57º SBPO
SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
Gramado • RS
DE 5 A 9 DE OUTUBRO DE 2025

A general variable neighborhood search approach for the job sequencing and tool switching problem

Laura Cristina Silva e Rocha
Instituto Federal do Sudeste de Minas Gerais
Campus Manhuaçu
lauracsrocha0@gmail.com

Leonardo Cabral da Rocha Soares
Instituto Federal do Sudeste de Minas Gerais
Campus Manhuaçu
leonardo.soares@ifsudestemg.edu.br

ABSTRACT

The *job sequencing and tool switching problem* is a well-known $\mathcal{NP}$-hard optimization problem with significant academic and industrial relevance. Although widely studied, the problem remains challenging, especially for large-scale instances. This paper presents the first application of the general variable neighborhood search metaheuristic for the problem, combining traditional and tailored local searches organized by a randomized variable neighborhood descent method. The approach is evaluated on large-scale benchmark instances, comparing the performance against the best-known solutions. Although not surpassing the state-of-the-art methods on average, the computational results show that the proposed method is competitive, achieving high-quality solutions with a notably low average gap of only 1.51%.

KEYWORDS. Flexible manufacturing. Job sequencing and tool switching problem. General variable neighborhood search.

Paper topics: POI, MH.

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

1. Introduction

Flexible manufacturing systems (FMS) have been studied since the 1960s to enable industries to deal with the constant variations in customers' demand for products [Yadav and Jayswal, 2018]. In this context, flexibility is crucial to providing a manufacturing system that can withstand certain variations in part styles without any interruption in the production line, and flexible machines are required. A flexible machine is a computer numerically controlled device that performs several jobs since the tools demanded were previously loaded in its *magazine* [Zeballos, 2010]. In this paper, a job is a series of tasks performed on raw materials or semi-finished parts like sanding, drilling, and cutting, among others, requiring a specific set of tools.

Although the machine magazine capacity can hold all the tools needed for a specific job, switching tools into the magazine may be necessary to perform a set of jobs sequentially. Tool switching requires stopping machine operations. The required new tools need to be retrieved from storage and transported to the production area. Some tools must be unloaded and returned to the storage for free magazine slots. Once the magazine has enough free slots, the news tools must be loaded, allowing the machine to perform the next job [Soares and Carvalho, 2020].

Production management in an FMS is not trivial. The scheduler must make several decisions to ensure efficient production even when the FMS includes only a single flexible machine, as demonstrated by Stecke [1983]. Decisions that affect the number of tool switches must receive special care since their configuration operations consume the most setup time in an FMS [Van Hop and Nagarur, 2004] and respond to 25% up 30% of its costs [Beezão et al., 2017]. To minimize the total number of tool switches, Tang [1986] introduces the *job sequencing and tool switching problem* (SSP) or, for short, the *tool switching problem*.

The SSP consists of two sub-problems: a *single-machine job scheduling problem* that seeks to determine the best job sequence and a *tooling problem* that aims to find the best tool loading policy for a given fixed job sequence. The first sub-problem is classified as $\mathcal{NP}$-hard by Crama et al. [1994] and drives the overall complexity of the SSP. For any fixed job sequence, the keep tool needed soonest (KTNS) policy, introduced by Tang and Denardo [1988], solves the tooling problem in polynomial time. Since its first approach, the researchers have extensively studied the SSP and its variations due to their academic relevance and broad practical applications across various industrial segments [Calmels, 2018]. In particular, a recent publication of a new benchmark set of challenging instances [Mecler et al., 2021] has drawn increased attention to the problem in operational research and high-performance computing [Soares and Carvalho, 2024; Macedo and Moreno, 2024; Almeida et al., 2025].

In this paper, we present a new approach for solving the SSP. A general variable neighborhood search (GVNS) combines traditional and tailored local search methods. The local searches are organized by a randomized variable neighborhood descent method (RVND), allowing the method to explore different regions of the search space over independent runs. The computational experiments used the benchmark instance set from the literature, and the analysis compares the results with the best-known solutions. Although the approach did not outperform the state-of-the-art, the results demonstrate its competitiveness, offering an alternative for generating high-quality solutions by a simple and effective metaheuristic.

The remainder of this paper follows this structure. The Section 2 formally describes the SSP. Section 3 reviews the existing literature on the SSP, highlighting the leading publications chronologically. Next, Section 4 presents the details of the proposed GVNS-RVND metaheuristic, and the computational results are reported in Section 5. Finally, we provide conclusions and future research directions in Section 6.

