




# An Investigation of Generic Approaches to Large Neighbourhood Search

Filipe Souza   

Insight SFI Research Centre for Data Analytics, National University of Ireland Galway, Ireland  
SFI Centre for Research Training in Artificial Intelligence, Cork, Ireland  
School of Computer Science & IT, University College Cork, Ireland

Diarmuid Grimes  

Munster Technological University, Cork, Ireland  
SFI Centre for Research Training in Artificial Intelligence, Cork, Ireland

Barry O’Sullivan  

Insight SFI Research Centre for Data Analytics, National University of Ireland Galway, Ireland  
SFI Centre for Research Training in Artificial Intelligence, Cork, Ireland  
School of Computer Science & IT, University College Cork, Ireland

---

## Abstract

A bottleneck in the more wide-spread use of approaches such as Large Neighborhood Search is the need for domain-specific knowledge. To this end, a number of generic LNS methods have previously been proposed that automate the selection of variables in the neighborhood with the aim of reducing the expertise requirement. Recently a new generic approach, Improved Variable-Relationship Guided LNS (iVRG), was proposed that showed promising initial results. This method combines static information regarding problem structure and dynamic information from search performance in its neighborhood selection.

In this work, we first show the generalisability of the approach by comparing it on two widely studied problems, car sequencing and steel mill slab, where it outperformed existing generic approaches. We then provide a detailed examination of iVRG, investigating its key components (static/dynamic information, the use of a Tournament Selection operator) to assess their individual impact and provide insight into iVRGs overall behavior.

**2012 ACM Subject Classification** Computing methodologies → Heuristic function construction

**Keywords and phrases** Combinatorial Optimization, Metaheuristics, Large Neighborhood Search (LNS), Machine Reassignment Problem, Car Sequencing Problem, Steel Mill Slab Problem

**Digital Object Identifier** 10.4230/LIPIcs.CP.2024.39

**Category** Short Paper

**Supplementary Material** *Software (Source Code)*: <https://github.com/filipesouza/it/iVRG-LNS>

**Funding** Supported by SFI Centre for Research Training in Artificial Intelligence under Grant No. 18/CRT/6223 and SFI under Grant No. 12/RC/2289-P2, co-funded under the European Regional Development Fund.

## 1 Introduction

Large neighborhood search [15] is a metaheuristic approach that works by iteratively improving an initial solution through optimising subsets of variables. Each iteration involves the selection and relaxation (unassignment) of a neighborhood of variables (*destroy* phase). The problem is then optimised (*repair* phase) with the neighborhood restricted to only those variables that can be searched over, all other variables are fixed to their values in the current solution. This has led to significant advances in terms of problem size that can be handled by such optimisation algorithms, enabling them to explore vast solution spaces efficiently.



© Filipe Souza, Diarmuid Grimes, and Barry O’Sullivan;  
licensed under Creative Commons License CC-BY 4.0

30th International Conference on Principles and Practice of Constraint Programming (CP 2024).

Editor: Paul Shaw; Article No. 39; pp. 39:1–39:10

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

There are a number of components that must be chosen for a given implementation of the basic LNS framework: the method to select the variables in a given neighborhood, the size of the neighborhood, and the solution approach (typically an exact method such as CP/MIP). However, the more general applicability of LNS to a wide range of problem domains remains a challenge as it often requires domain-specific insights to effectively design the neighborhood selection mechanism. To address this limitation a number of domain-independent neighbourhood selection approaches have been proposed down through the years [12, 7, 6, 10]. These approaches involve strategies that can adapt to various problem domains without relying on domain-specific knowledge.

Despite the importance of domain-independent neighbourhood selection approaches recent research in the area of generic neighborhood selection operators has been relatively scarce with much of the focus on *adaptive LNS* approaches [13, 5, 20]. These approaches use a portfolio of heuristics with the system adapting weights based on search performance to decide on the probability of selecting a given heuristic in the next iteration. The quality of these approaches is sensitive to the diversification of heuristics included in the portfolio. More recently there has also been a focus on machine learning approaches for learning neighborhood selection, albeit primarily restricted to mixed integer programming (MIP) [10, 22].

