Journal of Heuristics manuscript No.
(will be inserted by the editor)

A combined approach for analysing heuristic algorithms

Jeroen Corstjens · Nguyen Dang · Benoît Depaire · An Caris · Patrick De Causmaecker

Received: date / Accepted: date

Abstract When developing optimisation algorithms, the focus often lies on obtaining an algorithm that is able to outperform other existing algorithms for some performance measure. It is not common practice to question the reasons for possible performance differences observed. These type of questions relate to evaluating the impact of the various heuristic parameters and often remain unanswered. In this paper, the focus is on gaining insight in the behaviour of a heuristic algorithm by investigating how the various elements operating within the algorithm correlate with performance, obtaining indications of which combinations work well and which do not, and how all these effects are influenced by the specific problem instance the algorithm is solving. We consider two approaches for analysing algorithm parameters and components — functional ANOVA and multilevel regression analysis — and study the opportunity of using both approaches jointly. We present the results of a combined methodology that is able to provide more insights than when the two approaches are used separately. The illustrative case study examined in this paper analyses a large neighbourhood search algorithm applied on the Vehicle Routing Problem with Time Windows.

Keywords Functional Analysis of Variance · Multilevel Regression · Algorithm Performance · Vehicle Routing Problem with Time Windows · Large Neighbourhood Search

J. Corstjens, B. Depaire, A. Caris
UHasselt, Research Group Logistics
Agoralaan, 3590 Diepenbeek, Belgium
E-mail: jeroen.corstjens@uhasselt.be

N. Dang, P. De Causmaecker
KU Leuven, CODeS, imec-ITEC
Etienne Sabbelaan 53 (7659), 8500 Kortrijk, Belgium
E-mail: nguyenthithanh.dang@kuleuven.be

1 Introduction

Experimentation on heuristic algorithms commonly entails the computational testing of an algorithm on some benchmark problem set and comparing results against those of known algorithms. The objective is to be better than the competing algorithms for some performance measure (e.g., solution quality or computation speed). Such an approach for evaluating algorithms does not explain, however, 'why' one method achieves better results than other ones [8]. Which algorithm parameters contribute more or less to a good performing algorithm? Are there specific combinations of parameter values that work well or not? Or is the observed superior performance due to a more efficient implementation of the algorithm? These kind of questions often remain unanswered when following a competitive evaluation methodology. Nevertheless, such insights are necessary to truly understand algorithm behaviour [10] [14].

Recently, methodologies have been proposed for evaluating heuristic algorithms with the aim of gaining a more profound understanding of the heuristic parameters' impact on performance [1] [4] [9] [10]. In this research, we consider two model-based methodologies ([10], [1]) and investigate for a particular experiment case whether both approaches have consistency in the insights they deliver. More importantly, we look into possible opportunities for a complementary use of both methodologies. Our focus therefore lies on answering the following questions. How can both methodologies be formulated in a combined methodology and what is the added value of jointly applying both approaches? And are the insights deducted from the analysis results of both approaches consistent or are there any differences observed?

The investigated methodologies are functional analysis of variance (fANOVA) [10] and multilevel regression [1]. fANOVA ranks the effects of algorithm parameters according to the amount of variance in the performance measure data they explain, giving an indication of which effects are most important to performance. The multilevel regression methodology explicitly separates the performance impact of algorithm parameters and problem instances. It is focused on quantifying the relationship algorithm parameters have with the performance measure and how this relationship is moderated by the specificities of a certain problem instance. In this paper, we combine both these methodologies by relying on the importance analysis provided by fANOVA to formulate a proper regression model for the multilevel methodology. This prevents an overly complex regression model with many variable (interactions) that contribute little to performance. The multilevel regression model offers a more detailed analysis of the algorithm since it can investigate algorithm parameter effects for a specific setting of the other parameters and a specific problem instance. Moreover, the interpretation of effects in the fANOVA analysis is carried out through visual inspection of plots, which might be difficult for interaction effect. The interpretation of the regression analysis is based on quantified effects and plots are merely used to support and visualise

interpretation. The regression analysis also provides a statistical significance test to indicate whether there really is a link between the values chosen for a certain algorithm parameter and the obtained performance or whether any observed relationships are likely due to chance.

Both approaches and their combination are demonstrated on a case study in which a number of parameter settings for a Large Neighbourhood Search (LNS) algorithm are tested on a number of problem instances for the Vehicle Routing Problem with Time Windows (VRPTW). The paper is organized as follows: the methodologies of fANOVA and multilevel regression analysis are introduced in section 2 along with the proposed combination of the two approaches, the case study with LNS and VRPTW is explained in section 3, the applications of fANOVA and multilevel regression analysis on the case study are shown in section 3.1. Finally, conclusions and future work are given in section 4.

2 fANOVA and multilevel regression models

2.1 fANOVA

The *fANOVA* proposed in [10] is an approach for analysing the importance of algorithm parameters on the algorithm performance using a random forest prediction model and the functional analysis of variance [7]. The approach studies the contribution of every single parameter and every parameter interaction on the performance of the algorithm. In particular, given a data set of performance values of different algorithm parameter settings on a number of problem instances, fANOVA first builds a random forest-based prediction model to predict the average performance of every algorithm parameter setting over the whole problem instance space. Afterwards, the functional analysis of variance [7] is applied on the prediction model to decompose the overall algorithm performance variance into additive components, each one corresponds to a subset of the algorithm parameters. The ratio between the variance associated with each component and the overall performance variance is used as an indicator of the importance of the corresponding algorithm parameter subset. For example, the following shows a part of the output of an fANOVA analysis on the Large Neighborhood Search algorithm used in our study for a Vehicle Routing Problem with Time Windows problem instance set:

```
Sum of fractions for main effects 75.10%
Sum of fractions for pairwise interaction effects 6.26%
72.62% due to main effect: repair
1.72% due to main effect: destroy
1.60% due to interaction: repair x destroy
```

The interpretation of the first two lines is that 75.10% of the algorithm performance variance can be explained by single parameters, and 6.26% by

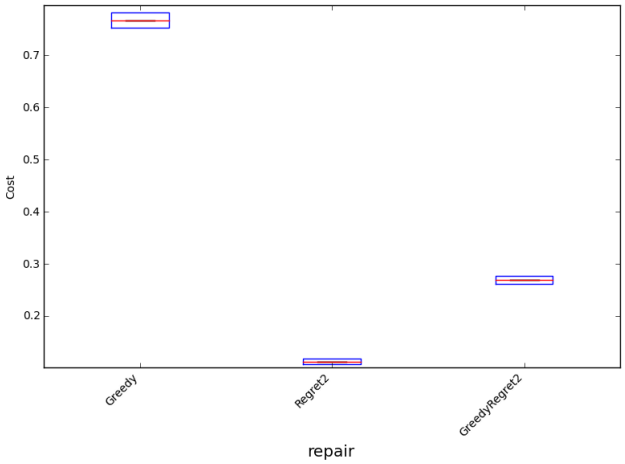


Fig. 1: fANOVA's marginal plot for the main effect of the single parameter *repair*.

the interactions of every pair of algorithm parameters. The remaining 18.64% is explained by higher order ($\geq$3) interactions and error inherent in the model. The third line indicates that the algorithm parameter *repair* is the most important parameter, as the parameter itself can explain a huge part (72.62%) of the overall algorithm performance variance. The other parameter *destroy* and the pairwise interaction of the two parameters *repair* and *destroy* are less important. The details of the algorithm and those parameters will be further described in sections 3 and 3.1.

In addition to the value indicating the importance of each algorithm parameter subset, fANOVA also provides some insights on which regions are good and bad (with a degree of uncertainty) for each parameter inside the subset throughout a marginal plot.

Given a specific value for each algorithm parameter in the subset, the corresponding marginal prediction value is the average performance value of the algorithm over the whole configuration space associated with all parameters not belonging to the subset. A marginal plot shows the mean and the variance of the marginal prediction values given by the random forest's individual trees. Figure 1 shows the marginal plot of the algorithm parameter *repair*. This categorical parameter has a domain of three values: *Greedy*, *Regret2* and *GreedyRegret2*, each of which is associated with a boxplot in Figure 1. The boxplot for *Greedy*, for example, shows the mean and the variance of the average algorithm performance values over the whole problem instance space of all algorithm parameter settings having *repair* = *Greedy*. The plot implies that *Regret2* is the best choice for the parameter *repair*.

An implementation of fANOVA is provided by the approach's authors as a Python package at `https://github.com/frank-hutter/fanova`. As a choice of implementation, the package only gives analysis results on the single parameter and pairwise interaction effects. The higher order (≥ 3) interactions are left out, probably due to potentially expensive computation time required and the fact that in many practical cases, single parameter and pairwise interaction effects are usually sufficient to explain the majority of algorithm performance variance.

