Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 55-10 (2022) 1237–1242

Impact of Encoding and Neighborhood on Landscape Analysis for the Job Shop Scheduling Problem

Israël Tsogbetse * **Julien Bernard** ** **Hervé Manier** * **Marie-Ange Manier** *

* *FEMTO-ST, CNRS, UTBM, 90010 Belfort, France, ({israel.tsogbetse, herve.manier, marie-ange.manier}@utbm.fr).*
** *FEMTO-ST, CNRS, UFC, 25000 Besançon, France (julien.bernard@femto-st.fr)*

Abstract: In order to improve the performance of metaheuristics, many studies focus on the efficiency of neighborhood operators and on the hybridization of various methods. Though, the solution encoding scheme has an important role in the search space's definition, their choice is usually made without any relevance study nor valuable justification. Thus, the present paper aims at conducting analysis on fitness landscapes generated by a basic job shop scheduling problem testing three encoding schemes combined with three operators to optimize the makespan scheduling criterion. Through the study of three different metrics, we show that the encoding schemes associated to neighborhood operators play a major role in the structure of the fitness landscape.

Keywords: optimization, job shop, encoding, neighborhood, fitness landscape

1. INTRODUCTION

Solving a scheduling problem means planning the execution of a set of tasks while respecting time constraints (deadlines, precedence, etc) and the capacity constraints on the resources used (Carlier and Chrétienne (1988)). In a job shop scheduling problem (JSP), tasks are grouped into jobs with preset sequences. In spite of exact optimization methods which are suitable for small scale problems, metaheuristics (Mihoubi et al. (2021), Madraki et al. (2022)), and artificial intelligence (Liu et al. (2022)) in general, are often used to provide good quality solutions when the complexity of problems increases.

Since the 1980's, intensive studies have been undertaken to better understand the dynamics of optimization methods according to Sörensen et al. (2018). Properties derived from this analytic works help to refine and justify the choices of the various components of optimization methods. Moreover characteristics of problem types and instances strongly influence the performance of these methods. Therefore, determining the relevance of the components of metaheuristics is necessary to understand how they contribute to the achievement of good solutions. Verel (2016) and Tari (2019) conducted researches on fitness landscapes to characterize binary and permutation landscapes. These studies mostly lay emphasis on the impact of neighborhood operators with respect to different classes of optimization problems. Notwithstanding the importance of the neighborhood operators, the encoding scheme also called solution representation is equally a crucial element on which the effectiveness of a heuristic may depend. Several encoding schemes have been used in optimization algorithms for the job shop scheduling problem. Yet, the choice of these encoding schemes is often based on intuition without real justification. This can lead the algorithm to obtain poor results for the problem at hand.

The goal of this paper is to study the impact of different encoding schemes and neighborhood operators on the fitness landscape of job shop scheduling problems. Through a series of experiments on well-known job shop scheduling benchmarks, we analyze the neutrality rate, the correlation length and the position type distribution of the fitness landscapes. We show that these metrics are tied to the encoding schemes in conjunction with neighborhood operators, the instances playing a minor role in the metrics.

The rest of the paper is structured as follows. Section 2 reviews existing researches of encoding schemes for scheduling problems and fitness landscapes measures. In section 3, a short description of the JSP is given. Section 4 provides details on the search space settings (encoding scheme, neighborhood operators) and metrics that were used for the experiments. The results obtained through experiment are presented in section 5.

2. RELATED WORK

Encoding schemes, also called solution representations, arouse some interest regarding scheduling problems. Bierwirth (1995) introduced a new encoding based on permutation with repetition for JSP to compensate the weakness of standard binary and permutation encodings according to the presence of infeasible solutions. The influence of synonymously and non-synonymously redundant representations on genetic algorithm (GA) performances are studied by Rothlauf and Goldberg (2003). For the unrelated machine environment, Durasević and Jakobović (2016)

