Roman Václavík - Algorithms for Personnel Scheduling Enhanced by Machine Learning Techniques

1 Overall evaluation and recommendation

The thesis investigates an interesting subject: how machine learning techniques can be used to improve algorithms for combinatorial optimization problems. A key idea is to use machine learning to extract knowledge from intermediate data instead of throwing it away. The area combining operations research techniques and machine learning techniques appears to be booming these years, probably due to the recent successes of machine learning and artificial intelligence. In this way, the thesis of Roman Václavík is one of the forerunners in a developing field. This is something he should be congratulated for.

The core part of the thesis consists of three scientific papers that all have been accepted and published in well-renowned, peer-reviewed academic journals. This speaks for the quality of the work. Based on the information at the back of the thesis Roman Václavík has contributed equally to all of the papers and is first author on two of them. Below I comment in detail about each of the three papers. The comments also contain some suggestions for improvements. I realize that it is not possible to implement these suggestions (as papers are already published and the thesis is in its final form), but I hope that the some of the comment perhaps could be useful for future work.

In general, the thesis is concise and well-written. It proposes novel techniques and tests these techniques experimentally. Some of the techniques have the potential to influence future research. In my mind, there is no doubt that the author of the thesis proved to have an ability to perform research and to achieve scientific results. I do recommend the thesis for presentation with the aim of receiving a Ph.D. degree.

1.1 Specific questions

The instruction letter asks me to comment on the following questions:

to what extent the subject of the thesis is relevant to the current needs of the scientific community The part of the Operations Research community that work with combinatorial optimization problems have, among other, two main tools available: heuristics and exact methods. The thesis presents ideas that potentially can speed up classes of heuristic and exact methods. Furthermore, the thesis couples Operations Research methods with machine learning methods, something that is gaining popularity currently so the subject is very well timed.

to what extent the main objectives of the work have been fulfilled The five objectives listed on page 5 of the thesis have been fulfilled. Objective 5 has perhaps only been partially fulfilled since opportunities to compare the proposed heuristic to state-of-the-art heuristics have been missed out in chapter 2.

to what extent the methods used in the thesis are appropriate The methods used are appropriate.

what the main results and contributions of the work are The main results and contributions of the paper are listed on page 15 and 16 of the thesis and I agree with the claims made here. If I should mention a single contribution (and probably repeating myself), then the main contribution is the successful combination of Operations Research and machine learning methods.

to what extent the work is important for the further development of science, As already mentioned, the thesis explores an area that currently is attracting more and more attention. The thesis can be seen as early work in the area, and especially the methods from chapter 2 and 3 have the potential for inspiring future research (in my opinion).

whether the thesis satisfies the conditions of a creative scientific work. The thesis proposes several novel techniques. I would clearly classify this as creative scientific work

2 Chapter 1, Introduction

The introduction to the thesis is rather short but gives a good overview of the objectives and contributions of the thesis. On page 7 and 8 it is argued that many approaches throw away intermediate data. I think that is quite dependent on the algorithm, many algorithms keep at least part of the intermediate data:

- Column generation algorithms keep (some of) the columns already generated and that helps us when solving the master problem (some old columns may re-enter the basis).
- Some heuristic algorithms keep a database of already observed solutions and try to combine the existing solutions together to form new solutions. For example, for the vehicle routing problem, some heuristics keep a

database of observed routes and repeatedly solves a set-partitioning problem that can combine the old routes in new ways in order to construct new solutions.

- Genetic algorithms keep part of the past solutions and use those to construct new solutions.
- Branch-and-cut algorithms use cuts generated in one node and use them in the following nodes processed (either in descendant nodes only or in the entire tree, based on the properties of the cut).

3 Chapter 2, Roster evaluation based on classifiers for the nurse rostering problem

This chapter investigates if it is possible to speed up a metaheuristic by speeding up the evaluation of changes to a solution of the nurse rostering problem. In order to do so, machine learning is used to evaluate the impact of changes to a solution. To the best of my knowledge, this is a quite novel idea and little related work exists. The authors investigate different machine learning techniques (e.g. neural networks, decision trees, and logistic regression) and ways of combining the result of simple machine learning algorithms in order to get a better classification. Neural networks clearly provide the best results, especially if improved by boosting algorithms that combine the output from several simpler networks (but this improvement comes at a cost of increased running time).