2.2 Multilevel regression

The multilevel methodology proposed by Corstjens et al. [1] is an approach for analysing the relationships between algorithm parameters, problem instance characteristics and algorithm performance using a multilevel regression (MLR) model. The approach studies how the performance measure (e.g., distance travelled) will change when modifying a parameter from one value to another and how this change is influenced by the problem instance the algorithm is solving. Unlike fANOVA, regression models do not focus on the variance in the data that is explained by each of the investigated variables, but on estimating the variable coefficients [11]. More precisely, regression is applied to describe the relationship between a response variable and variables that explain the response, the explanatory variables. This relationship is formulated mathematically in a regression model which describes how the response value will change when an explanatory variable changes with some value [12]. For every calculated coefficient estimate a confidence interval is provided to indicate whether the estimated performance change by some parameter value is significantly different from zero or not. If not, the observed impact is likely due to chance.

Furthermore, the multilevel aspect of the methodology enables to efficiently study how effects vary by group by relying on multilevel models[2]. Within the context of heuristic experimentation it is possible to identify a multilevel structure when experimenting with different parameter settings on a single problem instance of some combinatorial optimisation problem. We interpret a parameter setting as a set of values and included operators. Any observed differences in performance can then be attributed to the algorithm parameters and not the problem instance. To expose the influence of the problem instance, this structure is repeated for multiple problem instances. It can then be analysed how the performance impact of the various parameters changes when considering different problem instances. It enables answering questions like "Does a heuristic operator work equally well on instances with a few customers and instances with many customers? When service time is long on average, is it better to lower the cooling rate within simulated annealing or increase it? ..." These are the types of questions that can be answered by using a multilevel model [1].

Given a data set of performance values of different algorithm parameter settings on different problem instances, a multilevel regression model is formulated to predict the average expected performance for a particular parameter setting and problem instance. Note that the interpretation of predictions between fANOVA and MLR differs. Evaluating a certain parameter by fixing it at different values, fANOVA predicts an average performance for every value taking into account all values of all other algorithm parameters and the entire problem space. MLR, on the other hand, predicts for each value the average performance given a fixed setting of the other parameters and considering a specific problem instance. The regression model therefore enables a more detailed analysis of parameter effects.

The multilevel regression model can be considered as an extension to a classical regression model in which the coefficients have their own probability model, being the second, higher level. It has accompanying parameters which are the predictors at this second level [5]. A possible formulation of a multilevel regression model — using the same example as in section 2.1 — is given in equations (1) to (3) where three algorithm parameters are considered and three problem instance characteristics operating at the problem (i.e., group) level.

$$Y_i = \alpha_{j[i]} + \beta_{1j[i]} Greedy_i + \beta_{2j[i]} Regret2_i + \beta_{3j[i]} Random_i + \epsilon_i \quad (1)$$

$$\alpha_j = \mu_0^{\alpha} + \mu_1^{\alpha} customers_j + \mu_2^{\alpha} demand_j + \mu_3^{\alpha} servicetime_j e + \eta_j^{\alpha} \quad (2)$$

$$\beta_{kj} = \mu_0^{\beta_k} + \mu_1^{\beta_k} customers_j + \mu_2^{\beta_k} demand_j + \mu_3^{\beta_k} servicetime_j + \eta_j^{\beta_k} \quad (3)$$

where

$i \in I$ denotes the scenario, a combination of a certain problem instance with a certain parameter setting
$j \in J$ denotes the problem instance
The index variable $j[i]$ codes problem instance membership ($j[i] = j$), e.g., $j[90] = 5$ means the 90th scenario solves problem instance 5
Y_i is the objective function value of scenario i
$Greedy_i$, $Regret2_i$ and $Random_i$ are the values set for the parameters in scenario i
$customers_j$, $demand_j$ and $servicetime_j$ are the values for these problem instance characteristics in problem instance j
α_j is the varying regression intercept, representing the solution quality given problem instance j when all algorithm parameters are set equal to 0
β_{kj} represents the varying effect of algorithm parameter k on Y given scenario i and problem instance j
μ_0 represents a mean problem effect
$\mu_{1,2,3}$ represent the effect of the problem-level predictors on the varying algorithm parameter effects
η_j is the error at the problem level and is assumed to be $\sim N(0,\sigma^2)$

ϵ_i is the random error of scenario i and is assumed to be $\sim N(0,\sigma_e^2)$

Equation (1) represents the lowest level where the objective function value for scenario i is observed, with a scenario defined as the combination of a certain problem instance with a certain parameter setting. The objective function value observed is hypothesised to depend on the values that are set for the various heuristic parameters, as indicated by the variables $Greedy_i$, $Regret2_i$ and $Random_i$, and with the β coefficients representing the performance impact of each parameter. Equations (2) and (3) represent the problem level and define how the effects in equation (1) (i.e., the β's) are moderated by the problem instance characteristics, represented by the variables $customers_j$, $demand_j$ and $servicetime_j$. With this model we can learn the impact a single algorithm parameter has on performance and how it is influenced by the problem instance characteristics.

In the next subsection we explore the complementary use of fANOVA and multilevel regression.

2.3 A combined methodology

For the multilevel regression approach, the search for a suitable regression model that includes all relevant variables can be a cumbersome task [5]. The more variables and variable interactions are included, the more complex the model becomes and the more arduous it is to interpret the estimated effects. The challenge thus lies in deciding which explanatory variables and interactions to include in the model. We tackle this issue by relying on the results of the fANOVA analysis. Since it gives a ranking on the importance of effects, a regression model could then be formulated based on this ranking in order to prevent an overly complex model with many variables. The regression analysis can then more easily focus on these important effects, enabling a more detailed study of how they relate to the response variable. Furthermore, the multilevel regression adds contribution on the importance analysis by calculating confidence intervals for each of the effects. This indicates which effects are actually statistically significant and which are likely due to chance.

In the following sections a case study is performed on both analysis approaches.

3 Case study

Experiments are performed on a Large Neighbourhood Search algorithm (LNS). Our implementation of this algorithm is a simplification of the well-known

Adaptive Large Neighbourhood Search (ALNS) metaheuristic developed by Pisinger and Ropke [13] that is able to solve multiple variants of the vehicle routing problem, among which the VRPTW. The algorithm iteratively destroys and repairs the current solution, each time randomly selecting a destroy and repair operator from a set of operators. The more an operator has contributed to finding a better solution, the greater the probability it will be chosen in future iterations. This process is repeated until some stopping criterion is met. This algorithm was chosen because of its popularity and the multitude of parameters it contains, making it a suitable research subject for parameter analysis. However, a simplification is performed to reduce the number of parameters to investigate as it is currently not our aim to make performance statements about the ALNS, but rather focus on establishing the methodology to evaluate heuristic methods. Therefore, a less elaborate version of the heuristic method is preferred. The LNS algorithm does not adjust the probabilities for selecting repair and destroy operators every iteration based on their performance, but keeps them fixed and equal throughout the search process. We also consider less operators, more specifically three destroy and two repair operators.

The LNS algorithm is run on a data set consisting of 4000 different combinations of problem instances and parameter settings. The data set is generated according to a two-phase sampling scheme as applied in [1]. First, 200 instances for the vehicle routing problem with time windows (VRPTW) are generated by drawing random values for the problem instance characteristics listed in Table 1. For each of the generated problem instances 20 parameter setting variants are defined again by drawing random values for each of the algorithm parameters listed in Table 2.

The algorithm is run for each of the 4000 scenarios and returns a total cost measure indicating the total distance travelled by all vehicles to provide service to all customers in the problem.

3.1 Analysis of results

First, fANOVA is applied on the given algorithm performance data set. Then, a multilevel regression model is formulated based on the importance analysis provided by fANOVA, in particular all algorithm parameters and problem instance characteristics having a contribution percentage value higher than 1% are included. As will be discussed, the conclusions of both approaches are quite consistent, however, not all effects taken from fANOVA are statistically significant according to the multilevel regression model.

Table 1: Problem instance characteristics of the VRPTW and their ranges

Problem characteristics		
Characteristic	**Type**	**Value ranges**
number of customers	Integer	U[25, 400]
customer demand	Integer	U[10,50]
average service time	Integer	TRIA(min,max) min∼U[10,30] max∼U[30,50]
average time window width	Integer	TRIA[min,max] min∼U[20,50] max∼U[50,80]
maximum running time	Integer	TRIA(60,1800)

Only the characteristics that will be used in our algorithm performance analysis are listed. For a full list of all characteristics, the reader is referred to Table 4 in the Appendix.