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

2. Problem description

Consider a FMS with one single flexible machine containing a magazine with the capacity for C simultaneous tools; a set of jobs $J = \{1, \ldots, n\}$ to be sequentially processed; a set of available tools $T = \{1, \ldots, m\}$; and a subset of tools T_j ($T_j \in T$) required to process a job $j (j \in J)$, where $\mid T_j \mid \leq C$. The SSP consists of determining the schedule of job processes on the machine, such that the number of required tool switches is minimized. Additionally, the problem has the following characteristics: (i) all tools are of identical size and can occupy any free slot in the magazine; (ii) only one tool can be switched at a time and tool switching times are the same for all tools; (iii) the set of jobs and the set of tools needed for each job are known in advance; (iv) the tool switches can only occur between the processing of two sequential jobs; and (v) the tool switches caused by tool wearing or breaking are not considered. Regarding the initial tool loading, i.e, the tools loaded before the first job is processed, there is no standard in the literature. Some authors consider this setup operation tools switches, while others do not. In this work, as Mecler et al. [2021] and Almeida et al. [2025], a tool switch is counted only when one tool is replaced by another. Therefore, the initial tool loading is not included in the solution cost.

An instance for the SSP can be represented by a binary matrix Q, where each row corresponds to a job j, and each column corresponds to a tool t. If job j requires tool t for its processing, then $Q_{jt} = 1$; otherwise, $Q_{jt} = 0$. Table 1 presents a binary matrix for an instance where we consider an example with $n = 5$, $m = 6$ and $C = 4$.

Table 1: An instance of the SSP.

Jobs	Tools					
	1	2	3	4	5	6
1	0	1	0	1	0	1
2	1	1	1	0	1	0
3	0	0	1	0	1	1
4	1	1	0	1	0	1
5	1	1	1	0	1	0

Table 2: Example of a tool switching plan for $\pi = \{2, 5, 1, 3, 4\}$.

Tools	Jobs				
	2	5	1	3	4
1	$\underline{1}$	1	$\underline{0}$	0	$\underline{1}$
2	$\underline{1}$	1	1	$\underline{0}$	$\underline{1}$
3	$\underline{1}$	1	$\underline{0}$	$\underline{1}$	$\underline{0}$
4	0	0	$\underline{1}$	0	1
5	$\underline{1}$	1	0	1	$\underline{0}$
6	0	0	$\underline{1}$	1	1

A solution to the SSP is a permutation π of the columns of the *tool switching plan*. This permutation dictates the order in which the jobs are processed. The tool switching plan can be represented by a binary matrix $R^{\pi} = \{R_{tj}\}$, where $t \in T$, and $j \in J$. The column of the tool switching plan represents the magazine's state during the processing of a specific job. Thus, an element $R_{tj} = 1$ indicates the presence of the tool in the magazine, while $R_{tj} = 0$ indicates its absence.

Table 2 presents the tool switching plan to instance introduced in Table 1 with $\pi = \{2, 5, 1, 3, 4\}$. The tools removed or inserted immediately before the processing of a job are underlined. The first job to be processed, job 2, requires only three tools. Given the magazine capacity ($C = 4$), additional tools needed for upcoming jobs can be loaded in advance. Thus, before processing job 2, the magazine was loaded with tools $\{1, 2, 3, 5\}$. Tool $\{1\}$ was included because it is needed for job 5. This initial tool loading is not considered in the solution cost, i.e., there are not accounted for as tool switches. This initial setup contains all the tools needed to process the jobs 2 and 5. Before processing job 1, it was necessary to insert tools $\{4, 6\}$, which required removing two tools from the magazine. According to the KTNS policy, among the candidate tools for removal (those not required by the current job), those that will be needed the latest are selected. In this example, the tools $\{1, 3\}$ were removed, generating two tool switches before job 1. For job 3, only

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

the tool $\{3\}$ needed to be inserted, replacing tool $\{2\}$, adding one more tool switch. Finally, before processing job 4, tools $\{2, 1\}$ were inserted, while $\{3, 5\}$ were removed, generating two more tool switches. For the Table 1 instance, five tool switches are necessary to process all jobs in order $\pi = \{2, 5, 1, 3, 4\}$.