In this paper, we present an analysis of a recently proposed domain-independent neighbourhood selection approach, Improved Variable-Relationship Guided LNS (iVRG) [19]. This approach showed promising results, albeit only evaluated on Google’s Machine Reassignment Problem (MRP). We delve into its key components, including the use of structural relationships, search state information, and a tournament selection mechanism.

To provide a robust comparative analysis, we empirically compare iVRG with two state-of-the-art generic neighbourhood selection approaches: Propagation Guided LNS (PG-LNS) [12] and Cost Impact Guided LNS [6]. The former was originally evaluated using the Car Sequencing Problem (CSP), while the latter was tested on the Steel Mill Slab Problem (SMSP). We extend our previous analysis [19] to these two problem domains to assess the generalisability of iVRG. We then investigate the impact of the different components of the iVRG approach, the use of structural information, the use of search state information, and finally the use of tournament selection within the heuristic.

## 2 Related Work

### 2.1 Propagation Guided Large Neighbourhood Search

Perron et al. [12] proposed using propagation information to identify strongly connected neighbourhoods. To generate this information the basic PG-LNS approach starts from all variables unassigned and initially chooses a variable at random. It then repeatedly chooses a variable to assign until a predefined neighborhood size is reached. The variable to assign is chosen randomly from the top ten variables ranked according to their domain reduction after assigning the previously selected variable. If this list is empty, selection reverts to a random choice from the remaining relaxed variables.

A complementary approach, Reverse PG-LNS, was also proposed. This starts from all variables being assigned and repeatedly unassigns a variable until the desired neighborhood size is achieved. The variable to unassign is chosen randomly from a list of the ten variables with the highest *closeness* score accumulated from the previous selected variables. This score is based on the impacts computed in PG-LNS.

## 2.2 Cost-Impact Guided LNS

Cost-Impact Guided Large Neighbourhood Search (CIG-LNS) [6] selects variables for relaxation based on their impact on the cost (objective function). This cost impact is determined by observing the variations in the lower bound that occur when each variable is assigned a value. These variations are captured through *dives*, where a dive is the re-application of the current solution in a rearranged order. An additional parameter  $\alpha$  is incorporated into each variable’s score to control the level of diversification. The parameter value is in the range  $[0, 1]$ , with 0 equating to pure random selection and 1 being that the cost alone is used.

## 3 Improved Variable-Relationship Guided LNS

*Improved Variable-Relationship Guided LNS* [19] is a neighborhood selection operator that uses the structural relationships between variables to guide search towards *connected* neighborhoods. It combines this static information with dynamic information from search to prioritize variables with a higher likelihood of enhancing the solution, and uses tournament selection to only consider a subset of variables for selecting the next variable for the neighborhood. The latter serves to both increase diversification, and to reduce computational effort (since heuristic information is only computed for the subset of variables).

### 3.1 Structural Relationship

The structural relationship is incorporated in neighborhood selection by considering only variables linked to the most recently relaxed variable. Specifically, the probability of selecting a variable  $j$  subsequent to the relaxation of variable  $i$  is determined by the following formula:

$$\frac{1}{|C_i|} * \sum_{c \in C_{i,j}} \frac{1}{|V_c|}$$

where  $C_{i,j}$  denotes the set of constraints involving both variables  $j$  and  $i$ , while  $|V_c|$  is the arity of constraint  $c$ , and  $|C_i|$  is the number of constraints involving variable  $i$ . The logic of using constraint arity is that the relevance of the relationship between any two variables due to a constraint diminishes as the number of variables sharing that constraint increases. At its most extreme, when a constraint includes all variables, it fails to offer meaningful insight into the strengths of the relationships between those variables. Note since these are static values, this need only be computed once at the start of an LNS run on an instance.

However, there are also problems where variables without such a direct relationship can still contribute greatly to solution improvement when selected together (e.g. swapping their bins in a bin packing problem). For these problems, forcing all neighborhood variables to be connected may impede performance. Therefore, iVRG uses the combination of structural relationship and search state information (SSI) for half of the neighborhood variables, and otherwise only uses SSI.