Table 2: Parameters of the Large Neighbourhood Search algorithm

Algorithm parameters		
Parameter	**Type**	**Value ranges**
random seed	Integer	U[1, 1000000]
determinism parameter	Integer	U[1, 100]
noise parameter	Continuous	U[0, 1]
cooling rate	Continuous	U[0.01,0.99]
start temperature control parameter	Continuous	U[0.01,1]
destroy operators	Categorical	Random, Worst, Related, RandomWorst, RandomRelated, WorstRelated
repair operators	Categorical	Greedy, Regret2, GreedyRegret2

3.1.1 Application of fANOVA

fANOVA is basically a tool for analysing the importance of algorithm parameters on algorithm performance over a problem instance set. However, it can also be used to study the interaction between the algorithm parameters and the problem instance's characteristics by simply adding those features into the prediction model of fANOVA. The features are treated exactly in the same way as the algorithm parameters. In the fANOVA analysis on our case study, the definition of the algorithm parameters and the problem instance features, including names, types and domains, are the same as described in Tables 1 and 2.

Since the range of the cost values returned by the algorithm can vary among different instances, we first normalize the cost on an instance-basis before fANOVA is applied:

$$p_j^c = \frac{(f_j^c - \min_{c' \in C_j}, f_j^{c'})}{(\max_{c' \in C_j}, f_j^{c'} - \min_{c' \in C_j}, f_j^{c'})} \tag{4}$$

where f_j^c and p_j^c are the original and the normalized cost values of parameter setting c on problem instance j, and C_j is the set of all parameter settings that have been run on instance j.

Following is the output generated by fANOVA. Since we want to focus on important effects, only the ones with the contribution percentage values higher than 1% are listed. This threshold is chosen arbitrarily.

```
Sum of fractions for main effects 60.56%
Sum of fractions for pairwise interaction effects 16.54%
53.03% due to main effect: repair
4.73% due to interaction: repair x customer_number
4.36% due to interaction: repair x destroy
3.52% due to main effect: customer_number
3.05% due to interaction: destroy
2.96% due to interaction: destroy x customer_number
```

The marginal plot for each effect is given in Figure 2. Since we are only interested in the algorithm parameters and their interactions with the problem instance features, the main effect *customer_number* is omitted.

The single parameter *repair* explains a huge part (53.03%) of the total algorithm performance variance, indicating that this parameter plays the most important role in the performance of the algorithm. Figure 2a shows that *Greedy* is clearly the worst choice for the parameter *repair*. The algorithm gains the best overall performance with *Regret2* as the only repair operator, although *GreedyRegret2* comes quite close. How the impact of the chosen repair operator(s) changes given different problem instance sizes is explained in Figure 2b. We can see that the disadvantage of using the repair operator *Greedy* is getting clearer as the number of customers increases, especially when the number of customers is larger or equal to 50. The performance distinction between the two repair operators *Regret2* and *GreedyRegret2* only starts to become visible when the number of customers reaches 200.

The second categorical algorithm parameter, *destroy*, has much less importance than *repair*. For this parameter, the choice of values, sorted in increasing order of marginal normalized cost values — i.e., from good to bad performance —, is as follows: *RandomRelated*, *RandomWorstRelated*, *Random*, *WorstRelated*, *RandomWorst*, *Related*, *Worst*. The influence of different problem sizes on the impact of the chosen destroy operator(s) is depicted in Figure 2d, but this marginal plot is difficult to interpret visually.

The final marginal plot shows the pairwise interaction between the two categorical parameters — plotted in Figure 2e — and indicates consistency with the observations obtained from the main effects: *Greedy* is always the worst choice, despite its combination with any destroy operator; and the choice of *Regret2* generally offers better performance than *GreedyRegret2*, although not very clear. Moreover, among all combinations of *repair* and *destroy* operators, using *Regret2* as a *repair* operator combined with the destroy operator *Random* is predicted to perform best.

In the next section the multilevel evaluation methodology is applied and compared to the discussed fANOVA findings.

3.1.2 Application of multilevel regression

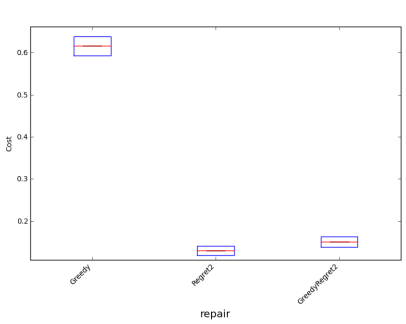
The multilevel regression analysis is performed on a second, independent, data set of 4000 scenarios which was generated according to the same sampling procedure as for the data set used by the fANOVA analysis. The motivation is to prevent overfitting analysis findings to a single data set. Searching for a model that is the best fit for a single data set might risk fitting noise in the data — patterns present in the sample but not in the population — and might result in a model which performs poorly on other data points from the same population. A fitted model should be able to make accurate predictions for new data points instead of only the data points used to learn the model [3]. Therefore, the multilevel regression analysis is performed using new sample data.

The fitted multilevel regression model includes the effects that account for 1% or more of the variance in the data according to the fANOVA results. It is then verified whether the model complied with the statistical assumptions of independence, normality and homoscedasticity typically underlying regression models. Such a model is found after taking the reciprocal transformation of the response variable and the cube root of the centred problem instance characteristic *Customer_Number*, the same transformations as applied in Corstjens et al.[1]. The resulting model describes a non-linear relationship between the performance measure and the algorithm parameters explaining it. It is formulated as in equations (5)-(7).

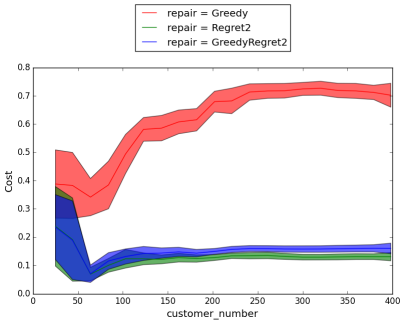
$$\begin{aligned}\frac{1}{Y_i} =& \alpha_{j[i]} + \beta_{1j[i]} Greedy + \beta_{2j[i]} Regret2 + \beta_{3j[i]} Random + \cdots + \\ & \beta_{8j[i]} RandomRelated + \beta_{9j[i]} Greedy \times Random + \cdots + \\ & \beta_{20j[i]} Regret2 \times RandomRelated + \epsilon_i \end{aligned} \tag{5}$$

$$\alpha_j = \mu_0^{\alpha} + \mu_1^{\alpha} Customer\_Number^{1/3} + \eta_j^{\alpha} \tag{6}$$

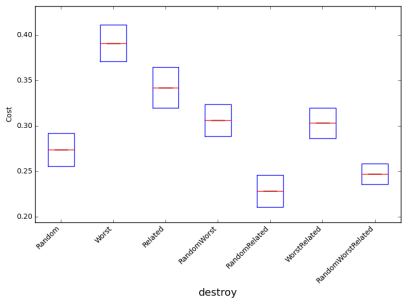
$$\beta_{zj} = \mu_0^{\beta_z} + \mu_1^{\beta_z} Customer\_Number^{1/3} + \eta_j^{\beta_z} \tag{7}$$



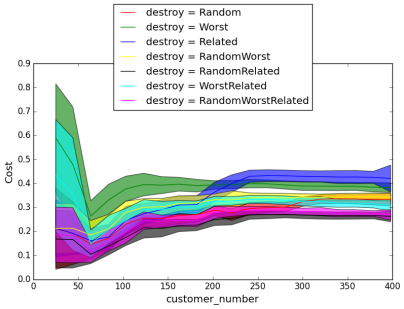
(a) repair



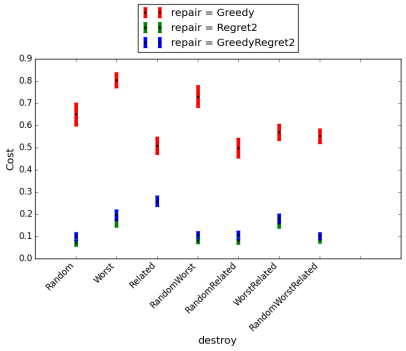
(b) repair x customer_number



(c) destroy



(d) destroy x customer_number



(e) repair x destroy

Fig. 2: Marginal plots of main effects and pairwise interaction effects of the fANOVA analysis (Only effects with the percentage of contribution on performance variance larger than 1% are shown.)