10.1016/j.ifacol.2022.09.559

compared two encodings, one based on permutation and the second on floating point, for different criteria. In the same logic, Fernandez-Viagas et al. (2019) compared four categories of solution representation for the hybrid flow shop scheduling problem. Their research lay emphasis on a possible link between the solution spaces generated by encodings and the quality of produced schedules. Vlašic et al. (2020) brought out a comparative study on seven encodings to prove the impact of encoding on GA results. Though comparison of different encoding schemes may determine one most suitable for a given problem, other researches try to understand the dynamics of search spaces and exploration through fitness landscape analysis. Works on landscapes mostly target problem hardness, effectiveness of neighborhood structures and landscape analysis techniques. For the JSP, Mattfeld et al. (1999) illustrated the differences between landscapes of hard and easy instances. To facilitate the characterization of fitness landscape, a quiet large survey of existing techniques has been provide by Malan and Engelbrecht (2013). Verel (2016) for his part, proposed a new model of NK fitness landscape based on local optima network. Later, Tari (2019) studied mechanisms of optimization algorithms to improve their understanding by the use of fitness landscapes analysis. Either binary (NK, UBQP) or permutation (FSP, QAP), each problem was handled with an operator chosen among 1-flip, insertion, swap. In the same scope, Merkuryeva and Bolshakovs (2020) concluded that the results of this type of analysis depends on many parameters such as solution representations, search policies and explained how measures are sensitive to random noise.

3. PROBLEM DESCRIPTION

The job shop scheduling problem (JSP) consists in scheduling jobs that have predefined sequences of operations (tasks) in a multi-machine environment. For each job, the operations are executed in a given order called sequence. The sequence varies from one job to another creating a multidirectional flow. Generally, a scheduling problem is made of tasks assignment (assigning resources to the tasks) and sequencing sub-problems (determining one execution order of tasks on those resources-and associated starting times). The JSP has several variants depending on workshops constraints. For more details, see Choi and Choi (2002) and Esquirol and Lopez (2008).

A JSP is defined by a set of n jobs $\{J_1, J_2, \ldots, J_n\}$ and a set of m machines $\{M_1, M_2, \ldots, M_m\}$. Each job J_i is composed of n_i operations $\{O_{i,1}, O_{i,2}, \ldots, O_{i,n_i}\}$. Each operation $O_{i,j}$ is assigned to a single machine which executes it in a processing time of p_{ij}. A job can only start processing an operation when the previous operation in the list is finished *(precedence constraint)*. See French (1982) and Caumond (2006). Jobs are available from the start of the system, release date r for all jobs is zero ($\forall i \in [1, n], r_i = 0$). All the information about the scheduling problem is known in advance. Setup times are included in the processing times while transportation times are negligible. Assumptions are made that products can wait in unlimited capacity stocks and a machine can only perform one task at a time, which involves *disjunctive constraints*. Once an operation is started, it cannot be interrupted until the execution ends (*no preemption*). According to the derived schedule, the time when a job i completes its execution is called completion time and denoted by C_i. The objective function considered in this study consists in minimizing the makespan $C_{\max} = \max_{1 \leq i \leq n} C_i$.

Table 1. Small JSP instance

Job	$O_{i,1}$	$O_{i,2}$	$O_{i,3}$
J_1	$M_1(4)$	$M_2(2)$	$M_3(1)$
J_2	$M_2(7)$	$M_3(3)$	$M_1(2)$

1	1	2	1	2	2

Fig. 1. Solution represented by JLE

4. SEARCH SPACES SETUP AND METRICS

In this section, we describe and analyze the encoding schemes and neighborhood operators used in the experiments. Then we present three metrics that characterize different aspects of the fitness landscape. We also show the different instances that have been used in the experiments.

4.1 Encoding schemes

Job list encoding, operation list encoding and machine list encoding are the selected encoding schemes. In order to illustrate their differences, a solution for each encoding is provided for a small JSP instance composed of two jobs, each with three operations, to be performed on three machines. Table 1 indicates the assigned machine and processing time p_{ij} for each operation $O_{i,j}$.

For each encoding, we use a simple scheduling constructor that tries to determine the earliest starting date for each operation. If the solution is feasible, the fitness $C_{\max}$ is computed from the schedule. Otherwise, the solution is infeasible and no reparation is done.

Inspired by Holsapple et al. (1993), *job list encoding* (JLE) represents a solution by a list of integers. The size of the list is equal to the total number of operations and each element of the list corresponds to a job index. Within this list, the index of a job appears as many times as it has operations. It covers the space of all possible active schedules. Two different solutions can give the same schedule (Bierwirth (1995)). In this encoding, all solutions are feasible as they respect all the constraints. Fig. 1 shows one of the 20 possible solutions for the instance described in Table 1. The first operation of J_1 will be scheduled first, then the second operation of the same job, then the first operation of J_2 followed by the third and the last operation of J_1. The schedule ends with the last two operations of J_2.