### 3.2 Search State Information

There are a number of different forms of search state information (SSI) that could be used, e.g. variables that are in conflict during search, variables whose selection resulted in large improvements during search, etc. In previous work on iVRG [19], two aspects of SSI were combined. The first is the *Variable Cost*, that centered on the principle that the most impactful neighborhood selections involve high-cost variables. This heuristic measures a

variable's contribution to the overall cost by computing the impact on the objective function of its removal from the current solution. The second SSI component is focused on diversification, maintaining a count of the number of iterations the variable was relaxed in. The heuristic considers the *Variable Cost* divided by the frequency of selection across previous iterations.

### 3.3 Tournament Selection

Tournament selection within the iVRG framework selects the variable with best SSI value from a subset of variables. This subset can be either selected based on their structural relationship to the previously relaxed variable (using the formula defined in Section 3.1), or randomly selected. As previously mentioned, variables that don't share a constraint can also contribute to solution improvement when selected together. Therefore, half the tournaments have the subset of variables selected randomly for the tournament, and the other half use the relationship to the previously relaxed variable.

## 4 Problem Description

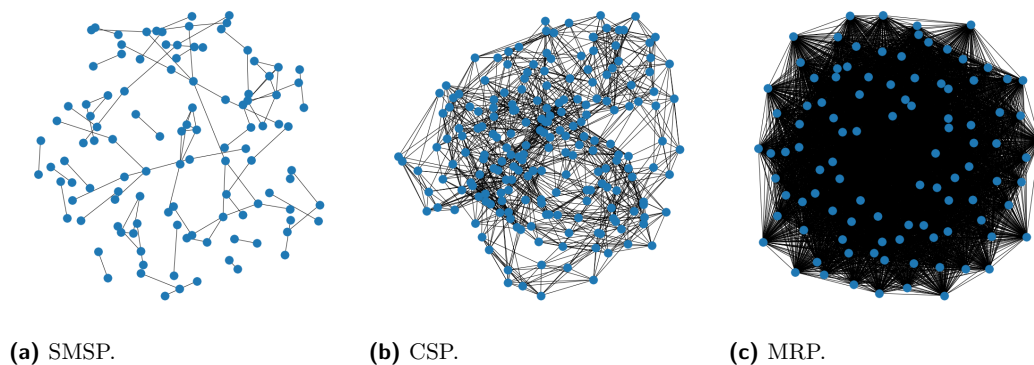
The Steel Mill Slab Problem (SMSP) [8, 14] involves assigning steel orders to slabs while minimising slab wastage. Each slab has a maximum weight capacity, and orders have specific weight and colour. The challenge is to efficiently group orders onto slabs in a way that we have a maximum of two colours per slab while minimising the total waste.

The Car Sequencing Problem (CSP) was initially formulated as a satisfaction problem [1, 16], involving the assignment of a sequence of cars to a limited number of slots in a production line where each bay installs specific options and has a limited capacity. Many optimisation variants with different objective functions have since been proposed (e.g. [11, 3]). In this work we consider the variant proposed by Souza *et al.* [18] that defines the objective function in terms of minimising the total number of options in the cars not placed in the production line.

The Machine Reassignment Problem (MRP) considered here was proposed by Google for the 2012 Roadef Challenge.<sup>1</sup> The problem requires optimising the reallocation of a set of processes to a set of machines with the goal of minimising a multi-objective function: weighted sum of objective function components. The problem further involves a number of constraints related to capacity, conflicting subsets of processes, as well as spread and dependency amongst groups of processes. Due to its complexity and specificity, the MRP has been the focus of many works [2].

The problem type instances were chosen based on what each approach (PG-LNS, CiG-LNS and iVRG) had used for their evaluation in their respective publications. They also represent a diverse set of combinatorial optimisation challenges with varying constraints and complexities (the low propagation of the SMSP, the higher constrained solutions of the CSP, and the density and size of the MRP instances). This variety ensures a comprehensive evaluation of each heuristic's adaptability and scalability. Figure 1 shows the structure of a sample instance for each of the problem types. We note that the SMSP is very sparse, with disconnected components. The MRP on the other hand is extremely dense, while the CSP is somewhat in the middle of the other two problem types in terms of its density.

<sup>1</sup> [https://www.roadef.org/challenge/2012/files/problem\\_definition\\_v1.pdf](https://www.roadef.org/challenge/2012/files/problem_definition_v1.pdf)



■ **Figure 1** Problem structure (variable relationship) of sample instances: (a) Steel Mill Slab Problem, (b) Car Sequencing Problem, and the (c) Machine Reassignment Problem.