All operator effects are modelled as varying effects depending on the problem instance characteristic *Customer_Number* as indicated by the fANOVA output. Table 3 lists all significant effects, meaning the 95% confidence interval of the impact estimate [l-95% CI, u-95% CI] does not include zero. The complete regression table can be found in Table 5 in the Appendix. The regression table shows significant effects for all but one individual operator effect and for all but one interaction between *Greedy* and the destroy operators. The interactions between *Regret2* and the destroy operators are not indicated significant. Furthermore, the individual repair operator effects are significantly influenced by the the problem size. The same goes for three individual destroy operator effects (*Random*, *Related* and *RandomRelated*) and four interactions between *Greedy* and the destroy operators. It influences the interaction of *Greedy* with *Random*, *Related*, *RandomWorst* and *WorstRelated*. All other operator effects show no significant influence of problem size. In the following paragraphs we discuss the interpretation of the significant effects.

Table 3: Regression table of significant effects[a]

Variable	Estimate	Est.Error	l-95% CI	u-95% CI
Intercept[b,c]	4,151.65	128.11	3,899.79	4,398.53
Greedy	−133.46	5.48	−144.15	−122.90
Regret2	16.98	3.66	9.78	24.18
Random	21.28	3.47	14.27	28.06
Worst	−11.32	3.69	−18.62	−3.98
Related	−60.46	4.70	−69.83	−51.33
RandomWorst	9.20	3.66	1.91	16.30
WorstRelated	−13.92	3.56	−20.97	−6.96
Customer_Number$^{1/3}$	−453.86	29.99	−513.62	−396.08
Greedy × Random	−88.45	7.90	−103.98	−72.62
Greedy × Worst	−89.67	8.87	−107.18	−72.35
Greedy × Related	68.31	6.76	55.07	81.35
Greedy × RandomWorst	−95.71	7.61	−110.70	−80.76
Greedy × RandomRelated	15.19	5.74	3.96	26.44
Greedy × Customer_Number$^{1/3}$	−16.51	1.23	−18.89	−14.11
Regret2 × Customer_Number$^{1/3}$	2.91	0.81	1.35	4.49
Random × Customer_Number$^{1/3}$	3.84	0.77	2.35	5.35
Related × Customer_Number$^{1/3}$	−9.73	1.05	−11.80	−7.70
RandomWorst × Customer_Number$^{1/3}$	1.59	0.82	0.02	3.21
Greedy × Random × Customer_Number$^{1/3}$	−15.83	1.76	−19.25	−12.36
Greedy × Related × Customer_Number$^{1/3}$	10.56	1.54	7.50	13.53
Greedy × RandomWorst × Customer_Number$^{1/3}$	−11.61	1.68	−14.87	−8.36
Greedy × WorstRelated × Customer_Number$^{1/3}$	3.13	1.45	0.25	5.94

[a] Since the reciprocal transformation of the response variable returned very small values causing difficulties in the sampling procedure of the *brms* package, all transformed (response) values were multiplied by a constant 100 000 000.

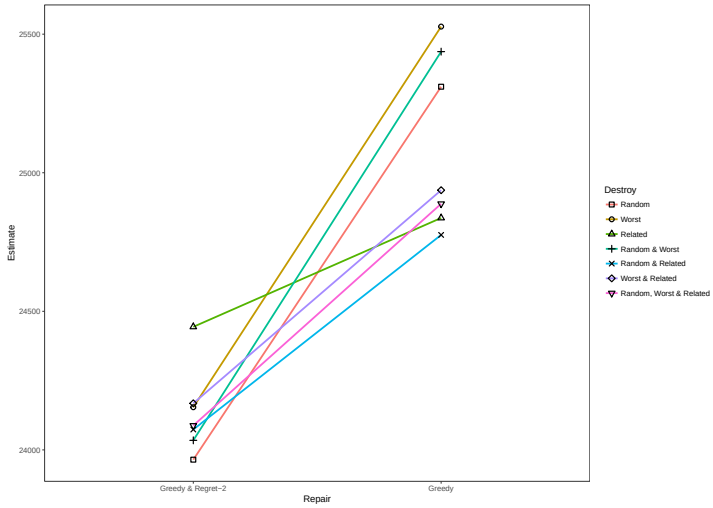
[b] The effects of Greedy & Regret-2 and Random, Worst & Related, the reference levels for the repair and destroy operator dummies, are accounted for in the Intercept.

[c] The Intercept value is backtransformed to the original scale through division by 100 000 000 and taking the inverse of the resulting value.

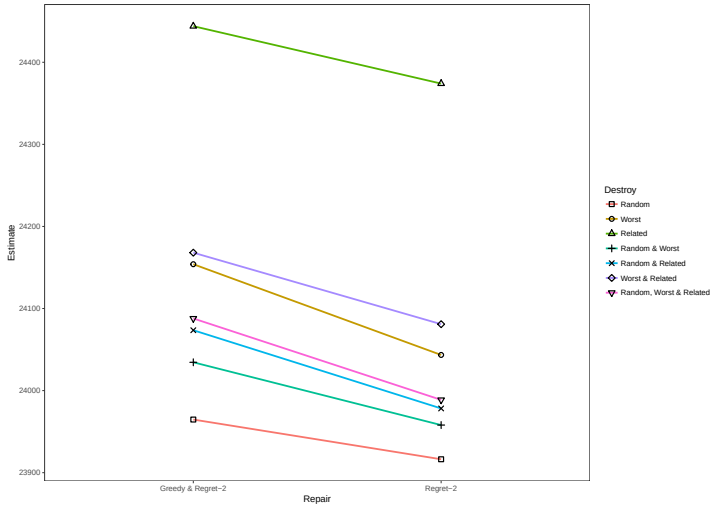
The operator variables are defined differently for this analysis compared to the fANOVA analysis. In order to enable the study of the impact of each possible combination of repair operators and of each possible combination of destroy operators on performance, dummy or binary variables are defined such that an effect estimate is obtained for each combination. So instead of one repair variable and one destroy variable, we have respectively three and seven variables in the regression model. Because of collinearity issues, not all repair and destroy dummies could be included and therefore one repair and one destroy operator configuration is chosen as a baseline. In both cases, it is the configuration including all repair or destroy operators. So the estimates for *Greedy* and *Regret2* represent the change in total cost when switching from using both repair operators (*GreedyRegret2*) to using only one of both. The estimates for *Random*, *RandomRelated*, and so on represent the change in total cost when switching from using all three destroy operators (*RandomWorstRelated*) to a configuration with less destroy operators.

Figure 3 plots the predicted objective function values for all repair and destroy operator configurations, all other variables fixed at their average value. By plotting the repair configurations on the horizontal axes we can observe the change in the predicted value when switching from one repair configuration to another one. Panel (a) displays the effect of switching to *Greedy* and shows the solution quality is expected to worsen for every destroy operator configuration it is combined with. The largest performance deterioration and overall worst result is predicted for the combination with *Worst*, while *Greedy* obtains its best performance result with *RandomRelated*. With *GreedyRegret2* best and worst performance results are expected with respectively *Random* and *Related*. The switch from *GreedyRegret2* to *Regret2* is plotted in panel (b) of Figure 3 and shows an expected performance improvement for all destroy operator combinations, but the best and worst combinations are the same as for *GreedyRegret2*. These predictions show that using regret-2 as the sole repair operator is expected to give the best performance results for all destroy operators it is combined with, while relying only on greedy repair is expected to give the worst results for all destroy operator combinations. It is also observed that the relative performance of the destroy operators per individual repair operator differs. The way a solution is destroyed has an impact on how good *Greedy* or *Regret2* is at repairing this solution. *Greedy* seems to have more difficulty in repairing a solution from which customers were removed randomly while *Regret2* is better able to cope with such a situation. *Regret2*, however, appears to find it more difficult to repair a solution from which related customers were removed — with relatedness interpreted in terms of distance as in Pisinger and Ropke [13]. These insights, confirming the analysis of Corstjens et al. [1], spark a new research challenge to discover why certain operator combinations perform (relatively) different.