The evaluation of the tool switching plan is conducted by applying Equation (1), as proposed by Crama et al. [1994]. This formulation quantifies the frequency with which a position R_{tj} transitions from a value of zero to one, indicating the insertions of new tools into the magazine.

$$Z_{SSP}(R^{\pi}) = \sum_{t \in T} \sum_{j \in J} r_{tj}^{\pi}(1 - r_{tj-1}^{\pi}) \tag{1}$$

The objective of the SSP is to determine a job sequence permutation $\pi \in \Pi$ that minimizes the total number of tool switches. The objective function of the SSP is expressed by Equation (2), where Π represents the set of all possible job permutations.

$$\min_{\pi \in \Pi} \left\{ Z_{SSP}^{\pi}(R) \right\} \tag{2}$$

3. Related works

Since its seminal publication by Tang and Denardo [1988], numerous authors have addressed the SSP and its variations. Calmels [2018] provides an extensive systematic review of the main variations of the problem. The following presents the leading publications for the original version of the SSP (also called Uniform SSP), organized chronologically.

As previously mentioned, the SSP was formally introduced by Tang and Denardo [1988], where the problem is modeled and solved as an instance of the Traveling Salesman Problem (TSP). Furthermore, they also developed the KTNS policy, which can solve the tooling problem optimally in polynomial time for a given fixed sequence of jobs to be processed. Later, Bard [1988] addressed the SSP by formulating it as a non-linear integer program and proposed a heuristic based on Lagrangian relaxation. His approach yielded high-quality solutions within a significantly reduced computational time.

The NP-hardness of the SSP for instances where $C \geq 2$ was proven by Crama et al. [1994]. Once again, the problem was modeled as an instance of the TSP. However, despite this approach being recurrent in the literature [Ahmadi et al., 2018; Djellab et al., 2000; Hertz et al., 1998; Laporte et al., 2004; Shirazi and Frizelle, 2001], Yanasse and Lamosa [2006], in a theoretical work, highlight the inefficiency of this modeling, since the SSP and TSP do not possess direct equivalence, except for the special case in which each job requires exactly C tools.

Laporte et al. [2004] proposed a new integer linear programming (ILP) model which showed improved linear relaxation results compared to the model by Tang and Denardo [1988]. Furthermore, a branch-and-bound and a branch-and-cut algorithm were presented and compared using instances generated by the authors. The former achieved optimal solutions on instances with up to nine jobs, and the latter with up to 25.

Yanasse [2009] presented an enumerative algorithm which outperformed the results obtained by Laporte et al. [2004] and proposed a new set of instances for the SSP. Senne and Yanasse [2009] presented three implementations of the enumeration-based beam search. However, the proposed methods were not compared with other algorithms from the literature.

Chaves et al. [2016] presented a hybrid algorithm that combines a biased random-key genetic algorithm (BRKGA) with a clustering search (CS) algorithm. Experiments were conducted using instances from Yanasse [2009] and Crama et al. [1994] and reported better results for all tested instances.

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

A new graph representation, a constructive heuristic, and a new implementation of the iterated local search (ILS) metaheuristic were proposed for the SSP by Paiva and Carvalho [2017]. Moreover, among the local searches used by the ILS, the authors explored the connection between the SSP and the consecutive block minimization problem (CBM). The computational experiments used the instances from Yanasse [2009], Crama et al. [1994], and Catanzaro et al. [2015] and obtained equal or better results for all considered problems.

Ahmadi et al. [2018] presented a q-learning genetic algorithm along with the modeling of the SSP as a TSP of second order (2-TSP) instance. However, they did not detail the metric for determining distances in the modeling. The experiments considered only a subset of the available instances [Catanzaro et al., 2015; Crama et al., 1994] and showed marginal improvements compared to Paiva and Carvalho [2017].

Despite not being able to solve the larger instances from Yanasse [2009] and Catanzaro et al. [2015] within the established one-hour time limit, the formulation of the SSP as an integer linear problem using a multi-commodity flow model presented by Silva et al. [2020] surpasses the results obtained by previous models, both in the number of instances solved at optimality, and in average running time.

Mecler et al. [2021] presented a hybrid genetic search (HGS) algorithm and introduced a new set of larger instances with up to 70 jobs and 105 tools. The results showed that the proposed method provided better average results from the new instances and for the instances available in the literature [Catanzaro et al., 2015; Crama et al., 1994; Yanasse, 2009], compared to the methods presented by Chaves et al. [2016], Paiva and Carvalho [2017], and Ahmadi et al. [2018].

