Response to the Reviewers' comments for paper "Variable neighborhood search for solving the k-domination problem" (old title: "VNS with GA-based parameter tuning for solving the k-domination problem")

Milan Predojević, Aleksandar Kartelj, and Marko Djukanović

April 18, 2023

We thank the reviewers for their comments. All comments have been carefully considered. Below we address all comments and describe the respective changes we have made to the manuscript. The reviewers' comments are quoted for clarity. All changes we made to the paper are highlighted in red.

Comments from *Reviewer #1*

> In this paper an interesting implementation of VNS algorithm is proposed. The only issue I found is missing explanation why the parameters are tuned using Genetic Algorithm and not some other package, for example IRACE? Other than that, the reported results are really outstanding.

Answer.
Thank you.
We used GA to fit into the conference framework – Evolutionary Computation, but later (after submission) we found that this was not necessary. Our standard procedure for parameter tuning is grid search or IRACE, so using GA in this situation was unfortunate. In the new version of the paper, we used the grid search instead.

Comments from *Reviewer #2*

> The main contributions of the paper are summarized in the introduction (Sec. 1). Some of them, however, are not properly supported by the experiments presented later.

1. Authors claim that the VNS significantly improves the state-of-the-art, but the comparison to the literature is not completely fair. Different from the literature approaches, the VNS uses specific parameter configurations for each instance. Besides that, the termination criteria of the literature approaches are not discussed and contrasted to those used for the VNS, and using different machines or different running time limits (for example) may impact in the quality of the results.

Answer.
In the new version of the paper, a single parameter configuration was used for all instances. The configuration was obtained by the grid search method for a random sample of 20 instances (out of 60 small to medium instances in total: 20 cities for $k \in \{1, 2, 4\}$).

The source code of BS algorithm proposed in [Corcoran and Gagarin(2021)] was not available and therefore we could not make direct comparison for the small to medium instances on the same computer. Therefore, we used the results reported in the paper and come up with a termination criteria similar to those in the cited paper. In our paper, we added a following discussion justifying the termination criteria:
"The termination criteria were chosen in a way that allows similar running times to those reported by comparison algorithms [Corcoran and Gagarin(2021)]. Namely, the best results were obtained with the BS configuration BS4, where 4 stands for the beam width. For this algorithm, in the case of $k = 2$, the average running times in seconds are 1736, 8834, 1257, 7156 and 3327 for the Bath, Belfast, Brighton, Bristol and Cardiff instances, respectively. As stated in [Corcoran and Gagarin(2021)], the comparison algorithms were also implemented in Python and executed on a desktop computer with an Intel Core i7-8700 CPU @3.20GHz. The algorithms SG, PG and BS terminate their execution as soon as the first feasible solution is reached."

Note also the following sentence, which describes the comparison approach:
"The results of SG, PG and BS for small to medium sized instances are taken from [Corcoran and Gagarin(2021)] as reported in their experimental section. The results for the five large instances were obtained by running the original implementation of PG (obtained from the authors). According to the same authors, BS was inefficient for the large instances due to its high computational complexity. Therefore, we did not include this algorithm in the large instance comparison."

2. Regarding the third conclusion, which states that "VNS is able to quickly produce solutions of reasonable quality which is not the case for approaches proposed in the literature", the paper only shows the quality of the final solutions and the running time required by the VNS to find them, but no information about the solutions found during the execution is given. It would be interesting to visualize the performance profile of some runs, identifying the best solutions

> found over time, comparing with the same visualizations for the literature approaches (and then argue that the VNS is the only approach that finds good solutions quickly). Moreover, Figures 3 and 4 show running times larger than 30 minutes (i.e., larger than the termination criterion) for most instances and values of k, which cannot be considered ”quickly” without a more informed discussion.