The previous observations are valid for an average instance having 211 customers, 29.93 units of average demand per customer, an average service time



(a)



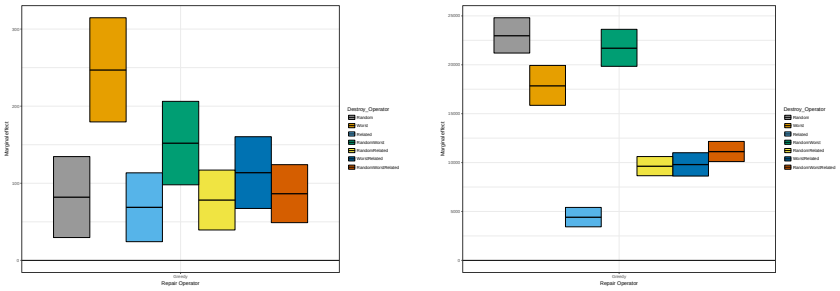
(b)

Fig. 3: Predicted total cost for an average problem instance (211 customers, 29.93 units of average demand) based on the non-linear model.

at each customer of 29.57 minutes and an average time window width of 50 minutes. Now, we investigate how these observations might be altered when the considered problem instance characteristic *Customer_Number* diverts from its average level of 211 customers. This is the aforementioned group effect — or problem instance effect in our case — on the performance impact of the repair operators. *Customer_Number* is the only problem instance characteristic that

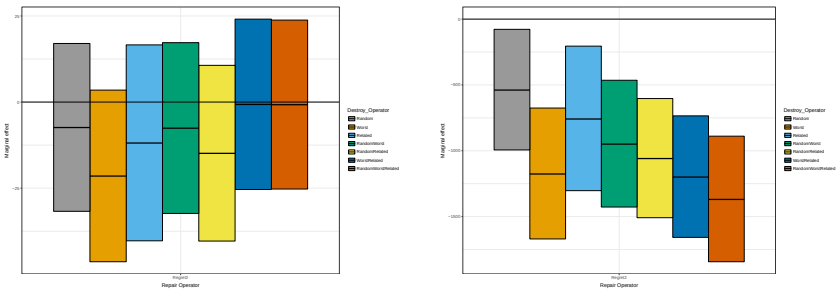
the fANOVA output indicates as having an important influence. Examination of the effect estimates in Table 3 shows that the number of customers to be served has a negative influence on the effect of switching to greedy (-16.51) and a positive influence on the effect of switching to regret-2 (2.91). This means that the more customers have to be served, the larger the performance gap with both repair operators becomes and the better regret-2 performs compared to greedy. For example, the estimated impact on performance for *Greedy* is formulated as $-133.46 - 16.51 * \Delta Customers^{1/3}$. The more customers, the more negative the estimate becomes. Besides a significant influence on the individual repair operator effects, the regression table also indicates a significant influence of *Customer_Number* on certain interaction effects. The interaction effect of *Greedy* with *Random* is negatively influenced, as indicated by the negative effect estimate -15.83 for *Greedy* $\times$ *Random* $\times$ *Customer_Number*. This influence can be interpreted as that the combination greedy and random is expected to perform increasingly worse than the combination of greedy with all destroy operators as more and more customers have to be served. The estimated impact is now expressed as $(-133.46-16.51*\Delta Customers^{1/3})+(-88.45-15.83*\Delta Customers^{1/3})$. Comparing the marginal effects of switching from both repair operators to greedy alone for the smallest (a) and largest (b) instance size in Figure 4 shows an increasing deterioration in performance when switching to *Greedy* as well as the effect with *Random* shifting away from the effect with *RandomWorstRelated*. In a similar way the significant interaction effects between *Greedy* and *Related*, *RandomWorst* and *WorstRelated* are analysed. For *Related* and *RandomWorst* additional customers strengthen the positive and negative effect these combinations have on performance. For the combination with *WorstRelated*, the negative performance impact is softened slightly when more customers have to served. There are no significant interactions effects with *Regret2*, implying that the various combinations' marginal effect is no different from the combinations with *GreedyRegret2*. Therefore, the influence of problem size on the effects with regret-2 is derived solely from the interaction term *Regret2* $\times$ *Customer_Number*. Figure 5 plots the marginal effect for the smallest and largest problem size. On the smallest problem instances, the marginal effect is insignificant indicating that there is no benefit of relying solely on regret-2 over using both repair operators. On the largest problem size, there is a clear significant improvement observed. The threshold value at which the effect turns significantly positive — i.e., where the 95% confidence interval no longer includes zero — is around 186 customers.

Similarly, the influence of *Customer_Number* on the effect of switching from *RandomWorstRelated* to any other (set of) destroy operator(s) can be investigated. The positive effects of *Random* (21.28) and *RandomWorst* (9.20) are further positively influenced when more customers have to be served, while the negative effect of *Related*(-60.46) is further negatively influenced. On the smallest problem instances the performance of the various destroy operators cannot be distinguished. Relying on worst removal alone is the only parameter setting that is expected to significantly — although barely — deteriorate the



(a) 25 customers (b) 400 customers

Fig. 4: Marginal effect of *Greedy* for a problem instance with (a) 25 customers and (b) 400 customers.

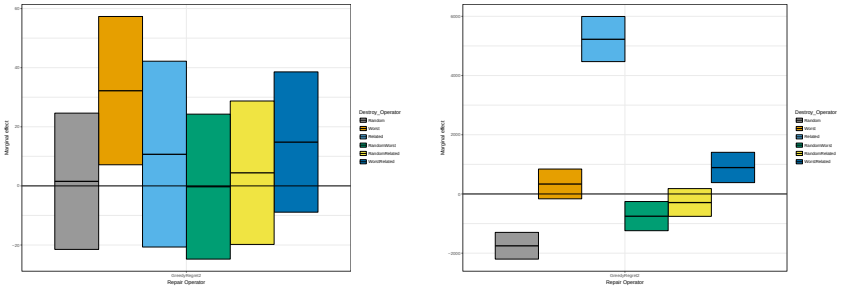


(a) 25 customers (b) 400 customers

Fig. 5: Marginal effect of *Regret-2* for a problem instance with (a) 25 customers and (b) 400 customers.

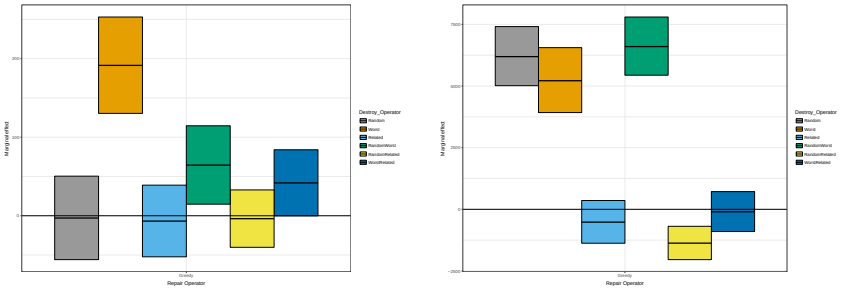
solution quality. On the larger problem instances choosing for *Random* clearly results in the largest improvement over *RandomWorstRelated*, while *Related* is expected to lead to a large deterioration in performance. Figure 6 plots the marginal effects of the destroy operators for a problem instance with (a) 25 customers and (b) 400 customers. These observations are valid for the combinations with *GreedyRegret2* and *Regret2* given that there are no significant interactions between regret-2 and the destroy operators. For the combinations with *Greedy*, there are significant interactions effects, some of which are also significantly influenced by the problem instance size. For example, there is a negative impact of switching to random removal alone when combined with greedy repair alone as deducted from the equation $21.28 - 88.45 = -67.17$. This negative impact will be softened by the positive influence of problem size on *Random*, but also intensified by the negative influence on the interaction term *Greedy* $\times$ *Random* for each additional customer above the average problem size (i.e., $(21.28 + 3.84 * \Delta Customers^{1/3}) + (-88.45 - 15.83 * \Delta Customers)$). On

the smallest problem instances the performance impact shifts to zero and becomes insignificant while on the larger problem instances the impact becomes increasingly negative. The same goes for the performance impact of *Random-Worst*, but it remains always significantly different from *RandomWorstRelated*. The impact of *Related* with *Greedy* is not significantly different from *RandomWorstRelated* with *Greedy* and this for all problem sizes. The influence of *Customer_Number* on related removal and on its interaction with greedy seem to neutralize each other. Finally, the significant influence on the interaction between *WorstRelated* and *Greedy* does not make this effect statistically different from the effect of *RandomWorstRelated* with *Greedy*, it remains at the border of significance. Figure 7 plots the marginal effects for the combination with *Greedy*.



(a) 25 customers (b) 400 customers

Fig. 6: Marginal effect of destroy operators (with *GreedyRegret2* or *Regret2*) for a problem instance with (a) 25 customers and (b) 400 customers.



(a) 25 customers (b) 400 customers

Fig. 7: Marginal effect of destroy operators (with *Greedy*) for a problem instance with (a) 25 customers and (b) 400 customers.