Soares [2023] proposed a new implementation of the ILS and BRKGA metaheuristics. Again, the connection between the SSP and CBM was considered in the SSP approach. The results were compared to those reported by Mecler et al. [2021]. Although the computational time was significantly lower, the quality of the solutions did not surpass the results of the previous work.

Macedo and Moreno [2024] applied the random keys concept to the ILS metaheuristic and used a constructive heuristic for initial solution generation. Experiments were conducted using the moderate-sized instances proposed by Macedo and Bueno [2023], and demonstrated a nearly 90% reduction in computational time compared to the conventional ILS implemented by Macedo and Bueno [2023]. Unfortunately, the method was not compared with the benchmark instances of SSP.

Recently, Almeida et al. [2025] presented a new parallel tempering (PT) implementation and applied it to the SSP as one of several case studies. The experiments considered only the instances proposed by Mecler et al. [2021]. Better or equal solutions values were reported for 7 of the 12 instance subsets. However, only marginal differences were reported, with percentage differences ranging from -0.14 to 0.11. Additionally, the authors presented a parallel implementation of the greedy pipe construction algorithm (GPCA), introduced by Cherniavskii and Goldengorin [2022]. This method, as the KTNS policy, given a fixed sequence of job solves the tooling problem in polynomial time; however, GPCA has an amortized complexity bounded by $\mathcal{O}(Cn)$, which is smaller than the $\mathcal{O}(mn)$ complexity of KTNS.

From the available instances for the SSP in the literature, only the largest set introduced by Mecler et al. [2021] continue to represent a challenge, given to the high quality of current solution methods. For this reason, the computational experiments detailed in Section 5 focus exclusively on this group of instances. Since no single method has proven uniformly superior for solving these problems, we use the best-known solution values, from the studies by Mecler et al. [2021] and Almeida et al. [2025], as a benchmark to evaluate the performance of the proposed method.

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

4. Methods

To approach the SSP, we present a new implementation of the metaheuristic GVNS [Mladenović and Hansen, 1997] with three local search procedures organized via an RVND [Subramanian et al., 2010]. Although the literature presents several successful applications of the GVNS and its variations to numerous optimization problems, including scheduling problems related to SSP, to the best of our knowledge, this is the first study that applies GVNS to the SSP. In the following sections, we describe the proposed implementation in detail.

4.1. General variable neighborhood search

Introduced by Mladenović and Hansen [1997], the variable neighborhood search (VNS) is a metaheuristic that, from an initial solution, explores the solution space by a systematic change of neighborhood both in an intensification step to find a local optimum and in a perturbation step to escape from local optima. A GVNS is a VNS extension that uses a variable neighborhood descent method (VND) or an RVND to organize the local searches in the intensification step. The proposed VNS approach to SSP is described next.

We considered two TSP heuristics to generate the GVNS initial solution. The first one, the *nearest neighbor algorithm*, from a given initial job chooses the next job to be scheduled with the smallest difference (distance), considering the tool requirements. We consider all jobs to be initial job. We initially considered the Hamming distance when calculating the matrix of distances. However, this metric did not report good results since it does not consider the magazine's history. Since $C \geq |T_j|$, it may have been possible to load additional tools in the magazine earlier. If the next job requires any of these tools, this will decrease the effective distance between the jobs. Thus, the distance matrix was dynamically constructed from the number of tool switches reported by KTNS, considering the partial scheduling on the current heuristic iteration. Draws are solved randomly. Both heuristics apply the same strategies for constructing the distance matrix and resolving ties.

The second heuristic, the *Bellmore and Nemhauser algorithm* [Bellmore and Nemhauser, 1968], follows a similar strategy. However, it considers both ends of the partial schedule. All possible permutation of two jobs are considered as initial schedule. Next, we calculate the distance of all jobs not yet allocated to both ends of the partial schedule. To each iteration, the heuristic allocates the job with the smallest distance to the corresponding end. Both heuristics are running sequentially. GVNS utilizes the solution with the best evaluation value as the initial solution.

In the intensification step, we apply the local search methods — largest regret, 3-opt, and or-opt — organized through an RVND. As an extension of VND [Mladenović and Hansen, 1997], the RVND systematically changes neighborhoods to explore the solution space. The exploration starts with the first neighborhood and continues until it finds a local optimum. Then, the method proceeds to the next neighborhood. Whenever a new best solution is found, exploration restarts from the first neighborhood. The method ends when the exploration finds a local optimum common to all neighborhoods. RVND differs from VND only in the order of the neighborhoods. While VND uses a fixed order, RVND orders the neighborhoods randomly each time the method is executed.