Operation list encoding (OLE) is similar to job list encoding. A solution is represented by an ordered list of operations. The operation list is a permutation of the set of all the operations (Vlašic et al. (2020)). Unlike JLE, infeasible solutions may be generated with OLE. The permutations of operations may violate the precedence constraints within a job. Among the possible solutions, the number of feasible solutions is equal to the number of possible solutions for JLE. There is a bijection between the set of feasible solutions of OLE and the set of possible solutions of JLE. Out of 720 possible solutions, the example

(a) | $O_{1,1}$ | $O_{1,2}$ | $O_{2,1}$ | $O_{1,3}$ | $O_{2,2}$ | $O_{2,3}$ |

(b) | $O_{2,1}$ | $O_{1,1}$ | $O_{1,2}$ | $O_{2,3}$ | $O_{1,3}$ | $O_{2,2}$ |

Fig. 2. Solutions represented by OLE

	(a)		(b)	
M_1	$O_{1,1}$	$O_{2,3}$	$O_{2,3}$	$O_{1,1}$
M_2	$O_{1,2}$	$O_{2,1}$	$O_{2,1}$	$O_{1,2}$
M_3	$O_{1,3}$	$O_{2,2}$	$O_{1,3}$	$O_{2,2}$

Fig. 3. Solutions represented by MLE

instance has 700 infeasible solutions. In Fig. 2, solution (a) is a feasible solution corresponding to the solution in Fig. 1 in JLE. Solution (b) is an infeasible solution, as it violates a precedence constraint with respect to operations $O_{2,3}$ and $O_{2,2}$ of J_2.

Machine list encoding (MLE) is a list of m list of operations that are assigned to a common machine, see Vallada and Ruiz (2011). The sizes of the lists of operations vary according to the number of operations for each machine. The permutations on the different lists are likely to cause violations of precedence constraints, hence the presence of infeasible solutions. In contrast to OLE where the number of infeasible solutions is precisely quantifiable, the number of feasible and infeasible solutions is inherently dependent on the instance being processed for MLE. An example of feasible solution (a) and infeasible one (b) with MLE are shown in Fig. 3. The instance has a total of 8 possible solutions including 3 infeasible ones.

4.2 Neighborhood operators

Let S be the set of solutions in a search space, for any solution $s \in S$, there exists a subset $N(s) \subseteq S$ called neighborhood of s given by a neighborhood operator $N : S \to 2^S$. Any element $s' \in N(s)$ is called neighbor of s. Three well-known operators are used in this study.

The *swap* operator consists in exchanging two elements of the solution chosen randomly. The result must not be the same as the initial solution, which could be possible for JLE. In the case of MLE, a machine is first chosen at random and then the swap operator is applied to the operations list of this machine.

The *insertion* operator consists in taking an element of the solution at random and assigning it a new location. Every element located between the old and the new location is shifted by one place in the direction of the old location. MLE is treated like the swap operator case.

The *reverse* operator consists in choosing two elements of the solution at random and reversing all the elements between these two elements included. Again, MLE is treated like the swap operator case.

Fig. 4 illustrates the swap, insertion and reverse neighborhood operators. The size of the neighborhoods varies for each operator.

Lemma 1. For all three neighbor operators (swap, insertion, reverse), a feasible neighbor of a solution s in OLE is also a neighbor in the corresponding solution s' in JLE, where s' is defined as the corresponding jobs of the operations in s.

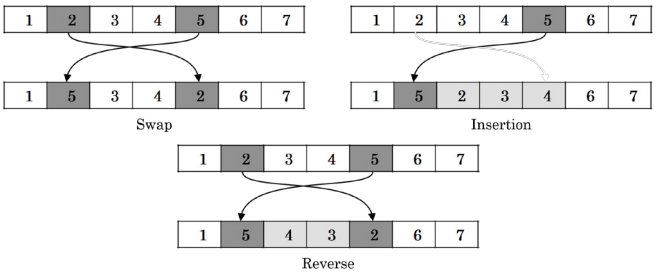

Fig. 4. Illustration of the three neighborhood operators

Let us consider the swap operator (the other two are analog). In order for a neighbor of s to be feasible, all the operations between the two chosen operations must belong to jobs that are different from the jobs of the limits, and the jobs of the limits must also be different. Otherwise, the constraint precedence would not be fulfilled and the solution would not be feasible. Hence, the jobs of the limits will be different and the swap in JLE will provide a different solution than s'.