Summarising the observations, the regression model suggests to avoid relying only on greedy repair as it is expected to give the worst results in all considered conditions. Concerning the sole use of regret-2, distinct conclusions are drawn for smaller and larger instances due to the significant influence of the number of customers an instance has to serve. On the smallest instances, there is no significant improvement over using both repair operators observed. Furthermore, most of the performance differences between the various destroy operator configurations cannot be distinguished from each other as well, meaning the choice of destroy operators is irrelevant for these problem sizes. Only the combination with worst removal is indicated as performing significantly worse than the scenario with all destroy operators. On the larger instances, performance differences are more clear. Regret-2 performs significantly better for all destroy operator combinations, but the combination with related removal is not recommended since it leads to the smallest expected performance improvement. Random removal is the preferred combination.

Comparing the regression analysis findings to the conclusions of the fANOVA analysis, conclusions are in general consistent. On the larger problem instances, both approaches find that using regret-2 alone combined with random removal is expected to perform best. For the smaller problem instances, there is consistency on that there is no clear performance difference between using either regret-2 alone or together with greedy. Concerning the destroy operators, the regression analysis cannot identify any (combination of) destroy operator(s) as the preferred one to use since their performance cannot be distinguished from each other. The fANOVA analysis is less clear on their performance for these problem instance sizes due to the difficulty of interpreting the provided marginal plot. The regression analysis thus provides a more clear interpretation. Furthermore, the observations on the problem size influence in both analysis are deducted from different effects. In the fANOVA results the 2-way interaction between repair/destroy operator(s) and *customer_number* is considered an important effect. The multilevel regression analysis, however, also analyses the influence of problem size in the 3-way interactions between repair and destroy operators and *customer_number*, an effect that the fANOVA tool does not take into account. The observed consistencies do make the regression analysis more robust since these findings are confirmed by a methodology (fANOVA) which does not rely on the statistical assumptions of independence, normality and homoscedasticity of the error terms.

In addition, the regression model facilitates a more detailed analysis since it provides effect estimates for a particular parameter setting and problem, while the fANOVA analysis estimates marginal performance for a particular parameter value by averaging over all other parameters and problem instance characteristics. The regression results are able to identify for each combination of repair and destroy operators a problem size interval for which a significant difference in performance is expected. For example, combined with both repair operators *Random* is expected to outperform *Worst* for 134 customers or

more, *Related* for 125 customers or more, *RandomWorst* for 214 customers or more, *WorstRelated* for 166 customers or more, *RandomRelated* for 209 customers or more, and *RandomWorstRelated* for 173 customers or more.

Finally, the formulated regression model would have been different if a fANOVA analysis was not performed in advance. We would have sought to fit a multilevel regression model that included all algorithm parameters and components and all problem-level predictors since there would be no prior knowledge on which elements have an important or significant impact on performance. Furthermore, parameter interactions would be included as well. In short, this more extensive model would have to estimate substantially more effects than the simpler model based on fANOVA. A possible extensive model was fitted (see Table 6 in Appendix) to illustrate our case. First of all, the time required to fit the model is almost twice as long for the extended model (about 44 hours) compared to the simple model (about 22 hours plus one hour for the fANOVA analysis). Then, comparing the significant effects of both models, it is observed that the majority of effects are significant in both models. One effect appear no longer significant in the larger model. This can be because of collinearity issues arising when including more and more variables in a regression model, which may cause variance estimates of coefficients to be inflated thereby possibly incorrectly showing non-significance of effects. This appears to be the case for the mentioned effect when calculating its variance inflation factor (VIF)[6]. For example, the effect showing the influence of problem size on the destroy operator set random and worst removal has a VIF of 6.26 indicating that the standard error of this coefficient is more than double ($\sqrt{6.26}$) as large as it would be if it was uncorrelated with the other predictor variables. For the simpler model based on fANOVA the VIF is smaller (6.02) and therefore the standard error is less inflated. The difference in VIF is small, but because the effect is at the border of significance, the slightly higher VIF makes the effect insignificant. Furthermore, no large value changes are observed when comparing the significant effect estimates from both models. This shows that the simple model does not lack any important variable which might bias the effect estimates, an issue known in statistics as 'omitted variable bias' [15].

Studying the predictive influence of every problem instance characteristic in the extended model, it can concluded that the problem size is the most influential problem instance characteristic as changing this factor leads to large changes in the total cost values. The other problem instance characteristics show influence as well for particular operator combinations, but the performance change they bring about is of a much smaller order than is the case for varying problem size values. Therefore, fANOVA understandably denoted the problem size as the most important problem characteristic. Furthermore, the predictions for varying problem sizes are in both models almost the same so the larger model does not provide additional information that alters these predictions. However, it does provide additional information on operator behaviour for other problem instance characteristics. For example, for greedy

repair wider time windows increasingly worsens the solution quality with all destroy operators, except related removal, for which the deterioration becomes smaller. In conclusion, we believe the regression model based on fANOVA is a sufficiently detailed model that provides insight into the effects related to the largest shifts in algorithm performance.

4 Conclusion

In this paper, we have presented the complementary use of two approaches for analysing performance of heuristic algorithms with multiple parameters: fANOVA [9] and multilevel regression (MLR) [1]. The analysis results provided by fANOVA are useful when formulating a proper regression model in MLR since it leads to a more concise regression model with less input variables. MLR, on the other hand, provides a more detailed analysis of the effects of algorithm parameters. The two methodologies are applied on different data sets drawn from the same algorithm parameter and problem characteristic distributions, thereby avoiding "overfitting" analysis findings. Experimental results on a case study for a large neighbourhood search algorithm applied on instances of the vehicle routing problem with time windows have shown that the MLR can help to give additional insights on the analysis results provided by fANOVA. Moreover, due to the fact that the current fANOVA toolbox only gives main and pairwise interaction effects' results, MLR can also help to investigate higher-order interaction effects (three-way interaction in this study) on selected important variables, thus giving a better understanding on the most important effects obtained from fANOVA. We believe that this line of research would make a contribution towards the engineering cycle of developing optimization algorithms.

Meanwhile, fANOVA is available as a Python package, and the multilevel regression needs to be implemented manually. For future work, we are considering making the combination of the two approaches more ready to use, i.e., the addition and the interpretation of the multilevel regression into fANOVA should be automated.

Acknowledgements This work is funded by COMEX (Project P7/36), a BELSPO/IAP Programme. The computational resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government department EWI.

References

1. Corstjens J, Depaire B, Caris A, Sörensen K (2016) A multilevel evaluation method for heuristics with an application to the vrptw. Submitted to Operations Research

2. De Leeuw J, Meijer E, Goldstein H (2008) Handbook of multilevel analysis. Springer
3. Dietterich T (1995) Overfitting and undercomputing in machine learning. ACM computing surveys (CSUR) 27(3):326–327
4. Fawcett C, Hoos HH (2015) Analysing differences between algorithm configurations through ablation. Journal of Heuristics pp 1–28
5. Gelman A, Hill J (2006) Data Analysis Using Regression and Multilevel/Hierarchical Models. Cambridge University Press
6. Hair JF, Anderson RE, Babin BJ, Black WC (2010) Multivariate data analysis: A global perspective, vol 7. Pearson Upper Saddle River, NJ
7. Hooker G (2012) Generalized functional anova diagnostics for high-dimensional functions of dependent variables. Journal of Computational and Graphical Statistics
8. Hooker JN (1995) Testing heuristics: We have it all wrong. Journal of Heuristics 1(1):33–42
9. Hutter F, Hoos HH, Leyton-Brown K (2013) Identifying key algorithm parameters and instance features using forward selection. In: International Conference on Learning and Intelligent Optimization, Springer, pp 364–381
10. Hutter F, Hoos HH, Leyton-Brown K (2014) An efficient approach for assessing hyperparameter importance. In: International Conference on Machine Learning, pp 754–7621
11. Montgomery D (2012) Design and Analysis of Experiments, 8th Edition. John Wiley & Sons, Incorporated
12. Moore DS, McCabe GP, Craig BA (2007) Introduction to the Practice of Statistics, 6th edn. W. H. Freeman
13. Pisinger D, Ropke S (2007) A general heuristic for vehicle routing problems. Computers & Operations Research 34(8):2403–2435
14. Rardin RL, Uzsoy R (2001) Experimental Evaluation of Heuristic Optimization Algorithms: A Tutorial. Journal of Heuristics 7(3):261–304
15. Stock J, Watson MW (2011) Introduction to Econometrics. Prentice Hall, New York