Answer.
This paper is accepted as a poster limited to four pages. For this reason, we are not able to provide a detailed analysis of the requested points. From the definition of the VNS algorithm and the preceding answers, it can be implicitly concluded that VNS generates many ”good” solutions in a reasonable time, which is not the case for BS4.

> According to Section 3, the proposed VNS algorithm explores neighborhoods with different sizes (number of vertices added and removed from the solution). However, such variable neighborhood strategy is used only in the shaking procedure, where a neighboring solution of a given local optima is randomly selected from one of these neighborhoods. The local search does not explore different neighborhoods. Instead, it performs a simple iterative (best) improvement procedure. Based on this, can you call this algorithm a VNS?

Answer.
The proposed VNS is well defined in the literature [Mladenović and Hansen(1997)]. The shaking step generally deals with diversification, while the local search step (as part of the VNS) is used for intensification. So, the shaking relocates the solution to its parameterized neighborhood, while local search further investigates this neighborhood thoroughly. There are several variants of VNS (see, for example, Chapter 12, devoted to VNS, in **??**). The variant we use is called Basic VNS. General VNS, for example, uses Variable neighborhood descent (VND) as the local search step, which offers a richer set of neighborhoods. In our case, Basic VNS performs well, but we might consider General VNS or Skewed VNS in a future study.

> Regarding the running time limit (30 minutes), it would be interesting to scale the time limit according to the size of instance. Figures 3 and 4 show that the required computational effort increases as instance size grows. Besides that, the paper should mention the termination criteria of the approaches from the literature and how it compares to the one used in the experimental evaluation, to ensure a fair comparison of results.

Answer.
Due to page limitations we are unable to display this scaling.

Termination criteria for comparison algorithms are explained with the sentence: ”The algorithms SG, PG and BS terminate their execution as soon as the first feasible solution is reached.”

As for the fairness of the comparison, we explained this in the answer to your first question.

> The parameter tuning step, detailed in Section 4.1, does not follow a well-established methodology for this task. First, the instances should be split into training and testing sets. The tuning is performed using the training set, and the parameter configurations are evaluated on the testing set. This avoids the so-called overtuning, producing parameter configurations whose performance is the same when solving new/unseen instances. As a consequence, we can compare the algorithm (with the produced configuration) with other approaches from the literature, since the latter use a single configuration/algorithm to solve all instances. In contrast, if specific configurations are used for each instance (as presented in the paper), the tuning time should be accounted in the comparison.

Answer.
Yes, we did it very badly, contrary to our previous practice. In the new version of the work, a single parameter configuration was used for all instances. The configuration was obtained by performing the grid search method on a training random sample of 20 instances.

> Finally, Tables 3 - 5 present the results for the VNS and the best approach from the literature. It would be better to show the complete results, i.e. for all tested algorithms.

Answer.
Due to page limitations, we are unable to provide a more detailed view. We believe this is sufficient because we have picked out the best solution to compare it with. Note that due to the reduction in size to a poster format, we now have only one table that summarizes most of the previous information.

> Minor issues:
>
> 1. More detail about the applications of the k-domination problem (lines 100 - 103) can be given.
> 2. The discussion about the results (lines 499 - 512) are repetitive, since the observations are the same for the different values of k. It could be summarized.
> 3. Figures 3 and 4 can use filling patterns for the different values of k to ease the visualization in grayscale.
> 4. The running times reported in Figures 3 and 4 are averages over how many replications?
> 5. Instead of using the experimental results reported in the literature (for SG, PG and BS), why not running them again, given that the source code is available?

6. If space is an issue to accomodate any suggestion, Tables 3, 4 and 5 can be aggregated in a single table, Figures 3 and 4 can be aggregated in a single figure, and Figures 1 and 2 can be replaced by a single table, aggregated with Table 1.

Answers.