4.3 Metrics

The main goal of the *fitness landscape analysis* is to understand the structure of the search space from the point of view of an optimization algorithm. According to Jarboui et al. (2013), the definition of the landscape allows not only to answer questions related to a better modeling (encoding, neighborhood, etc) of a given problem but also to guide the choice of a metaheuristic and its settings. The landscapes of the selected encoding schemes and neighborhood operators will be studied using the following metrics.

Neutrality occurs in a landscape when two neighbor solutions have the same fitness value. A neutral neighbor s' of a solution s is a solution such that $s' \in N(s)$ and $f(s) = f(s')$. A landscape is considered neutral if a significant proportion of pairs of neighbor solutions are neutral. A neutral landscape thus does not imply a flat landscape but rather the presence of successive neutrality, which can be manifested by features such as plateaus and peaks in a landscape. Neutrality rate N_r of solution s denotes its proportion of neutral neighbors. It is calculated with (1). Landscape neutrality is given by the average of all neutrality rates along a random walk within the search space. Fitness landscape is often considered to be statistically isotropic. See Malan and Engelbrecht (2013) and Tari (2019).

$$N_r = \frac{|\{s' \in N(s), f(s') = f(s)\}|}{|N(s)|} \tag{1}$$

Roughness is concerned with the number and distribution of local optima and the size of their basins of attraction. A landscape is said to be rough if the neighboring solutions have very different fitness values. On the contrary, a smooth landscape is one where the neighbors have almost the same fitness value. One of the most widely used indicator for roughness measurement is the auto-correlation function. Auto-correlation is calculated from a set of random walks and corresponds to the fitness correlation between the solutions encountered during a random walk as a function of the distance between them. The

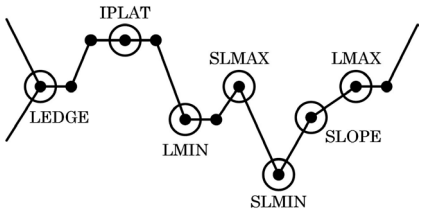

Fig. 5. Position types in fitness landscape

auto-correlation $\rho(n)$ is given by (2) where $\rho(n) \in [-1, 1]$; $n \in \{0, \ldots, t\}$ is the number of steps from the initial solution x_0, l the total number of solutions, f_t the fitness of the t^{th} solution, f_{t+n} is the fitness of the solution after n steps counted from the t^{th}, $\overline{f}$ is the average of all fitness. $|\rho(n)| = 1$ indicates a maximum correlation and a value close to 0 indicates absence of correlation.

$$\rho(n) = \frac{\frac{1}{l-n} \times \sum_{t=1}^{l-n}(f_t - \overline{f})(f_{t+n} - \overline{f})}{\frac{1}{l} \times \sum_{t=1}^{l}(f_t - \overline{f})^2} \tag{2}$$

Using the auto-correlation function, the correlation length τ (the distance beyond which the majority of points become uncorrelated) is obtained by (3). A small value of τ denotes a rougher landscape. See Weinberger (1990) and Kauffman (1993).