Appendix

Table 4: Problem Instance Characteristics

Problem instance characteristics		
Characteristic	**Type**	**Value ranges**
number of customers	Integer	U[25, 400]
vehicle capacity	Integer	150
x/y-coordinates	Integer	U[0,500]
customer demand	Integer	U[10,50]
Service time	Integer	TRIA(min,max) min∼U[10,30] max∼U[30,50]
time window depot	Integer	Start = 0; End = 900
time window customer		
- time window centre	Integer	U[0 + travel time, 900 - travel time - service time]
- time window width	Integer	TRIA[min,max] min∼U[20,50] max∼U[50,80]
- start		Centre - 0.5*width
- end		Centre + 0.5*width
Maximum running time	Integer	TRIA(60,1800)

Table 5: Regression table

Variable	Estimate	Est.Error	l-95% CI	u-95% CI
Intercept	4,151.65	128.11	3,899.79	4,398.53
Greedy	−133.46	5.48	−144.15	−122.90
Regret2	16.98	3.66	9.78	24.18
Random	21.28	3.47	14.27	28.06
Worst	−11.32	3.69	−18.62	−3.98
Related	−60.46	4.70	−69.83	−51.33
RandomWorst	9.20	3.66	1.91	16.30
WorstRelated	−13.92	3.56	−20.97	−6.96
RandomRelated	2.50	3.56	−4.46	9.55
Customer_Number$^{1/3}$	−453.86	29.99	−513.62	−396.08
Greedy × Random	−88.45	7.90	−103.98	−72.62
Greedy × Worst	−89.67	8.87	−107.18	−72.35
Greedy × Related	68.31	6.76	55.07	81.35
Greedy × RandomWorst	−95.71	7.61	−110.70	−80.76
Greedy × WorstRelated	5.65	6.39	−6.70	18.42
Greedy × RandomRelated	15.19	5.74	3.96	26.44
Regret2 × Random	−8.77	5.06	−18.68	1.43
Regret2 × Worst	2.32	5.22	−7.96	12.43
Regret2 × Related	−5.04	5.54	−15.91	5.86
Regret2 × RandomWorst	−3.68	5.14	−13.65	6.61
Regret2 × WorstRelated	−2.14	5.01	−11.93	7.85
Regret2 × RandomRelated	−0.65	5.14	−10.71	9.39
Greedy × Customer_Number$^{1/3}$	−16.51	1.23	−18.89	−14.11
Regret2 × Customer_Number$^{1/3}$	2.91	0.81	1.35	4.49
Random × Customer_Number$^{1/3}$	3.84	0.77	2.35	5.35
Worst × Customer_Number$^{1/3}$	0.58	0.82	−1.03	2.20
Related × Customer_Number$^{1/3}$	−9.73	1.05	−11.80	−7.70
RandomWorst × Customer_Number$^{1/3}$	1.59	0.82	0.02	3.21
WorstRelated × Customer_Number$^{1/3}$	−1.26	0.80	−2.81	0.34
RandomRelated × Customer_Number$^{1/3}$	0.79	0.78	−0.73	2.31
Greedy × Random × Customer_Number$^{1/3}$	−15.83	1.76	−19.25	−12.36
Greedy × Worst × Customer_Number$^{1/3}$	−3.20	1.96	−7.03	0.60
Greedy × Related × Customer_Number$^{1/3}$	10.56	1.54	7.50	13.53
Greedy × RandomWorst × Customer_Number$^{1/3}$	−11.61	1.68	−14.87	−8.36
Greedy × WorstRelated × Customer_Number$^{1/3}$	3.13	1.45	0.25	5.94
Greedy × RandomRelated × Customer_Number$^{1/3}$	2.01	1.27	−0.49	4.48
Regret2 × Random × Customer_Number$^{1/3}$	−2.06	1.14	−4.31	0.16
Regret2 × Worst × Customer_Number$^{1/3}$	−1.25	1.16	−3.51	1.03
Regret2 × Related × Customer_Number$^{1/3}$	−1.77	1.25	−4.19	0.66
Regret2 × RandomWorst × Customer_Number$^{1/3}$	−1.19	1.15	−3.45	1.02
Regret2 × WorstRelated × Customer_Number$^{1/3}$	−0.37	1.13	−2.56	1.84
Regret2 × RandomRelated × Customer_Number$^{1/3}$	−1.24	1.14	−3.48	0.95

[a] The effects of Regret-2 & Greedy and Random, Worst & Related, the reference levels for the repair and destroy operator dummies, are accounted for in the Intercept.

Table 6: Regression table large model

Variable	Estimate	Est.Error	l-95% CI	u-95% CI
Intercept	4,155.43	127.41	3,901.56	4,402.79
Greedy	−134.86	5.28	−145.22	−124.59
Regret2	17.19	3.62	9.93	24.22
Random	20.76	3.46	13.92	27.58
Worst	−11.20	3.63	−18.36	−4.13
Related	−60.37	4.40	−68.99	−51.79
RandomWorst	9.94	3.67	2.74	17.01
WorstRelated	−13.64	3.56	−20.64	−6.73
RandomRelated	2.67	3.49	−4.17	9.56
Cooling_rate	2.14	2.24	−2.23	6.59
Start_temperature_ctrl_param	−2.45	2.11	−6.59	1.75
Noise_param	−10.51	3.41	−17.08	−3.79
Determinism_param	0.10	0.05	−0.003	0.21
Customer_Number$^{1/3}$	−460.22	29.13	−516.48	−403.65
Avg_demand	311.17	134.11	54.63	578.72
Avg_service_time	−52.64	30.16	−112.13	6.94
Avg_time_window_width	43.59	20.86	1.25	83.80
Runtime	27.54	23.19	−19.01	72.27
Greedy × Random	−84.60	7.79	−100.09	−69.34
Greedy × Worst	−86.67	8.43	−103.29	−70.35
Greedy × Related	69.73	6.78	56.50	82.87
Greedy × RandomWorst	−92.14	7.44	−106.74	−77.80
Greedy × WorstRelated	6.41	6.22	−5.56	18.75
Greedy × RandomRelated	17.86	5.66	6.87	29.00
Regret2 × Random	−9.39	5.08	−19.30	0.69
Regret2 × Worst	−0.41	5.19	−10.64	9.78
Regret2 × Related	−7.11	5.42	−17.59	3.52
Regret2 × RandomWorst	−3.29	5.18	−13.52	6.97
Regret2 × WorstRelated	−2.87	5.00	−12.66	7.21
Regret2 × RandomRelated	−1.50	5.09	−11.45	8.60
Random × Determinism_param	−0.06	0.08	−0.21	0.09
Worst × Determinism_param	−0.20	0.08	−0.36	−0.04
Related × Determinism_param	−0.48	0.08	−0.64	−0.32
RandomWorst × Determinism_param	−0.06	0.08	−0.22	0.09
WorstRelated × Determinism_param	−0.10	0.07	−0.25	0.05
RandomRelated × Determinism_param	−0.03	0.07	−0.18	0.11
Greedy × Noise_param	−22.22	5.20	−32.55	−12.00
Regret2 × Noise_param	−3.13	4.71	−12.28	6.05
Greedy × Customer_Number$^{1/3}$	−16.40	1.18	−18.72	−14.08
Greedy × Avg_demand	3.41	5.29	−6.98	13.76
Greedy × Avg_service_time	1.91	1.28	−0.59	4.40
Greedy × Avg_time_window_width	−2.30	0.82	−3.89	−0.69
Greedy × Runtime	0.41	0.99	−1.55	2.33
Regret2 × Customer_Number$^{1/3}$	3.00	0.81	1.43	4.57
Regret2 × Avg_demand	−0.59	3.86	−8.36	6.87
Regret2 × Avg_service_time	−0.86	0.89	−2.60	0.87
Regret2 × Avg_time_window_width	0.40	0.56	−0.70	1.49
Regret2 × Runtime	−0.02	0.67	−1.35	1.26
Random × Customer_Number$^{1/3}$	3.78	0.76	2.27	5.25
Random × Avg_demand	−1.20	3.42	−7.96	5.40
Random × Avg_service_time	0.29	0.85	−1.38	1.98
Random × Avg_time_window_width	−0.07	0.55	−1.15	0.99
Random × Runtime	−0.28	0.63	−1.53	0.96
Worst × Customer_Number$^{1/3}$	0.46	0.83	−1.20	2.10
Worst × Avg_demand	−0.65	3.33	−7.29	5.86
Worst × Avg_service_time	−0.24	0.90	−1.99	1.54
Worst × Avg_time_window_width	−0.45	0.57	−1.56	0.66
Worst × Runtime	0.49	0.72	−0.95	1.90
Related × Customer_Number$^{1/3}$	−9.83	1.01	−11.82	−7.85
Related × Avg_demand	−0.24	4.27	−8.67	8.15
Related × Avg_service_time	0.48	1.07	−1.64	2.59
Related × Avg_time_window_width	−2.43	0.72	−3.84	−1.02
Related × Runtime	2.64	0.82	1.02	4.21