## 5 Experimental Setup

The experiments were run on a machine running Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-70-generic) with 16 cores and 32Gb of RAM. All runs had a runtime cutoff of 2 minutes per instance for the 2 smaller problem types: the Car Sequencing Problem and the Steel Mill Slab Problem. While for the more complex Machine Reassignment Problem, the cutoff was 5 minutes per instance. Those runtimes were chosen to be consistent with the values used in previous literature. The MRP was introduced in the ROADEF 2012 with a runtime cutoff of 5 minutes and most of the papers that addressed this problem focused on this setting. The runtime used in the original PG-LNS paper [12] for their CSP experiments was 2 minutes. In the paper that introduced CIG-LNS [6] the authors used an iteration-limit of 1000 for their SMSP experiments rather than a time-limit, therefore we used 2 minutes for the SMSP in these experiments, which generated more than 2000 iterations on average for each approach.

Gecode<sup>2</sup> was the CP solver used for subproblem optimisation in the two smallest instance types. However as it could not handle the large MRP, a dedicated solver was implemented for this problem type. Furthermore, as the approaches have stochastic components, the presented results are the average of 10 runs with different seeds. Table 1 presents the parameter configurations that were used to run the experiments. The failure threshold / first solution improvement were the stopping conditions used by the CP solver per iteration.

■ **Table 1** Configurations parameters for the benchmark experiments.

Parameter	Value
Runtime	120 seconds (SMSP, CSP); 300 seconds (MRP)
Neighbourhood Size	10 variables
Tournament Size	10 variables
Failure Threshold	200

It should be noted that for PG-LNS we implemented the best configuration presented in [12], that iterates through the following three neighborhood operators: Propagation Guided; Reverse Propagation Guided; and purely random selection. However, we defined the

<sup>2</sup> <https://www.gecode.org>

neighbourhood size based on the number of relaxed variables instead of the search space size, in order to compare all approaches on the same neighbourhood size. Similarly for CIG-LNS, we implemented the optimal configuration identified in [6]. In particular the value of  $\alpha$  was set to 0.5, and a dive was performed after every 10 unsuccessful LNS iterations and subsequent to each improve solution.

## 5.1 Benchmark Instances

For the MRP, the three sets of instances from the 2012 RoadeF challenge [9] were used ( $A, B$  and  $X$ ), where each set has 10 instances. The  $A$  instance set is composed of smaller instances with a maximum of 1k variables and domains of size 100. The other two sets of instances are more complex and larger, with up to 50k variables and domains of up to 5k. For the CSP, three sets of hard instances proposed by Caroline Gagne and available on the CSPLib [16] were used. There are 10 instances in each set and the total number of cars per instance (equal to the number of variables, and to domain size -1) is 200, 300 and 400, respectively, for the three sets. Finally, for the Steel Mill Slab Problem, the same set of instance used in [6] was used. This set involves 80 instances, each with 111 variables and domain size 111, divided into four groups based on the number of slab capacities available (2,3,4 and 5).

## 5.2 Evaluation Metrics

The metrics used for our evaluation on the three different problem types are a normalized scoring metric to assess the quality of the solution, and a similarity metric to assess the diversity of the neighborhoods. Note all three problem types are minimisation.

The score metric is that used in ROADEF'12 [9]. The metric measures the distance the solution found is from the best known solution, but also considers how much improvement was made from the initial solution. It is calculated  $((Cost - BK)/initialCost) * 100$ , where  $Cost$  denotes the cost of the solution found,  $BK$  is the best known cost for the problem instance [21, 2, 17, 4], and  $initialCost$  is the cost of the initial solution. For fair comparison, note that the same initial solution was used for all algorithms for a given run on a given instance. This score was chosen because the solution costs have a huge difference in the scale, e.g. the best known solution for MRP instance A2\_1 is 151 and for A2\_2 is 720671511.