$$\tau = \frac{-1}{\ln(|\rho(1)|)} \tag{3}$$

Position type distribution is a measure that classify a solution according to the fitness differences observed in its neighborhood. There are seven types of solutions: SLMIN, SLMAX, LMIN, LMAX, SLOPE, LEDGE and IPLAT. The type of point is SLMAX (SLMIN) if it is strictly a local maximum (minimum), while the types LMAX and LMIN denote non-strict local maxima and minima respectively. A solution s is a SLOPE if some of its neighbors have a larger fitness value and others a smaller value than s. A LEDGE solution s has neighbors with the same fitness value as s and neighbors with larger and smaller fitness values than s. A solution s belongs to the IPLAT type if all its neighbors have the same fitness value as s. Point types are shown in Fig. 5. The distribution of fitness points produces a quantitative analysis on the characterization of a fitness landscape. A fitness landscape is more neutral if the percentage of IPLAT, LMIN, LMAX and LEDGE is higher. See Czogalla and Fink (2012) and Zhao et al. (2019).Position type distribution requires a uniform selection of feasible solutions within the search space. So it is almost impossible to get such selection with MLE since it is hard to find feasible solutions with a random selection. OLE has the same problem, but there is a bijection between the feasible solutions in OLE and the possible solutions of JLE. So we select a random solution in JLE and transform it in OLE to achieve uniform selection.

4.4 Instances

Experiments are performed on a different sets of problem instances from several classical JSP benchmarks. We select different sets to ensure the experiments are done on instances with different characteristics. For each set, we select instances with a wide range of jobs and machines.

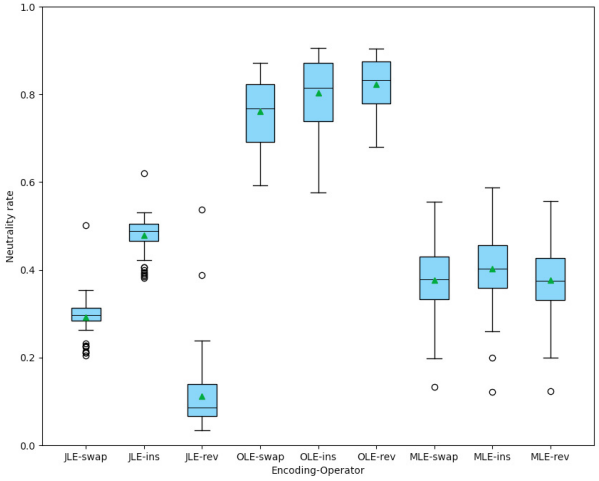

Fig. 6. Distribution of neutrality rate

Thus, we have a total of 51 instances that vary from 2 jobs on 2 machines to 100 jobs on 20 machines. The problem instances were taken from Taillard; Yamada & Nakano; Storer, Wu & Vaccari; Applegate & Cook; Lawrence; Adams, Balas & Zawack; Fisher & Thompson; Caumond and Lopez benchmarks.

5. RESULTS AND DISCUSSION

5.1 Neutrality rate

In this experiment, for each encoding scheme and neighborhood operator, for each instance, we select 10 random solutions and make a random walk of 1000 steps. We compute the neutrality rate as stated before and the average on the 10 walks, which gives a neutrality rate for the instance. We concatenate all the neutrality rates over all the instances in a boxplot. Fig. 6 shows the neutrality rate of all the instances for each pair of encoding schemes and neighborhood operators.

We observe that the neutrality rate highly depends on the encoding schemes and neighborhood operators, more than on the instances. It's especially visible for JLE encoding where the boxplots corresponding to each operator are totally disjoint. OLE and MLE seem to not be sensitive to the neighborhood operator. In fact, for these encodings, the neighbors of a solution are very similar for the three operators. As stated in the proof of lemma 1, in order to be feasible, the two indices that are chosen for each operator must be close enough. In particular, when the first index is ℓ and the second one is $\ell \pm 1$, the resulting neighbors are common to the three operators. These neighbors have the highest probability to be feasible. So, with a large part of feasible neighbors that are identical, the neutrality rate is very similar for the three operators. On the contrary, for JLE, the neutrality rate can be quite low (around 10% for reverse operator) or quite high (around around 50% for insertion operator).

Generally, the points that are outside the boxplots correspond to small instances (like the one in Table 1 that is part of our instances). The point clouds below the boxplots for JLE/swap and JLE/insertion correspond to the instances of the `swv` set of problems.