Regarding the local searches, the *largest regret method* ranks each job according to the number of tool switches generated by its scheduling in its current position. Subsequently, this local search repositions each job in the position where it generates the fewest tool switches. The algorithm reallocates jobs following the order established by the ranking, ensuring it repositions first the jobs that caused the most tool switches. Each movement is evaluated, and the local search ends if the solution is improved. All three local searches use this first improvement strategy.

When applied to SSP, the traditional *3-opt heuristic* breaks the solution into three points at each move, generating four distinct segments that can be reordered into seven new solutions. Although it is computationally expensive, given its neighborhood bounded by $\mathcal{O}(n^3)$, this local

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

search allows a comprehensive search space exploration and helps reach high-quality local optima. The third local search, the *or-opt algorithm*, breaks the solution into three segments and relocates them, keeping the internal order of segments. However, the movements performed by this local search are reachable by the 3-opt method, given the first improvement strategy and the random order used by RVND, not using this local search harms the final solution quality, as noted in the preliminary experiments.

In the perturbation step, we use two different methods. We use a movement from the *double bridge method* until the method reaches 50% of the method's stopping criterion. When applied to the SSP, this method splits the job scheduling into four points and reorganizes them. Once the method exceeds 50% of the stopping criterion, the perturbation is made by randomly reallocating a percentage of the job. This percentage starts at 1% and increases linearly up to 7% in the last method iteration.

The stopping criterion utilized was the number of iterations without solution improvement. This value was defined in the preliminary experiments and set as $\lceil n \times 2 \rceil$, considering a tradeoff between solution quality and running time. Additionally, a time-based stopping trigger was incorporated to prevent excessively long running times, terminating the algorithm after 14,400 seconds. The Algorithm 1 presents the steps of the proposed method. The algorithm receives an initial solution s_0 as an input parameter. It then improves this solution using the RVND method and sets it as the current solution s (line 1). The main loop (lines 2 to 6) applies the method repeatedly until the stopping criterion is satisfied. The algorithm perturbs the current solution according to the percentage p of the stopping criterion (line 3). RVND improves the new solution (line 4). If the new solution (s'') has a better evaluation value than the current solution (s), this new solution becomes the current solution (lines 5 and 6). After achieving the stopping criteria, the method ends and s is returned (line 7).

Algorithm 1: General variable neighborhood search

```
   input : initial solution s_0
1  s ← RVND(s_0);
2  while stopping criterion is not meet do
3  |   s' ← SHAKE(s, p);
4  |   s'' ← RVND(s');
5  |   if evaluation function (s'') < evaluation function (s) then
6  |   |   s ← s'';
7  return s;
```

Algorithm 2 presents the steps of the RVND. An initial solution s is given as an input parameter. The order of neighborhood exploration is randomized by an auxiliary variable (lines 1 and 2). A control variable k controls the systematic change in neighborhood structures (line 3). The main loop (lines 4 to 15) ensures the exploration of neighborhoods until a local optimum common to all neighborhoods is reached. Neighborhoods are explored according to the defined random order (lines 5 to 10), one per iteration. Whenever a neighboring solution with a better solution value is found, solution s is updated and exploration restarts with the first neighborhood (lines 11 to 13). If the solution is not improved in the current neighborhood, it moves to the next neighborhood (lines 14 and 15). When exploring all neighborhoods does not improve the current solution, the method ends, and the current solution is returned (line 16).

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

Algorithm 2: Random variable neighborhood descent

input : solution s
$order \leftarrow [1, 2, 3]$;
shuffle$(order)$;
$k \leftarrow 1$;
while $k \neq 4$ **do**
  if $k = order[1]$ **then**
    $s' \leftarrow$ *largest regret*(s);
  if $k = order[2]$ **then**
    $s' \leftarrow$ *3-opt*(s);
  if $k = order[3]$ **then**
    $s' \leftarrow$ *or-opt*(s);
  if *evaluation function* $(s') <$ *evaluation function* (s) **then**
    $s \leftarrow s'$;
    $k \leftarrow 1$;
  else
    $k \leftarrow k + 1$;
return s;

5. Computational experiments

We conducted a series of computational experiments to measure the quality of the proposed method's solutions and evaluate its behavior. The experiments were run on a computer with a 3.49 GHz *Apple M2* processor, 8 GB of RAM, and the *macOS Sonoma* operating system. We implemented all methods in C++ and compiled them using GCC 15.0.0 with the optimization flags `-O3` and `-march=native`.