The similarity metric represents the average percentage of intersection (common variables in neighborhoods) observed across the first 1,000 iterations of the LNS, as illustrated in the equation below. In the case where an approach did not manage to perform 1,000 iterations in the defined runtime (which occurred only for PG-LNS on the largest MRP problem sets), the metric represents the average percentage of intersection in all iterations.

$$Similarity = \frac{1}{\binom{1000}{2}} \sum_{i=0}^{999} \sum_{j=i+1}^{999} \frac{|N[i] \cap N[j]|}{|N[i]|}$$

# 6 Results

## 6.1 Comparison of different domain-independent neighborhood operators for LNS

We first investigated if the performance previously shown on MRP [19] would hold on the SMSP and the CSP. We compared the three generic neighborhood selection heuristics and a pure random approach ( $Rand$ ), in terms of average score, similarity and number of iterations



across problem sets. The results are given in Table 2, showing that iVRG consistently outperformed the other approaches on all problem sets. CIG-LNS and PG-LNS had varying performance across the problem types, with the latter outperforming the former on the CSP but the opposite the case for the other two problem types. Indeed in our experiments the random approach outperformed both on the SMSP, albeit by a small amount.

Somewhat surprisingly we find that iVRG has the highest similarity in the two smallest problem types. However, we note that it was not significantly higher than the other approaches. Indeed all approaches contain a strong random component, but a very low similarity is not necessarily a good characteristic, for example to try to maintain a balance between diversification and intensification of search.

An analysis of the number of iterations performed by the different approaches also provides insights. Firstly, PG-LNS and CIG-LNS suffered from scalability. In particular for the MRP, the number of iterations performed on the instance sets B and X, which had instances with up to 50k variables and domains of maximum size 5k, were more than 80% less than on the A set. In comparison, Rand and iVRG had less than 50% drop in iterations. The cost of computing the cost-impact / propagation-impact was prohibitive for these large instances.

The iterations on the other two problem sets demonstrate the two extreme cases in neighborhood selection that we wish to avoid, both of which are heavily influenced by a lack of relationship amongst variables in the selected neighborhood. In one case there is no search space to search, as the only consistent values the relaxed variables can take are the values they take in the current solution. Therefore, there are many iterations performed, with few nodes explored. This can be seen for the CSP, with Rand and CIG-LNS performing nearly an order of magnitude more iterations than PG-LNS and iVRG. Indeed analysis of the average nodes explored per iteration shows that Rand and CIG-LNS explored around 10 nodes per iteration on average, compared to 200-300 for PG-LNS and iVRG. We also note that both the latter two approaches were significantly better in terms of score on this problem type compared to the former two.

The opposite case is where there is too large a search space, due to lack of propagation when values assigned. This is the case for the SMSP where the variables are not strongly connected, as shown in Figure 1a. Here, iVRG performed many more iterations than the others, with average nodes per iteration of 100 for iVRG compared to approximately three times as many by the other approaches. Given the failure threshold of 200, this shows that the solver was able to improve most neighborhoods chosen by iVRG, but rarely was able to improve any of the neighborhoods chosen by the other methods for the SMSP.

■ **Table 2** Comparison of iVRG, PG-LNS, CIG-LNS, and Random Selection on the three problem types: Steel Mill Slab (SMSP), Car Sequencing (CSP), and Machine Reassignment (MRP).