1. Unfortunately, we do not have enough space for a more detailed description.
2. TODO
3. Unfortunately, due to space limitations on the poster, we had to remove all the figures. But the running times can now be found in Table 1.
4. Now, when running times are included in Table 1, we think it is clear that they are averages for ten runs. There is a sentence that mentions that VNS was run ten times:
   "PG and VNS algorithms are run ten times (using different random seeds) per each problem instance."
5. The source code (or executables) for BS was not available, so the results for small to medium instances were simply copied from the corresponding paper. For large instances, we ran the original algorithm PG and reported the results.
6. That's a good appetizer, thank you. We merged all the information into a single table (Table 1).

Comments from *Reviewer #3*

There is a fundamental issue with the tuning performed, which is done per instance without any intention of searching for parameter configurations that can generalize to unseen instances. Moreover, it seems the same effort was not applied to tune the other competing algorithms, which makes the comparison quite unfair. In particular, beam-search has a number of very fundamental parameters that would benefit from being tuned according to the instance size.

Answer.
You are absolutely right, it was unfortunate to fit on per instance basis. In the new version of the paper, we use a random (training) sample of instances to find a single combination of control parameters using grid search (instead of GA) – see Reviewer 1's comments.

For BS we did not have the source code (or binaries), so we could not tune it. Of course, the algorithm can be re-implemented, but we would like to leave that to future research, since our work was accepted as a poster and not a full paper.

> The approach makes even less sense in the case of parameters such as dmin and dmax that have a strong effect in constraining the set of solutions that are searched. Moreover, a wrong value of those parameters can effectively make impossible finding an infeasible solution. Thus tuning them only makes sense if one knows the actual optimal solution, which is not generally true. For unseen instances of a given size, one could calculate good values based on the properties of the graph.

Answer.
We agree, dmin and dmax have strong effect on the search trajectory. Tuning on per instance basis was very wrong idea.

> Another issue is that results on small and medium instances are taken from a different paper but it is unclear if those papers used a similar termination criterion on a similar powerful computer. Even assuming that for the small and medium instances the timelimit of 30 minutes is not reached and the VNS stops after the maximum number of 2000 iterations, it is unclear how those iterations are comparable with the termination criterion of SG, PG and BS.

Answer.
Reviewer 2 asked almost the same question (question 1), so please read the answer to that question first. Since VNS works fundamentally differently from the algorithms we compared, we had to find an exit criterion that was relatively fair. We established exit criteria that result in similar running times. Also, the computer we used for testing was downgraded in the new version of the paper – Intel i5 instead of Intel i9. The comparison algorithms were run on Intel i7.

> The paper lists as a contribution that the parameters are tuned. But properly tuning parameters is just a step of proper experimental comparison and not a scientific contribution. It would similar to saying that a contribution of the paper is to do statistical tests to assess significance.

Answer.
You are right, we no longer mention parameter tuning as a contribution.

> The tables should report variance (or std. deviation) to show the variance of across multiple runs. Without variances, it cannot be said if the performance difference is sufficiently clear or a statistical test is needed to assess significance.

Answer.
Done.

> The tables could also report running times, both for VNS and the algorithm of the literature (there is plenty of space in the paper).

Answer.
Done for the VNS. The running times of comparison algorithms are only partially reported in the literature. Also, the paper is not accepted as a full paper but as a poster with a limit of four pages. Therefore, we had to combine all the information into a single table and remove all the figures, remove the pseudo-code for local search, reduce the introduction, remove descriptions of the fast fitness assessment, etc.

> The paper claims that VNS is stopped after 30 minutes, however, figures 3 and 4 show runtimes longer than 1800 seconds

Answer.
This is explained with sentences: "The termination criteria of the VNS are: (*i*) the maximum running time of 30 minutes, and (*ii*) the maximum number of 20000 iterations. The time limit of 30 minutes is not checked during initialization (line **??** in Algorithm **??**). This means that VNS can take more than 30 minutes to finish for some very large instances, such as Dublin and Boston."