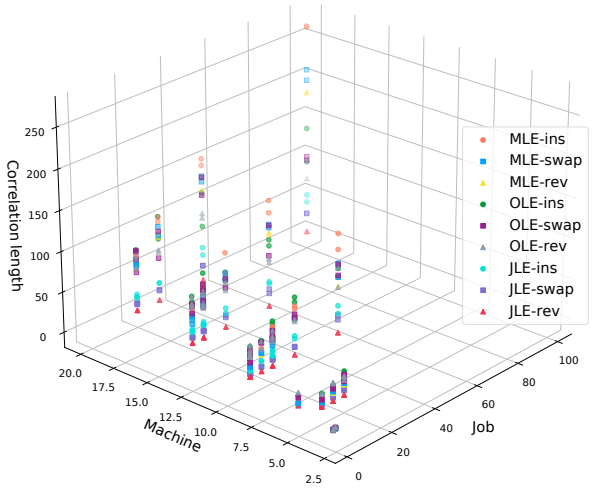

Fig. 7. Correlation length of instances

Table 2. Correlation length ℓ according to jobs and machines: $\ell = \alpha n + \beta m + \gamma$

Enc.-Oper.	α	β	γ	R^2	30×15
JLE-swap	0,244	0,263	1,901	0,875	13.166
JLE-ins	0,386	0,52	3,417	0,882	22.797
JLE-rev	0,001	0,03	1,937	0,217	2.417
OLE-swap	0,687	3,154	-6,868	0,805	61.052
OLE-ins	0,67	3,208	-1,67	0,843	66.55
OLE-rev	0,546	2,484	1,369	0,859	55.009
MLE-swap	1,788	2,72	-28,299	0,975	66.141
MLE-ins	2,395	2,813	-31,908	0,983	82.137
MLE-rev	1,582	2,796	-26,616	0,972	61.784

5.2 Correlation length

In this experiment, we make the same random walks as for neutrality. Then we compute the average correlation length. We plot all the points according to the number of jobs and the number of machines. Fig. 7 shows the correlation length of all the instances according to the number of jobs and the number of machines. Each color corresponds to a pair of encoding scheme and neighborhood operator.

We observe that the points are grouped by color. Again, the encoding scheme and neighborhood operator seem to be a major factor for the correlation length compared to the instance. As the points of a same color seems to be in a plan, we try to make a linear regression in 2D. Table 2 shows the parameters obtained for each pair of encoding scheme and neighborhood operator. We also give the adjusted coefficient of determination (R^2), and the estimated value for an instance of 30 jobs and 15 machines.

We note that job-reverse does not give a good fit. In fact, all the values in this case are around 2. Applying the reverse operator twice on a solution is nearly like generating a random solution with absolutely no correlation with the initial solution.

As for the other results, the linear model is quite good to explain the correlation length. JLE gives shorter correlation length than OLE or MLE. The neighbors in the case of JLE can be quite different but not so different. The landscape remains quite smooth.