Problem	Group	Score				Similarity				#Iterations (x1000)			
		Rand	PG	CIG	iVRG	Rand	PG	CIG	iVRG	Rand	PG	CIG	iVRG
SMSP	2	10.23%	10.24%	10.79%	<b>5.51%</b>	<b>9.01%</b>	10.10%	9.62%	10.24%	1.7	1.7	1.6	<b>4.4</b>
	3	10.80%	11.81%	11.59%	<b>5.17%</b>	<b>9.01%</b>	10.06%	9.76%	10.26%	2.0	1.9	1.9	<b>4.3</b>
	4	5.51%	5.97%	5.68%	<b>2.81%</b>	<b>9.01%</b>	10.11%	9.85%	10.12%	2.3	2.3	2.3	<b>7.6</b>
	5	4.78%	5.57%	4.58%	<b>2.13%</b>	<b>9.01%</b>	10.17%	10.05%	10.08%	2.7	2.7	2.6	<b>7.6</b>
	<b>Overall</b>	<b>7.83%</b>	<b>8.40%</b>	<b>8.16%</b>	<b>3.91%</b>	<b>9.01%</b>	<b>10.11%</b>	<b>9.82%</b>	<b>10.17%</b>	<b>2.2</b>	<b>2.1</b>	<b>2.1</b>	<b>6.0</b>
CSP	200	9.71%	5.36%	8.97%	<b>4.43%</b>	<b>5.00%</b>	5.01%	5.26%	5.51%	78.2	12.7	<b>131.7</b>	18.4
	300	10.36%	5.46%	9.57%	<b>3.83%</b>	<b>3.33%</b>	3.34%	3.44%	3.64%	52.4	9.1	<b>87.3</b>	12.8
	400	11.58%	5.67%	10.11%	<b>3.86%</b>	2.50%	<b>2.50%</b>	2.55%	2.72%	32.9	6.3	<b>55.5</b>	9.1
	<b>Overall</b>	<b>10.55%</b>	<b>5.50%</b>	<b>9.55%</b>	<b>4.04%</b>	<b>3.61%</b>	<b>3.61%</b>	<b>3.75%</b>	<b>3.95%</b>	<b>54.5</b>	<b>9.3</b>	<b>91.5</b>	<b>13.4</b>
MRP	A	3.69%	5.25%	3.17%	<b>2.33%</b>	<b>4.56%</b>	5.11%	8.80%	5.06%	87.3	7.6	<b>98.7</b>	70.1
	B	0.31%	0.94%	0.36%	<b>0.26%</b>	<b>0.26%</b>	0.26%	3.14%	0.35%	<b>52.2</b>	0.8	13.6	44.0
	X	0.46%	0.62%	0.41%	<b>0.34%</b>	0.29%	<b>0.25%</b>	3.69%	0.38%	<b>53.9</b>	0.8	15.7	34.9
	<b>Overall</b>	<b>1.49%</b>	<b>2.27%</b>	<b>1.31%</b>	<b>0.98%</b>	<b>1.70%</b>	<b>1.87%</b>	<b>5.21%</b>	<b>1.93%</b>	<b>64.5</b>	<b>3.0</b>	<b>42.7</b>	<b>49.7</b>

## 6.2 Analysis of iVRG Components

Given the performance of iVRG in the previous section, we next investigated the contribution of each of its three main components to this performance. In particular we compared iVRG against the following iVRG versions: without tournament selection (*NonT*); without using search information (*NonS*); and finally without using the structural relationship (*NonR*). For *NonS*, variables were chosen randomly, albeit maintaining the structural relationship with previously selected variables in the neighborhood.

The results are presented in Table 3, and show that overall the use of structural relationship had the biggest impact. *NonR* consistently had the biggest drop in performance compared to iVRG. Tournament selection was the next most important, with significant drops in performance on the CSP and the MRP. Somewhat surprisingly, the results of *NonS* demonstrate that the search state information has only a relatively small impact on iVRG performance across all three problem sets, compared to the other two components.

Interestingly, we find a stronger correlation between similarity and performance here than in the previous table. *NonR* has consistently higher similarity than iVRG. This indicates a more diverse neighborhood selection in iVRG due to the effect of variable relationships, which restrict the variables available to relax to a different group (based on relationships of the first chosen variable) every iteration.

*NonT* also had higher similarity for all except the sparse SMSP problems, with these problems being the only ones where *NonT* had comparable performance with iVRG. Of course, tournament selection has less impact on smaller instances. Note that under the current settings of 10 for both neighborhood size and tournament size, 100 variables are considered in each LNS iteration when choosing the neighborhood, and SMSP instances had only 111 variables.

The average iterations metric reveals the importance of the relationship information in neighborhood selection in iVRG. Without this, iVRG suffers from the same pitfalls as discussed in Table 2. The sparseness of the SMSP results in searching a large search space, even with just 10 variables in the neighborhood, due to the lack of propagation. On the other hand, it performs a much greater number of iterations on the two other problem sets, but many of these required little or no search as again disconnected neighborhood variables in much more constrained instances resulting in most variables being assigned without search.

■ **Table 3** iVRG compared to iVRG without: Tournament Selection (*NonT*), search state information (*NonS*), and variable-relationship (*NonR*).

