructed. For the transportation times the machines are arranged in a square matrix. The distance between two machines is then defined by the manhattan distance between the machines multiplied by the scaling factor for the transportation times. The skills of a machine are selected following the approach of [9]. Using a triangular distribution with lower limit 1, upper limit $\max\limits_{i,j} |M_{ij}|$ and mode $\operatorname*{avg}\limits_{i,j} |M_{ij}|$ the number of machines per skill is defined. Afterwards the selected number of machines are sampled randomly from the set of machines. The processing time of the machine for the skill is randomly sampled from the uniform distribution between the given limits. The changeover times are selected randomly using a uniform distribution within the given limits. For the definition of the jobs we select the number of jobs, for each job the number of skills and for each skill the skill itself uniformly at random.

As described earlier we shifted the notation to a resource and process based notation. Thus, we discontinued with the standard FJSP notation. Instead, we used a yaml based syntax for the LOS data set instances. This also allows easier extensibility and readability. Each FJSP is described by a map with the keys distances, skills, changeovers and jobs. The values are represented as (nested) lists. To ensure the comparison with benchmarks specified by the standard FJSP notation, we established backwards compatibility with it. By setting the setup and transportation times to 0 and mapping every operation in the former notation to a different skill in our yaml notation, the comparison to previous used benchmarks can be established.

4 Results

We applied the dispatching rules of Table 2 and an scheduling algorithm using Reinforcement Learning (RL) on the benchmark data set of [12] as well as on the LOS data set. Since the setup and transportation times are 0 for the instances in [12], the dispatching rules SST and SST + SPT are not applied in this case. Figure 1 - Figure 5 show the results of the scheduling algorithms illustrated as box plot created by 21 FJSP instances of [3], 10 of [5], 18 of [7], 264 of [9] and 100 of the LOS data set.

Table 2. Overview of dispatching rules

Abbreviation	Dispatching Rule	
SIRO	Setup in Random Order	
SST	Shortest Setup Time	$\min p_l^{ij}$
SPT	Shortest Processing Time	$\min s_{ijl} + t_{lk}$
SPT + SST	Shortest Processing + Setup time	$\min p_l^{ij} + s_{ijl} + t_{lk}$

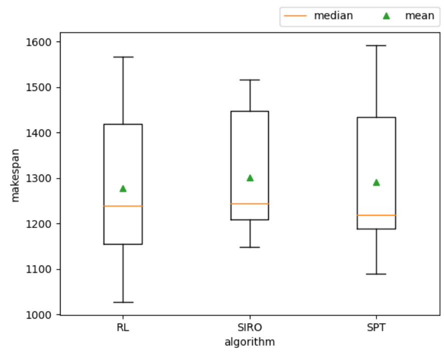

Fig. 1. Evaluation of dispatching rules and an RL scheduling algorithm on the data set by [3]

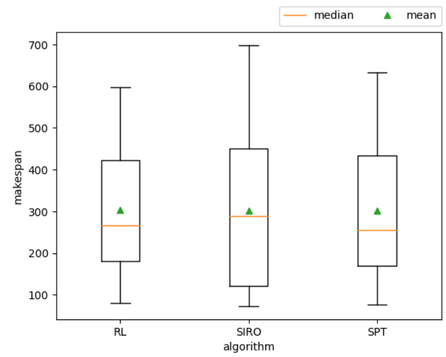

Fig. 2. Evaluation of dispatching rules and an RL scheduling algorithm on the data set by [5]

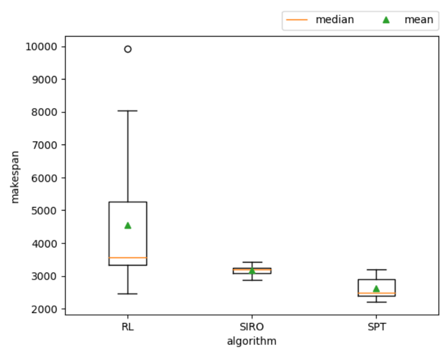

Fig. 3. Evaluation of dispatching rules and an RL scheduling algorithm on the data set by [7]