RandomWorst × Customer_Number$^{1/3}$	1.53	0.82	−0.07	3.11
RandomWorst × Avg_demand	0.62	3.89	−7.05	8.29
RandomWorst × Avg_service_time	−0.77	0.93	−2.58	1.05
RandomWorst × Avg_time_window_width	0.36	0.57	−0.75	1.47
RandomWorst × Runtime	0.02	0.68	−1.32	1.33
WorstRelated × Customer_Number$^{1/3}$	−1.24	0.79	−2.80	0.33
WorstRelated × Avg_demand	3.96	3.25	−2.38	10.29
WorstRelated × Avg_service_time	0.55	0.86	−1.17	2.23
WorstRelated × Avg_time_window_width	−0.29	0.53	−1.29	0.74
WorstRelated × Runtime	0.70	0.69	−0.64	2.06
RandomRelated × Customer_Number$^{1/3}$	0.79	0.79	−0.77	2.33
RandomRelated × Avg_demand	−0.30	3.48	−7.10	6.43
RandomRelated × Avg_service_time	−0.26	0.85	−1.92	1.42
RandomRelated × Avg_time_window_width	−0.56	0.55	−1.63	0.51
RandomRelated × Runtime	1.00	0.67	−0.33	2.31
Cooling_rate × Customer_Number$^{1/3}$	−0.70	0.50	−1.68	0.27
Cooling_rate × Avg_demand	−0.18	2.37	−4.80	4.53
Cooling_rate × Avg_service_time	−0.29	0.56	−1.37	0.81
Cooling_rate × Avg_time_window_width	−0.72	0.36	−1.42	−0.02
Cooling_rate × Runtime	−0.83	0.42	−1.64	−0.01
Start_temperature_ctrl_param × Customer_Number$^{1/3}$	−0.54	0.48	−1.48	0.39
Start_temperature_ctrl_param × Avg_demand	0.02	2.23	−4.32	4.39
Start_temperature_ctrl_param × Avg_service_time	1.03	0.52	−0.003	2.03
Start_temperature_ctrl_param × Avg_time_window_width	−0.19	0.35	−0.86	0.49
Start_temperature_ctrl_param × Runtime	−0.17	0.40	−0.96	0.61
Determinism_param × Customer_Number$^{1/3}$	−0.0003	0.01	−0.01	0.01
Determinism_param × Avg_demand	0.02	0.02	−0.03	0.06
Determinism_param × Avg_service_time	−0.001	0.01	−0.01	0.01
Determinism_param × Avg_time_window_width	0.001	0.004	−0.01	0.01
Determinism_param × Runtime	−0.01	0.004	−0.01	0.002
Noise_param × Customer_Number$^{1/3}$	−1.20	0.50	−2.18	−0.20
Noise_param × Avg_demand	0.27	2.32	−4.28	4.83
Noise_param × Avg_service_time	−0.71	0.54	−1.78	0.36
Noise_param × Avg_time_window_width	−0.62	0.35	−1.31	0.08
Noise_param × Runtime	0.89	0.41	0.08	1.68
Greedy × Random × Customer_Number$^{1/3}$	−15.25	1.72	−18.55	−11.86
Greedy × Random × Avg_demand	−2.10	8.59	−18.77	15.04
Greedy × Random × Avg_service_time	0.61	1.93	−3.18	4.44
Greedy × Random × Avg_time_window_width	−1.30	1.19	−3.64	1.06
Greedy × Random × Runtime	3.43	1.46	0.58	6.34
Greedy × Worst × Customer_Number$^{1/3}$	−2.62	1.95	−6.39	1.23
Greedy × Worst × Avg_demand	−1.55	8.70	−18.66	15.49
Greedy × Worst × Avg_service_time	1.76	2.01	−2.20	5.68
Greedy × Worst × Avg_time_window_width	−1.45	1.35	−4.10	1.20
Greedy × Worst × Runtime	3.00	1.65	−0.28	6.24
Greedy × Related × Customer_Number$^{1/3}$	10.38	1.54	7.35	13.43
Greedy × Related × Avg_demand	10.60	7.09	−3.15	24.54
Greedy × Related × Avg_service_time	−1.34	1.64	−4.57	1.90
Greedy × Related × Avg_time_window_width	2.79	1.12	0.59	4.96
Greedy × Related × Runtime	−1.05	1.32	−3.61	1.57
Greedy × RandomWorst × Customer_Number$^{1/3}$	−10.93	1.69	−14.26	−7.62
Greedy × RandomWorst × Avg_demand	0.80	7.70	−14.09	15.89
Greedy × RandomWorst × Avg_service_time	3.09	1.81	−0.49	6.61
Greedy × RandomWorst × Avg_time_window_width	−1.52	1.14	−3.76	0.74
Greedy × RandomWorst × Runtime	2.51	1.37	−0.15	5.19
Greedy × WorstRelated × Customer_Number$^{1/3}$	2.92	1.42	0.09	5.67
Greedy × WorstRelated × Avg_demand	6.12	6.62	−6.90	19.12
Greedy × WorstRelated × Avg_service_time	−1.04	1.54	−4.10	2.03
Greedy × WorstRelated × Avg_time_window_width	0.49	0.98	−1.43	2.41
Greedy × WorstRelated × Runtime	0.50	1.24	−1.94	2.91
Greedy × RandomRelated × Customer_Number$^{1/3}$	1.86	1.25	−0.53	4.35
Greedy × RandomRelated × Avg_demand	−0.65	5.67	−11.78	10.50
Greedy × RandomRelated × Avg_service_time	0.32	1.34	−2.30	2.95
Greedy × RandomRelated × Avg_time_window_width	0.65	0.86	−1.02	2.34
Greedy × RandomRelated × Runtime	−1.33	1.08	−3.42	0.78
Regret2 × Random × Customer_Number$^{1/3}$	−2.14	1.12	−4.31	0.10

Regret2 × Random × Avg_demand	1.06	5.56	−9.82	12.02
Regret2 × Random × Avg_service_time	0.57	1.24	−1.84	3.00
Regret2 × Random × Avg_time_window_width	−0.03	0.79	−1.59	1.54
Regret2 × Random × Runtime	0.62	0.93	−1.20	2.42
Regret2 × Worst × Customer_Number$^{1/3}$	−1.80	1.17	−4.12	0.54
Regret2 × Worst × Avg_demand	2.41	5.23	−7.67	12.74
Regret2 × Worst × Avg_service_time	0.54	1.27	−1.92	3.03
Regret2 × Worst × Avg_time_window_width	−0.05	0.85	−1.72	1.62
Regret2 × Worst × Runtime	−0.59	0.97	−2.47	1.33
Regret2 × Related × Customer_Number$^{1/3}$	−1.64	1.22	−4.02	0.77
Regret2 × Related × Avg_demand	2.88	5.72	−8.37	14.13
Regret2 × Related × Avg_service_time	−0.06	1.37	−2.75	2.64
Regret2 × Related × Avg_time_window_width	0.35	0.86	−1.35	2.03
Regret2 × Related × Runtime	−1.23	0.99	−3.18	0.70
Regret2 × RandomWorst × Customer_Number$^{1/3}$	−1.22	1.15	−3.48	1.05
Regret2 × RandomWorst × Avg_demand	−1.90	5.75	−12.99	9.55
Regret2 × RandomWorst × Avg_service_time	1.72	1.27	−0.77	4.26
Regret2 × RandomWorst × Avg_time_window_width	−0.40	0.79	−1.93	1.20
Regret2 × RandomWorst × Runtime	−0.39	0.93	−2.19	1.45
Regret2 × WorstRelated × Customer_Number$^{1/3}$	−0.79	1.12	−3.00	1.42
Regret2 × WorstRelated × Avg_demand	2.40	5.22	−7.70	12.67
Regret2 × WorstRelated × Avg_service_time	−0.34	1.23	−2.73	2.10
Regret2 × WorstRelated × Avg_time_window_width	−0.03	0.78	−1.57	1.50
Regret2 × WorstRelated × Runtime	−0.53	0.92	−2.34	1.28
Regret2 × RandomRelated × Customer_Number$^{1/3}$	−1.13	1.13	−3.31	1.09
Regret2 × RandomRelated × Avg_demand	−0.50	5.51	−11.25	10.30
Regret2 × RandomRelated × Avg_service_time	0.63	1.23	−1.76	3.06
Regret2 × RandomRelated × Avg_time_window_width	0.55	0.82	−1.05	2.16
Regret2 × RandomRelated × Runtime	−0.75	0.95	−2.62	1.16