Results show that a significant speedup can be achieved but at the cost of sacrificing solution quality compared to an ordinary evaluation of solutions. The paper illustrates that the approach is promising, but the experimental setup leaves one a bit in doubt about the conclusions (see detailed comments below). I would have expected to see a comparison of results and running time to other heuristics (proposed in the literature) for the nurse rostering problem.

The chapter enters an interesting area that combines OR and machine learning techniques. This research area seems to be gathering momentum currently and in that way the chapter can be an inspiration for several future studies. As future work it could be interesting to use a similar approach for a problem where evaluating the changes to a solution is much harder, problems exist where evaluating the feasibility after a change to the solution is NP-complete (example: 2D bin-packing)

3.1 Detailed comments

- C2, page 667, abstract: *Due to the complexity of this problem and the size of the real-world instances, it is not possible to use exact methods, and thus heuristics, meta-heuristics, or hyper-heuristics must be employed.* Exact methods are very quickly dismissed. I would have a bit more cautious with the wording here.

- C2. page 674, figure 3. The pseudo code should have been explained better and perhaps it could have been written more compactly. Examples of issues that are unclear to me in the pseudo code:
  - line 3, stopping criterion: It is not clear to me what stopping criterion is used in the algorithms. Examples of stopping criteria are mentioned on page 673, but which one is employed in the implementation?
  - How exactly is the tabu list used? What exactly is made tabu? The solution just visited? The reverse move? Something else? The statements in line 4 and 30 to 36 are not clear to me.
- C2. page 677. Why not have vectors corresponding to both changed rosters as input to the network (instead) of having two separate networks and then trying to determine a combined output from these.
- C2. page 678. Has P been defined?
- C2. Page 681: Why only a subset of instances, many more are available? The reader may think that the instances that support the desired conclusion have been selected.
- C2. Page 681. Was the machine learning algorithms trained on each specific instance? Shouldn't the ML algorithms be trained on one set of instances and then be tested on another set? I understand that you have split your set of input data into training data and test data, but if the training data is for a specific instance it seems like the ML algorithm gets targeted to "know" that specific instance. Is training time included in the time for ML-based evaluations?
- C2. Page 682, table 2 and several of the following tables. Representing the absolute difference in the objective function is not so informative. Also, tell the reader what the objective value obtained is or report percent wise deterioration of objective function.
- C2. page 685. ***To measure the results as accurately as possible, we only counted the total time required by the given evaluation methods***: If I understand this correctly it is not the entire heuristic that is timed, but only evaluations. That does not seem entirely fair to me.

4 Chapter 3, Accelerating the Branch-and-Price Algorithm Using Machine Learning

This chapter studies if it is possible to accelerate a branch-and-price (BAP) algorithm by using machine learning techniques. The key idea is that if the pricing problem is being solved by a branch-and-bound based method then it is sometimes possible to speed up the solution time of the pricing problem by providing tight upper bounds (in the case of minimization). Very similar pricing problems are solved repeatedly with the only difference between iterations being that the objective of the pricing problem changes. The objective of the pricing problem is controlled by the dual variables obtained after solving the master problem. Therefore it is entirely the dual variables that "decides" the objective of the pricing problem and the hypothesis is that an upper bound for the pricing problem can be predicted based on the values of the dual variables.

A simple linear function is used to predict the value of the pricing problem objective and the machine learning component learns the parameters of the function. A specially designed loss function is used in order to encourage the model to overestimate the objective function rather than underestimate it (since an underestimate in principle means that the pricing problem must be resolved). Two applications are studied: Nurse rostering (also studied in chapter 2) and Time division multiplexing. Results are best for the Nurse rostering problem where a speedup of 40% is achieved.

The paper is interesting and presents a novel idea, that I have not encountered before. The results show that a speedup is possible based on the idea and the idea is quite generic and could potentially be used for other branch-and-price algorithms. Having said that, there are also other ways of speeding up the solution of a pricing problem, the most obvious is by solving the pricing problem using a customized heuristic. The heuristic can be used as long as it is able to produce columns with negative reduced cost (the authors also mention this speed-up method). It would be interesting to compare the speed-up obtained by the machine learning method to that obtained by a heuristic tailored to the specific pricing problem. If one has to implement the machine learning from scratch it seems like it would be just as easy to implement a simple greedy heuristic or a local search method for the pricing problem. Of course, the machine learning method can be re-used for new problems, so in that sense, the machine learning algorithm is more "economic".