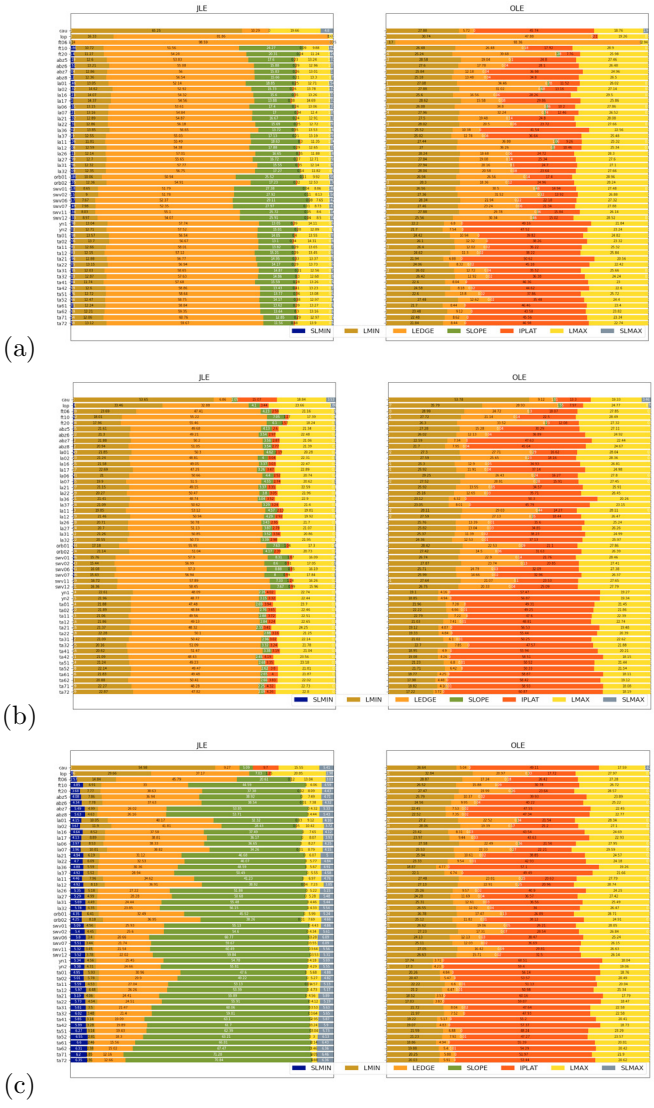

Fig. 8. Position type distributions for (a) swap (b) insertion (c) reverse

Insertion gives smoother landscape than swap and swap gives smoother landscape than reverse. Roughness increases with the number of elements that have been impacted by the operator, regarding their relative order to the other elements. The insert operator impacts a single element, the swap operators impacts two elements and the reverse operators impacts as many elements as the size of the range that is reversed.

5.3 Position type distribution

In this experiment, we select 10000 solutions chosen uniformly among feasible solutions. Then we compute 5 neighbors at random. From these neighbors, we compute the position type of the solution. Fig. 8 shows the distribution for the swap operator, the insertion operator and the reverse operator respectively.

We observe a notable difference between JLE and OLE: the proportion of IPLAT is greater for the OLE and the proportion of LEDGE and SLOPE is more important for JLE. This is a consequence of the higher neutrality observed for the OLE compared to JLE. Again, we observe

that the three distributions for the OLE are quite identical. This is the same explanation as the identical neutrality rate: the neighborhoods are very similar.

The proportion of SLOPE, SLMIN and SLMAX is more important for JLE/reverse that for JLE/swap or JLE/insertion. In this case, the reverse operator can modify a large part of the solution, which leads to a very different schedule and hence to a different fitness. We also observe that the proportion of strict local optima increases with the number of elements that have been impacted by the operator. This observation is in line with the correlation length of these neighborhoods as roughness is a measure of the number of local optima.

6. CONCLUSION

As a conclusion, a fitness landscape is greatly influenced by the encoding scheme associated with a neighborhood operator, more than the instance itself. This work leads us to think that the choice of encoding schemes and neighborhood operators should be done thoroughly before the choice of any metaheuristics, as it could improve dramatically the performance of the metaheuristic.

Several perspectives are open from these first results. In the short term, we aim at applying the same approach to real applications, and first of all to extensions of basic job shop scheduling problems, in order to verify if similar conclusions could be drawn for more constrained problems: flexible job shop scheduling, or/and problems with transport constraints and/or time windows. In other words, we want to study the relationship between the specificities of the problem and the performance of the encodings. Other problems could also be investigated such as vehicle routing problems. In the longer term, we hope that our work will allow us to provide recommendations on the choice of relevant encodings/operators according to the characteristics of the problem studied.

REFERENCES

Bierwirth, C. (1995). A generalized permutation approach to job shop scheduling with genetic algorithms. *Operations-Research-Spektrum*, 17(2), 87–92.

Carlier, J. and Chrétienne, P. (1988). *Problèmes d'ordonnancement : modélisation, complexité, algorithmes*. 1 vol. (326 p.). Masson.

Caumond, A. (2006). *Le problème de jobshop avec contraintes: modélisation et optimisation*. Ph.D. thesis, Université Blaise Pascal - Clermont Ferrand II, https://tel.archives-ouvertes.fr/tel-00713587.

Choi, I.C. and Choi, D.S. (2002). A local search algorithm for jobshop scheduling problems with alternative operations and sequence-dependent setups. *Computers & Industrial Engineering*, 42(1), 43–58. doi:10.1016/S0360-8352(02)00002-5.

Czogalla, J. and Fink, A. (2012). Fitness landscape analysis for the no-wait flow-shop scheduling problem. *Journal of Heuristics*, 18, 25–51. doi:https://doi.org/10.1007/s10732-010-9155-x.

Durasević, M. and Jakobović, D. (2016). Comparison of solution representations for scheduling in the unrelated machines environment. *39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 1336–1342. doi:10.1109/MIPRO.2016.7522347.