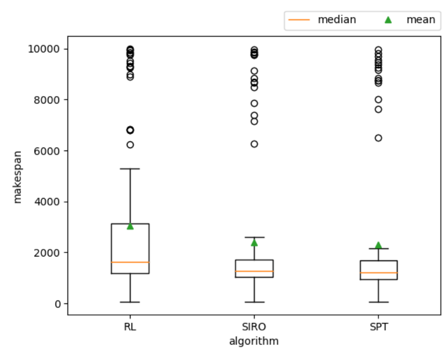

Fig. 4. Evaluation of dispatching rules and an RL scheduling algorithm on the data set by [9]

For the results of the data set of [12] (Figure 1 - Figure 4) the results do not vary much between the different algorithms. The boxes as well as the whiskers overlap most of the time. The largest difference of the mean respectively the median occurs in the data set of [7]. Here the median differs between the RL algorithm and the SPT dispatching rule by 1063, the mean respectively by 1922.5. While RL performs the best in the data set of [3], SPT has the best performance in the data sets of [7] and [9]. In [5] RL and SPT perform nearly the same.

For the evaluation of the LOS data set, we created over 8000 FJSP instances with the parameter assignment depicted in Table 3. Figure 5 shows the results evaluated on 100 FJSP instances of the RL scheduling algorithm and the dispatching rules. In contrast to the Figures 1-4 not all boxes for the different scheduling algorithms overlap anymore. The mean performance of the algorithm increases in the order RL, SST+SPT, SPT, SST, SIRO with values 3007.3,

Table 3. Parameter assignment for the LOS data set

maximal and minimal number of machines	25
maximal and minimal number of jobs	100
number of skills	25
maximal and minimal number of skills per job	5-8
average number of machines per skill: $\underset{i,j}{\mathrm{avg}}\, \lvert M_{ij} \rvert$	12.5
maximal number of machines per skill: $\max\limits_{i,j} \lvert M_{ij} \rvert$	20
maximal and minimal processing time	10-100
maximal and minimal setup time	10-50
scaling factor for transportation times	20

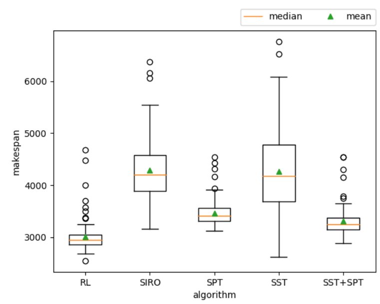

Fig. 5. Evaluation of a RL scheduling algorithm and multiple dispatching rules on the LOS benchmark with parameter selection according to Table 3

3308.8, 3466.35, 4262.19, 4287.16. While the dispatching rules SIRO and SST have a rather larger box and long whiskers, the RL algorithm as well as the dispatching rules SPT and SST+SPT have a smaller box with shorter whiskers.

5 Discussion

The boxes and whiskers of the different scheduling approaches all overlap in the data sets of [3] and [5]. Hence, there is no advantage between the scheduling algorithms visible. The results are only slightly better for the data sets of [7] and [9]. Here one can see an advantage of the dispatching rules SIRO and SPT in contrast to the RL algorithm. These observation suggest, that the RL algorithm is not able to generalize well. One possible reason is that there are too few instances for training of the RL algorithm. Since there is also no advantage between the dispatching rules in Figure 1 and Figure 2 and only a slight advantage in Figure 3 and Figure 4, this also suggests that the data sets only have few potential for online scheduling. Since they were created in the 90s before the fourth industrial revolution started, they were invented for offline scheduling rather than online scheduling. At that time manufacturing systems were rather static and fixed. Thus, offline scheduling techniques where production plans can be computed in advance are generally in favor, since they can achieve more optimal results. With Industry 4.0 dependencies within and among manufacturing systems increased. Hence, flexibility and agility became more important and with this the necessity for online scheduling approaches. This makes these ancient data sets insufficient and outdated for the current research.