The SSP has four benchmark sets of instances available in the literature. However, the classic sets ([Crama et al., 1994], [Yanasse, 2009], [Catanzaro et al., 2015]) do not present challenges compared to the high quality of the state-of-the-art methods to SSP. Only the benchmark set proposed by [Mecler et al., 2021], here named *SSP-I*, remains challenging, presenting the most significant problems in the SSP literature. This set consists of 60 instances and has problems with up to 70 jobs, 105 tools, and a magazine with a capacity for holding up to 55 simultaneous tools. Therefore, we consider only this set in the experiments reported next. As mentioned in Section 3 , the most widely recognized benchmark results were published by [Mecler et al., 2021] and [Almeida et al., 2025]. Since neither method consistently outperforms the other, we used the best-performing results from either studies (HGS/PT) to evaluate the proposed method.

Table 3 presents, for the SSP-I set, the best known solution values and the results obtained by the proposed method across 10 individual runs. Each row in the table shows the average values for five different instances, grouped by the number of jobs (n), available tools (m), and magazine capacity (C). Regarding the benchmark values, we report the best average solution values (S^*) and the average solution values (S). For the proposed method, in addition we present the average initial solution values (I), the average running time (T) in seconds, the average standard deviation, and the percentage distance ($gap(\%)$) calculated between the best known solution value and the best solution value reported by the GVNS-RVND method. The gap is computed as $100 \times \frac{\text{GVNS-RVND}(S^*) - \text{HGS/PT}(S^*)}{\text{HGS/PT}(S^*)}$. Bold values identify the best results for each group of instances.

Although the proposed method did not report average improvements compared to the best known values, the reported results were competitive for all instance subsets. Considering all instances, the average gap of only 1.51%, ranging between average gaps of 0.37% and 3.35%, con-

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

Table 3: Results for the SSP-I set of instances.

			HGS/PT		GVNS-RVND					
n	m	C	S^*	S	I	S^*	S	T	σ	$gap(\%)$
50	75	25	**268.40**	268.56	323.16	269.40	271.70	2456.28	1.54	0.37
50	75	30	**196.20**	196.40	253.46	198.60	201.70	2259.91	2.23	1.12
50	75	35	**147.20**	147.38	205.38	150.60	152.62	2167.16	1.89	2.31
50	75	40	**109.80**	109.92	169.02	112.80	115.36	1972.68	1.91	2.73
60	90	35	**414.60**	415.60	502.78	417.80	421.12	11811.09	2.08	0.77
60	90	40	**319.60**	320.88	403.46	322.40	326.36	9244.71	2.51	0.88
60	90	45	**247.20**	248.70	331.24	251.20	255.44	9309.28	2.89	1.62
60	90	50	**191.00**	191.94	275.10	197.40	200.12	8144.27	2.38	3.35
70	105	40	**576.60**	577.44	681.92	580.20	584.60	14473.70	3.07	0.62
70	105	45	**459.00**	459.90	563.22	463.60	468.72	14477.81	3.29	1.00
70	105	50	**369.60**	371.08	472.86	374.80	380.64	14478.13	3.57	1.41
70	105	55	**298.60**	299.90	400.40	304.20	309.84	14464.22	3.36	1.88

firms the high quality of the presented solutions. The additional time-based stopping criterion was activated only for instances with 70 jobs. Nevertheless, the average gap observed for these instances was just 1.23%, suggesting that the time limit did not negatively affect solution quality. The average standard deviation reported for this set was only 2.56, indicating the ability of the method to generate independent high-quality solutions with low variation. Among the heuristics used to generate the initial solutions, the nearest neighbor heuristic reported the best result in 53.00% of the runs, while the Bellmore and Nemhauser algorithm was used in the remaining 47.00%. Regarding the local search methods, the one contributing most to the final solution quality was or-opt, accounting for 54.63% of the improvements. Next, we have the largest regret, and 3-opt methods, responsible for 26.34% and 19.02% of the improvements, respectively.