Esquirol, P. and Lopez, P. (2008). *Production scheduling*, volume 1, chapter 2, Basic Concepts and Methods in Production Scheduling, 5–32. ISTE Ltd and John Wiley & Sons, Inc., hermès science publications edition.

Fernandez-Viagas, V., Perez-Gonzalez, P., and Framinan, J.M. (2019). Efficiency of the solution representations for the hybrid flow shop scheduling problem with makespan objective. *Computers & Operations Research*, 109, 77–88. doi:10.1016/j.cor.2019.05.002.

French, S. (1982). *Sequencing and scheduling : an introduction to the mathematics of the job-shop*. Chichester, West Sussex : E. Horwood.

Holsapple, C.W., Jacob, V.S., Pakath, R., and Zaveri, J.S. (1993). A genetics-based hybrid scheduler for generating static schedules in flexible manufacturing contexts. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(4), 953–972. doi:10.1109/21.247881.

Jarboui, B., Siarry, P., and Teghem, J. (2013). *Métaheuristiques pour l'ordonnancement monocritère des ateliers de production*. Lavoisier.

Kauffman, S.A. (1993). *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press.

Liu, R., Piplani, R., and Toro, C. (2022). Deep reinforcement learning for dynamic scheduling of a flexible job shop. *International Journal of Production Research*, 1–21. doi:10.1080/00207543.2022.2058432.

Madraki, G., Mousavian, S., and Salmani, Y. (2022). A theoretical framework to accelerate scheduling improvement heuristics using a new longest path algorithm in perturbed DAGs. *International Journal of Production Research*, 0(0), 1–21. doi:10.1080/00207543.2021.2017057.

Malan, K.M. and Engelbrecht, A.P. (2013). A survey of techniques for characterising fitness landscapes and some possible ways forward. *Information Sciences*, 241, 148–163. doi:10.1016/j.ins.2013.04.015.

Mattfeld, D.C., Bierwirth, C., and Kopfer, H. (1999). A search space analysis of the Job Shop Scheduling Problem. *Annals of Operations Research*, 86(0), 441–453. doi:https://doi.org/10.1023/A:1018979424002.

Merkuryeva, G. and Bolshakovs, V. (2020). Benchmark Fitness Landscape Analysis. *International journal of simulation: systems, science & technology*. doi:10.5013/IJSSST.a.12.02.05.

Mihoubi, B., Bouzouia, B., and Gaham, M. (2021). Reactive scheduling approach for solving a realistic flexible job shop scheduling problem. *International Journal of Production Research*, 59(19), 5790–5808. doi:10.1080/00207543.2020.1790686.

Rothlauf, F. and Goldberg, D.E. (2003). Redundant representations in evolutionary computation. *Evolutionary Computation*, 11, 381–415. doi:10.1162/106365603322519288.

Sörensen, K., Servaux, M., and Glover, F. (2018). A history of metaheuristics. *Handbook of heuristics*, 1–18.

Tari, S. (2019). *Stratégies d'exploration de paysages de fitness : application à la résolution approchée de problèmes d'optimisation combinatoire*. Ph.D. thesis, Université d'Angers, HAL, https://tel.archives-ouvertes.fr/tel-02469501.

Vallada, E. and Ruiz, R. (2011). A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *European Journal of Operational Research*, 211(3), 612–622. doi:https://doi.org/10.1016/j.ejor.2011.01.011.

Verel, S. (2016). *Apport à l'analyse des paysages de fitness pour l'optimisation mono-objective et multiobjective*. Ph.D. thesis, Université du Littoral Côte d'Opale, HAL, https://hal.archives-ouvertes.fr/tel-01425127.

Vlašic, I., Durasević, M., and Jakobović, D. (2020). A comparative study of solution representations for the unrelated machines environment. *Computers & Operations Research*, 123, 105005. doi:10.1016/j.cor.2020.105005.

Weinberger, E. (1990). Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics volume*, 63, 325–336. doi:https://doi.org/10.1007/BF00202749.

Zhao, F., Xue, F., Yang, G., Ma, W., Zhang, C., and Song, H. (2019). A Fitness Landscape Analysis for the No-Wait Flow Shop Scheduling Problem With Factorial Representation. *IEEE Access*, 7(18484678), 21032 – 21047. doi:10.1109/ACCESS.2019.2896355.