Regarding the choice of machine learning algorithm, I am wondering (after looking at Figure 4 and Figure 5) if a moving average increased by a few percents would work almost as well as the more complex machine learning model?

All-in-all I find the chapter interesting and well written and it fully deserves its publication in EJOR.

4.1 Detailed comments

- C3, page 1057: *The RMP is usually solved by a LP solver; and columns are added gradually in each iteration of the column generation. Because the LP solver uses the previous result to follow up with the current result, the computation time is minimized, and thus potential time reductions would be negligible.* For some problems it is the LP solver that is spending most of the time, see e.g.[1]. In that paper, there is an example where the LP takes more than 90% of the time (see table 6 in [1])

- C3, page 1058: constraint (1). Normally $c(x)$ is not complicated, it is just a linear function. It is typically the constraints of the pricing problem that makes it difficult to solve (the constraints are not shown in the equation). You can integrate the constraints into $c(x)$ by setting $c(x) = \infty$ when x is not a feasible solution, but that is not explained, so I guess that is not the intention.

- C3, page 1058: Is it important to have the discount factor? Did you experiment with not having this? The old data points are just as valid as the new ones (as long as they are for the same pricing problem (i.e. same nurse) and the pricing has not been modified by branching decisions. Is the logarithmic discount factor actually working a bit like the moving average idea mentioned in the overall comments?

- C3, page 1063. Regarding time-division multiplexing. It is not clear to me when is this problem is being solved in real life? In real-time as the applications are running? If so, I guess we have almost no time to solve the problem. In an offline way? If so, how do we know about which applications will be running?

5 Chapter 4, Adaptive online scheduling of tasks with anytime property on heterogeneous resources

This chapter studies a scheduling problem where clients submit computing tasks to a central server that allocates the tasks to heterogeneous computing resources. The computing tasks have the ***anytime property*** meaning that even if the computation is stopped early a solution will be available but it will not have the same quality as if the task were allowed to finish. It is assumed that solution quality increases when more time is allocated to the task. Solution methods for optimization problems often have this property. The solution quality obtained at a give time-expenditure is given by a processing time curve (could also be called quality curve) with time on one axis and a quality measure on the other axis.

The workload of the client-server system is not evenly spread and at times the workload is so high that the computing resources are not enough in order for the tasks to finish before their deadline if the full running time of the tasks is enforced. In this case, the system can choose to lower the target quality of some of the tasks in order for the tasks to finish within their time limit. The problem is complicated by several (realistic) issues

- Computing resources are heterogeneous and therefore a task may take a longer time to execute on one computing resource compared to another.
- All tasks are not known in advance but appear over time.
- For a given incoming task the processing time curve is not known in advance but must be estimated based on features of the task.

The machine learning component of the chapter relates to the last bullet point. The processing time curve is approximated by a piecewise linear curve and the ML algorithm should be able to predict the value at certain points based on the feature of the task. Two machine learning algorithms are attempted: k-nearest neighbor and regression trees and of these k-nearest neighbor is deemed the best.

With respect to scheduling and quality control (choosing the target quality based on expected load) rather simple algorithms are presented [simple is not meant in a bad way] and results show that the overall system is able to schedule task and control target quality in a way such that deadlines are met in most cases while suffering some quality loss (which is inevitable).

The chapter is well-written and the proposed method seems well thought-out and not very far from something that could be used in a real-life setting (assuming that the tasks that are presented to the system all are of a similar type such that their processing time curves can be estimated).

6 Conclusion

The instruction letter states that the review should end by the following sentences (the also appear in the overall evaluation and recommendation): **The author of the thesis proved to have an ability to perform research and to achieve scientific results. I do recommend the thesis for presentation with the aim of receiving a Ph.D. degree.**

Professor Stefan Røpke - Technical University of Denmark

28/2 - 2019
Date

Signature

References

[1] Issmail Elhallaoui, Daniel Villeneuve, François Soumis, and Guy Desaulniers. Dynamic aggregation of set-partitioning constraints in column generation. *Operations Research*, 53(4):632–645, 2005.