A statistical analysis was performed to compare the two methods. The best average values are considered as individual inputs. The Shapiro–Wilk normality test [Shapiro and Wilk, 1965] confirmed the null hypothesis that the results compared from GVNS ($W = 0.96835$, *p-value* = 0.8928) and HGS/PT ($W = 0.96869$, *p-value* = 0.8966) could be modeled according to a normal distribution, with a confidence interval of 95%. The parametric *Student's t-test* [Student, 1908] was applied to verify whether there is a significant difference between the compared results. The test revealed a significant difference between the results of the methods, with HGS/PT having the best mean values ($t = -8.7205, df = 11, p = 0.000001426$) at a significance level of 0.05.

6. Conclusion and future works

In this work, we presented for the first time an application of the general variable neighborhood search metaheuristic to the job sequencing and tool switching problem, a well-known $\mathcal{NP}$-hard problem with significant academic and practical relevance. The proposed method was evaluated against state-of-the-art approaches using a widely adopted benchmark set of instances. Although no new best-known results were obtained, the method demonstrated competitive performance, reporting high-quality solutions with an average gap of only 1.51%. Considering that the current state-of-the-art methods rely on strategies that explore multiple solutions simultaneously, a trajectory-based approach – operating on a single solution at a time – was not expected to yield substantial improvements. Therefore, the studies directly derived from this work will focus on developing a parallelized version of the proposed method, allowing the simultaneous exploration of multiple solutions and aiming to achieve the new best results.

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

7. Acknowledges

This work was supported by the Minas Gerais Research Support Foundation (Fundação de Amparo à Pesquisa do Estado de Minas Gerais – FAPEMIG), the National Council for Scientific and Technological Development (Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq), and the Federal Institute of Education, Science and Technology of the Southeast of Minas Gerais (Instituto Federal de Educação, Ciência e Tecnologia do Sudeste de Minas Gerais – IF Sudeste MG).

References

Ahmadi, E., Goldengorin, B., Suer, G. A., and Mosadegh, H. (2018). A hybrid method of 2-tsp and novel learning-based ga for job sequencing and tool switching problem. *Applied Soft Computing*, 65:214 – 229. ISSN 1568-4946.

Almeida, A. L. B., de Castro Lima, J., and Carvalho, M. A. M. (2025). Revisiting the parallel tempering algorithm: High-performance computing and applications in operations research. *Computers & Operations Research*, p. 107000.

Bard, J. F. (1988). A Heuristic for Minimizing the Number of Tool Switches on a Flexible Machine. *IIE Transactions*, 20(4):382–391. ISSN 0740-817X. URL `https://doi.org/10.1080/07408178808966195`. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/07408178808966195.

Beezão, A. C., Cordeau, J.-F., Laporte, G., and Yanasse, H. H. (2017). Scheduling identical parallel machines with tooling constraints. *European Journal of Operational Research*, 257(3):834–844.

Bellmore, M. and Nemhauser, G. L. (1968). The traveling salesman problem: a survey. *Operations Research*, 16(3):538–558.

Calmels, D. (2018). The job sequencing and tool switching problem: state-of-the-art literature review, classification, and trends. *International Journal of Production Research*, p. 1–21.

Catanzaro, D., Gouveia, L., and Labbé, M. (2015). Improved integer linear programming formulations for the job sequencing and tool switching problem. *European Journal of Operational Research*, 244(3):766 – 777. ISSN 0377-2217.

Chaves, A. A., Lorena, L. A. N., Senne, E. L. F., and Resende, M. G. C. (2016). Hybrid method with CS and BRKGA applied to the minimization of tool switches problem. *Computers & Operations Research*, 67:174–183. ISSN 0305-0548.

Cherniavskii, M. and Goldengorin, B. (2022). An almost linear time complexity algorithm for the tool loading problem. *arXiv preprint arXiv:2207.02004*.

Crama, Y., Kolen, A. W. J., Oerlemans, A. G., and Spieksma, F. C. R. (1994). Minimizing the number of tool switches on a flexible machine. *International Journal of Flexible Manufacturing Systems*, 6(1):33–54. ISSN 0920-6299, 1572-9370.

Djellab, H., Djellab, K., and Gourgand, M. (2000). A new heuristic based on a hypergraph representation for the tool switching problem. *International Journal of Production Economics*, 64(1): 165–176.

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

Hertz, A., Laporte, G., Mittaz, M., and Stecke, K. E. (1998). Heuristics for minimizing tool switches when scheduling part types on a flexible machine. *IIE Transactions*, 30(8):689–694. ISSN 0740-817X.