> The stacked bars in Figs 3 and 4 are difficult to understand: For example, does a run for Manchester with k=4 take 5000 seconds or 5000 - 2500? Having the bars side by side will make more sense and avoid such questions.

Answer. We think it is more appropriate now when all relevant information are included in Table 1.

> "The time limit of 30 minutes is not checked during initialization [...] This means that VNS can take more than 30 minutes to finish" -¿ However, it also means that VNS does not execute and only the initial local search has an effect. So one may wonder how much of the work reported is done by the initial local search rather than the actual VNS algorithm.

Answer.
This is a correct observation. VNS does not even start in these situations. We make this clear in the paper with a sentence:
"More precisely, in these situations, after initialization, the most important steps of the VNS (shaking and subsequent local search) are not even performed due to an expired time limit."

Since this is a poster paper now, it is by definition a work in-progress, so some aspect may be incomplete or may change in the future.

On poster papers from conference website https://gecco-2023.sigevo.org/Call-for-Papers: "Such papers should still present work in genetic and evolutionary computation that is original and new, but has not yet reached the stage of the more mature, complete research results that are typically published in full papers at GECCO. ”

> ”each processor outside the must have” -¿ the dominating set?

Answer.
Fixed.

> ”All (20) small to medium sized problem instances were tuned separately for each $k \in \{1, 2, 4\}$, resulting in a total of 60 control parameter configurations. [. . .] Parameter tuning for large instances was too inefficient. Therefore, we used reasonable manually configured parameters for these instances” -¿ This approach is fundamentally wrong since it is overfitting the parameters to each problem instance. Which parameter settings should be used for a new unseen instance of the problem? The fact that it was not possible to extrapolate the parameters found in small and medium sized instances to larger instances clearly indicates that nothing was learned from this tuning effort due to being too specific and not general enough.
>
> This is why actual parameter tuning methods encourage a clear separation between training and testing instances. See:
>
> https://doi.org/10.1613/jair.2861
>
> https://doi.org/10.1016/j.orp.2016.09.002
>
> Moreover, the parameters could be tuned per groups of similar instance size and an extrapolation done for larger instances: Franco Mascia, Mauro Birattari, and Thomas Stützle. Tuning Algorithms for Tackling Large Instances: An Experimental Protocol. In P. M. Pardalos and G. Nicosia, editors, Learning and Intelligent Optimization, 7th International Conference, LION 7, volume 7997 of Lecture Notes in Computer Science, pp. 410–422. Springer, Heidelberg, 2013.

Answer.
You are right. As for tuning on groups of similar instance size, we now consider all small and medium sized instances as one group. This might be handled differently in the future version of the paper once it is fully mature. Larger instances are currently solved with the same parameters, but wee will certainly consider this idea of extrapolating parameters in future research.

> The datasets and codes could have been made available for review via the supplementary material or an anonymous Zenodo upload.

Answer.
Zenodo was not known to us, thank you for this suggestion. Now all materials are available via the public GitHub repo: `https://github.com/mikiMilan/k-domination`

Comments from *Reviewer #4*

> The paper proposes an efficient search metaheuristic for solving the k-domination problem.
>
> It describes the motivation and method clearly. The experiment description, including parameter tuning, is detailed and could thus be likely reproduced from the paper.
>
> The algorithmic components are motivated well (both in terms of efficiency and solution quality) and demonstrably improve over the chosen baselines empirically. While I feel that I cannot judge whether the baselines and benchmark are appropriate for the evaluation, the results are convincing to me. I therefore recommend accepting the paper.

Answer.
Thank you.

References

[Corcoran and Gagarin(2021)] Padraig Corcoran and Andrei Gagarin. 2021. Heuristics for k-domination models of facility location problems in street networks. *Computers & Operations Research* 133 (2021), 105368.

[Mladenović and Hansen(1997)] Nenad Mladenović and Pierre Hansen. 1997. Variable neighborhood search. *Computers & operations research* 24, 11 (1997), 1097–1100.