Problem	Group	Score				Similarity				#Iterations (x1000)			
		iVRG	NonT	NonS	NonR	iVRG	NonT	NonS	NonR	iVRG	NonT	NonS	NonR
SMSP	2	<b>5.51%</b>	6.23%	6.24%	10.38%	10.24%	9.92%	<b>9.82%</b>	22.42%	<b>4.4</b>	4.3	4.1	1.8
	3	5.17%	<b>4.93%</b>	5.34%	14.41%	10.26%	10.03%	<b>9.96%</b>	21.57%	4.3	<b>4.5</b>	4.4	2.1
	4	2.81%	<b>2.79%</b>	2.82%	13.68%	10.12%	9.88%	<b>9.85%</b>	21.27%	7.6	7.9	<b>8.0</b>	2.1
	5	2.13%	<b>2.06%</b>	2.28%	17.38%	10.08%	9.90%	<b>9.84%</b>	20.97%	7.6	<b>8.0</b>	7.4	1.9
	Overall	<b>3.91%</b>	4.00%	4.17%	13.96%	10.17%	9.93%	<b>9.87%</b>	21.56%	6.0	<b>6.2</b>	6.0	2.0
CSP	200	<b>4.43%</b>	10.47%	4.50%	9.59%	5.51%	12.57%	<b>5.02%</b>	6.03%	18.4	14.1	14.2	<b>86.8</b>
	300	<b>3.83%</b>	11.11%	4.28%	10.71%	3.64%	10.92%	<b>3.35%</b>	4.02%	12.8	9.7	9.9	<b>56.1</b>
	400	<b>3.86%</b>	9.57%	3.87%	11.32%	2.72%	7.21%	<b>2.50%</b>	3.06%	9.1	6.1	7.4	<b>34.9</b>
	Overall	<b>4.04%</b>	10.38%	4.22%	10.54%	3.95%	10.23%	<b>3.62%</b>	4.37%	13.4	10.0	10.5	<b>59.3</b>
MRP	A	<b>2.33%</b>	6.04%	2.66%	5.85%	5.06%	23.83%	<b>4.75%</b>	10.90%	70.1	65.4	63.1	<b>91.2</b>
	B	<b>0.26%</b>	0.74%	0.29%	0.39%	0.35%	20.10%	<b>0.28%</b>	0.61%	44.0	8.5	44.9	<b>57.0</b>
	X	<b>0.34%</b>	0.90%	0.37%	0.46%	0.38%	20.83%	<b>0.31%</b>	0.67%	34.9	7.2	40.8	<b>89.2</b>
	Overall	<b>0.98%</b>	2.56%	1.11%	2.23%	1.93%	21.59%	<b>1.78%</b>	4.06%	49.7	27.0	49.6	<b>79.1</b>



The advantages of tournament selection is particularly prominent on the large MRP B and X instances, where the cost of computing heuristic values and sorting across 1000s of variables is prohibitive. Here, in the worst case, the SSI must be calculated for 50,000 variables, compared to 100 variables with tournament selection. This demonstrates that the benefit of tournament selection is not only in terms of diversification, as shown by NonT similarity scores, but also in terms of scalability.

## 7 Conclusion

We have demonstrated the generalisability of the recently proposed iVRG. It was shown to significantly outperform similar approaches (PG-LNS and CIG-LNS) on three challenging problem types of different characteristics. Additional analysis revealed that it is able to avoid two of the main pitfalls of neighborhood selection, unlike the comparison approaches. The main iVRG component from this respect was the use of static information regarding the relationship of variables, in order to select neighborhoods with a high degree of connectivity.

An ablation study of the main components was then performed, considering the use by iVRG of: static structural information; dynamic search state information; and finally the use of tournament selection within the neighborhood operator. The results revealed that the problem structure was the most important aspect, followed closely by tournament selection. The former contributes to the algorithm’s ability to adapt for distinct problem characteristics, effective even for problem such as Steel Mill Slab with a low level of relationship between decision variables. Tournament selection was shown to be effective in increasing diversification, and scalability. While, the search state information had relatively low impact in iVRG compared to the other two components, it still yielded consistent improvements over random selection. Indeed these two components are generic and could possibly result in improved performance if plugged into other LNS approaches.