Laporte, G., Salazar-Gonzalez, J. J., and Semet, F. (2004). Exact algorithms for the job sequencing and tool switching problem. *IIE Transactions*, 36(1):37–45. ISSN 0740-817X.

Macedo, H. G. and Bueno, L. F. (2023). Estudo comparativo de diferentes métodos de solução para o problema de minimização de trocas de ferramentas. In *Anais do LVI SBPO*, São José dos Campos. Sobrapo.

Macedo, H. G. and Moreno, C. F. (2024). Problema de minimização de trocas de ferramentas com uma busca local iterada baseada em chaves aleatórias. In *Anais do LVI SBPO*, Fortaleza. Sobrapo.

Mecler, J., Subramanian, A., and Vidal, T. (2021). A simple and effective hybrid genetic search for the job sequencing and tool switching problem. *Computers & Operations Research*, 127:105153.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Computers & operations research*, 24(11):1097–1100.

Paiva, G. S. and Carvalho, M. A. M. (2017). Improved heuristic algorithms for the job sequencing and tool switching problem. *Computers and Operations Research*, 88:208 – 219. ISSN 0305-0548. URL http://www.sciencedirect.com/science/article/pii/S0305054817301971.

Senne, E. L. F. and Yanasse, H. H. (2009). Beam search algorithms for minimizing tool switches on a flexible manufacturing system. *Proceedings of The 11th Wseas International Conference on Mathematical and Computational Methods In Science and Engineering (MACMESE '09)*, p. 68–72.

Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2333709.

Shirazi, R. and Frizelle, G. D. M. (2001). Minimizing the number of tool switches on a flexible machine: an empirical study. *International Journal of Production Research*, 39(15):3547–3560. ISSN 0020-7543.

Silva, T. T. d., Chaves, A. A., and Yanasse, H. H. (2020). A new multicommodity flow model for the job sequencing and tool switching problem. *International Journal of Production Research*, p. 1–16.

Soares, L. C. R. and Carvalho, M. A. M. (2024). Uma nova abordagem rápida e competitiva para o problema de minimização de trocas de ferramentas. In *Anais do LVI SBPO*, Fortaleza. Sobrapo.

Soares, L. C. R. and Carvalho, M. A. M. (2020). Biased random-key genetic algorithm for scheduling identical parallel machines with tooling constraints. *European Journal of Operational Research*, 285(3):955–964.

Soares, L. C. (2023). *Otimização de processos produtivos em sistemas de manufatura flexível*. PhD thesis, Universidade Federal de Ouro Preto. Available on https://www.repositorio.ufop.br/items/17c5dad2-2648-4b78-a60a-16cd43b51be5.

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br

Stecke, K. E. (1983). Formulation and solution of nonlinear integer production planning problems for flexible manufacturing systems. *Management Science*, 29(3):273–288.

Student (1908). The probable error of a mean. *Biometrika*, 6(1):1–25. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2331554.

Subramanian, A., Drummond, L. M., Bentes, C., Ochi, L. S., and Farias, R. (2010). A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 37(11):1899–1911.

Tang, C. (1986). A job scheduling model for a flexible manufacturing machine. In *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, volume 3, p. 152–155. IEEE.

Tang, C. S. and Denardo, E. V. (1988). Models arising from a flexible manufacturing machine, part I: Minimization of the number of tool switches. *Operations Research*, 36(5):767–777. ISSN 0030-364X.

Van Hop, N. and Nagarur, N. N. (2004). The scheduling problem of PCBs for multiple non-identical parallel machines. *European Journal of Operational Research*, 158(3):577–594.

Yadav, A. and Jayswal, S. (2018). Modelling of flexible manufacturing system: a review. *International Journal of Production Research*, 56(7):2464–2487.

Yanasse, H. H. (2009). Um novo limitante inferior para o problema de minimização de trocas de ferramentas. *Anais do XLI Simpósio Brasileiro de Pesquisa Operacional*, p. 2841–2848.

Yanasse, H. H. and Lamosa, M. J. P. (2006). On solving the minimization of tool switches problem using graphs. In *XII International Conference on Industrial Engineering and Operations Management*, p. 1–9, Fortaleza, Brazil.

Zeballos, L. (2010). A constraint programming approach to tool allocation and production scheduling in flexible manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, 26(6): 725–743.

DOI: 10.59254/sbpo-2025-212460
https://proceedings.science/p/212460?lang=pt-br