For this reason, a new FJSP data set for online scheduling is necessary. By adjusting the values of e.g. the transportation times, setup times or the processing times, we can vary the impact of these factors. This affects the performance of the scheduling algorithm. Hence, we can see a real advantage of the RL algorithm as well as the SST+SPT and SPT dispatching rules over the SIRO and SST dispatching rule in the FJSP as one would expect. This is evident by a visible shift in the boxes and the whiskers, as well as their absolute length. The later is indicating a lower variance in the scheduling results, making the schedules more robust.

The FJSP instance is variably adjustable by its parameter selection. In this way we can scale up the instances itself and examine scheduling algorithms on large scale instances. On the other hand it is also possible to generated numerous amounts of instances for the given parameters. Thus we can generate a large data set with sufficient training instances for e.g. Reinforcement Learning based algorithms to train on. Due to the explicit characterization of the used data set by its parameters (cp. Table 3) the FJSP instances can be recreated. Hence, a comparison of different scheduling algorithms is possible.

6 Conclusion and Future Work

We present a method for the creation of a benchmark data set for FJSP with setup and transportation times. Furthermore, we ensure backwards compatibility to former FJSP data sets. We present results for different dispatching rules on the data set showing its potential for optimization using online scheduling. While the data set was invented for online scheduling, it is not limited to it and can also be used for offline scheduling. Through the potential of the data set for offline scheduling needs to be investigated in further studies. Besides that, future research might focus on the extension of the data set to holonic manufacturing systems and how the holons can be represented in the data set and considered during generation.

Acknowledgement. This work has been funded by the European Commission through the HORIZON-CL4-2022-HUMAN-02 project HumAIne - Hybrid Human-AI Decision Support for Enhanced Human Empowerment in Dynamic Situations (No.101120218).

References

1. Adams, J., Balas, E., Zawack, D.: The shifting bottleneck procedure for job shop scheduling **34**, 391–401 (1988). https://doi.org/10.1287/mnsc.34.3.391
2. Applagate, D., Cook, W.: A computational study of the job-shop scheduling instance. ORSA Journal on Computing **3**, 149–156 (1991)
3. Barnes, J., Chambers, J.: Flexible job shop scheduling by tabu search. Graduate Program in Operations and Industrial Engineering, The University of Texas at Austin, Technical Report Series, ORP96-09 (1996)
4. Beasley, J.E.: Or-library: distributing test problems by electronic mail. Journal of the operational research society **41**(11), 1069–1072 (1990), http://people.brunel.ac.uk/ mastjjb/jeb/info.html
5. Brandimarte, P.: Routing and scheduling in a flexible job shop by tabu search. Annals of Operations research **41**(3), 157–183 (1993)
6. Col, G.D., Teppan, E.: Large-scale benchmarks for the job shop scheduling problem (Jan 2021)
7. Dauzère-Pérès, S., Paulli, J.: An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search. Annals of Operations Research **70**, 281–306 (1997)
8. Fisher, H.: Probabilistic learning combinations of local job-shop scheduling rules. Industrial scheduling pp. 225–251 (1963)
9. Hurink, J., Jurisch, B., Thole, M.: Tabu search for the job-shop scheduling problem with multi-purpose machines. Operations-Research-Spektrum **15**(4), 205–215 (1994)
10. Lawrence, S.: Resouce constrained project scheduling: An experimental investigation of heuristic scheduling techniques (supplement). Graduate School of Industrial Administration, Carnegie-Mellon University (1984)
11. Li, Y., Manner, R., Manderick, B.: A genetic algorithm applicable to large-scale job-shop instances. In: Parallel Problem Solving from Nature, pp. 281–290. North-Holland (1992)
12. Mastrolilli, M.: Flexible job shop problem, https://people.idsia.ch/~monaldo/fjsp.html
13. Pfrommer, J., Schleipen, M., Beyerer, J.: Pprs: Production skills and their relation to product, process, and resource. In: 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation (ETFA). pp. 1–4. IEEE (2013)
14. Pinedo, M.L.: Scheduling, vol. 29. Springer (2012)
15. Storer, R., Wu, S., Vaccari, R.: New search spaces for sequencing instances with application to job shop 38 (1992) 1495–1509manage. Sci **38**, 1495–1509 (1992)
16. Taillard, E.: Benchmarks for basic scheduling problems. european journal of operational research **64**(2), 278–285 (1993)