---

## References

- 1 Christian Artigues, Emmanuel Hebrard, Valentin Mayer-Eichberger, Mohamed Siala, and Toby Walsh. Sat and hybrid models of the car sequencing problem. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 268–283. Springer, 2014.
- 2 Dario Canales, Nicolas Rojas-Morales, and Maria-Cristina Riff. A survey and a classification of recent approaches to solve the google machine reassignment problem. *IEEE Access*, 8:88815–88829, 2020.
- 3 Jens Gottlieb, Markus Puchta, and Christine Solnon. A study of greedy, local search, and ant colony optimization approaches for car sequencing problems. In *Applications of Evolutionary Computing: EvoWorkshops 2003*, pages 246–257. Springer, 2003.
- 4 Stefan Heinz, Thomas Schlechte, Rüdiger Stephan, and Michael Winkler. Solving steel mill slab design problems. *Constraints*, 17(1):39–50, 2012.
- 5 Philippe Laborie and Daniel Godard. Self-adapting large neighborhood search: Application to single-mode scheduling problems. *Proceedings MISTA-07, Paris*, 8, 2007.
- 6 Michele Lombardi and Pierre Schaus. Cost impact guided lns. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 293–300. Springer, 2014.
- 7 Jean-Baptiste Mairy, Yves Deville, and Pascal Van Hentenryck. Reinforced adaptive large neighborhood search. In *The Seventeenth International Conference on Principles and Practice of Constraint Programming (CP 2011)*, page 55. Springer Berlin/Heidelberg, Germany, 2011.
- 8 Ian Miguel. CSPLib problem 038: Steel mill slab design. <http://www.csplib.org/Problems/prob038>.

- 9 H Murat Afsar, Christian Artigues, Eric Bourreau, and Safia Kedad-Sidhoum. Machine reassignment problem: the roaddef/euro challenge 2012, 2016.
- 10 Vinod Nair, Mohammad Alizadeh, et al. Neural large neighborhood search. In *Learning Meets Combinatorial Algorithms at NeurIPS2020*, 2020.
- 11 Laurent Perron and Paul Shaw. Combining forces to solve the car sequencing problem. In *International Conference on Integration of Artificial Intelligence (AI) and Operations Research (OR) Techniques in Constraint Programming*, pages 225–239. Springer, 2004.
- 12 Laurent Perron, Paul Shaw, and Vincent Furnon. Propagation guided large neighborhood search. In *International Conference on Principles and Practice of Constraint Programming*, pages 468–481. Springer, 2004.
- 13 Stefan Ropke and David Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, 40(4):455–472, 2006.
- 14 Pierre Schaus, Pascal Van Hentenryck, Jean-Noël Monette, Carleton Coffrin, Laurent Michel, and Yves Deville. Solving steel mill slab problems with constraint-based techniques: Cp, lns, and cbls. *Constraints*, 16:125–147, 2011.
- 15 Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *International conference on principles and practice of constraint programming*, pages 417–431. Springer, 1998.
- 16 Barbara Smith. CSPLib problem 001: Car sequencing. <http://www.csplib.org/Problems/prob001>.
- 17 Filipe Souza, Diarmuid Grimes, and Barry O’Sullivan. A large neighborhood search approach for the data centre machine reassignment problem. In *Irish Conference on Artificial Intelligence and Cognitive Science*, pages 397–408. Springer, 2022.
- 18 Filipe Souza, Diarmuid Grimes, and Barry O’Sullivan. Variable-relationship guided lns for the car sequencing problem. In *Irish Conference on Artificial Intelligence and Cognitive Science*, pages 437–449. Springer, 2022.
- 19 Filipe Souza, Diarmuid Grimes, and Barry O’Sullivan. Improved variable-relationship guided lns for the data centre machine reassignment problem. In *Irish Conference on Artificial Intelligence and Cognitive Science*, page to appear. Springer, 2023.
- 20 Charles Thomas and Pierre Schaus. Revisiting the self-adaptive large neighborhood search. In *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 557–566. Springer, 2018.
- 21 Ayad Turky. *Bi-level hyper-heuristic approaches for combinatorial optimisation problems*. PhD thesis, RMIT University, 2019.
- 22 Yaoxin Wu, Wen Song, Zhiguang Cao, and Jie Zhang. Learning large neighborhood search policy for integer programming. *Advances in Neural Information Processing Systems*, 34:30075–30087, 2021.