Towards prediction of algorithm performance in real world optimisation problems

Tommy Messelis [a] Stefaan Haspeslagh [a] Burak Bilgin [b]

Patrick De Causmaecker [a] Greet Vanden Berghe [b]

[a] *K.U.Leuven Campus Kortrijk, E. Sabbelaan 53, B-8500 Kortrijk*
[b] *KaHo Sint Lieven, Gebr. Desmetstraat 1, B-9000 Gent*

Abstract

We investigate the applicability of an existing framework for algorithm runtime prediction to the field of metaheuristics, in particular applied to the real world problem of nurse rostering. Apart from predicting the runtime, we look at other performance criteria as well. These so called empirical hardness models are based on readily computable features of the problem instances. These problem features are basic properties or characteristics that are thought to be influencing the complexity of the problem instances. We follow two approaches, one in a domain specific setting, and later in a more general setting where problems are represented using a Propositional Satisfiability (SAT) formulation. Both approaches lead to accurate prediction models in a small proof-of-concept problem distribution. The framework can be used to help understanding the complexity, build algorithm portfolios and allow for quick quality approximation when the resources for computing a solution are not available.

1 Hardness analysis

In many problems encountered in computer science, some instances appear to be harder than others, when solved with a particular solution method. From another point of view, the same instance can appear much harder for one algorithm than for another. In other words, there is no single best algorithm for all instances of a given problem instance distribution. The key in finding optimal solutions, as fast as possible, is selecting the right solution method for the given instance. This is the starting point for algorithm portfolios. An important concept is the notion of empirical hardness. The hardness is not always intrinsic to the problem instance but is more often related to the combination of an instance and a solution method. Empirical hardness thus means the complexity of a problem instance when solved with a particular solution method, measured by some performance criteria.

Empirical hardness models try to model algorithm behaviour, based on features of the problem instances. These features are efficiently computable properties of the instances. The choice of these features is of high importance and determines the success of the model. Coming up with a set of features is often a difficult task, one usually needs expert insight into the problem domain to identify certain properties that are thought to influence the empirical hardness of the instances. Leyton-Brown et al. in [5] introduce a general framework for predicting the runtime of an algorithm on a specific problem instance:

1. Identify the problem instance distribution.
2. Select one or more algorithms.
3. Select a set of inexpensive, distribution independent features.
4. Sample the instance distribution to generate a training set of problem instances. For each instance, run all the algorithms and determine the runtime. Compute all features for all instances.
5. Eliminate redundant or uninformative features.

6. Use machine learning techniques to select functions of features that predict the runtimes of all algorithms.

It is demonstrated that this framework is successful for the Winner Determination Problem within combinatorial auctions [5]. The same authors also applied it to the Propositional Satisfiability Problem, successfully building an algorithm portfolio that already won several SAT competitions [8]. The authors stress the importance of the instances used for learning to be sufficiently similar to the instances for which they want to predict the algorithm runtime in the end.

Our aim is to generalize this framework to predict other performance criteria (not only runtime), for real world problems where complete solution methods often become infeasible and metaheuristics that offer a good enough solution have to be used.

2 Case: Nurse rostering

We will demonstrate the feasibility of performance prediction in real world scheduling and timetabling problems by applying it to the Nurse Rostering Problem (NRP). The problem is by nature a highly constrained problem and very hard to solve. A set of shifts needs to be assigned to nurses with the appropriate skill while several constraints (either hard or soft) need to be taken into account. The nurse rostering problem instance corresponds to a set of variables and a formula (i.e. a conjunction of clauses) on these variables. Generating SAT features from a complex real world setting as the NRP has, as far as we are aware, not been undertaken.

The nurse rostering problem can be categorized according to the $\alpha|\beta|\gamma$ notation that is presented in [3]. α denotes the resources (personnel environment), β the organisation of the work, and γ the optimisation requirements. This covers a wide range of variations according to the differences in the characteristics of these elements. The problem instances can involve nurses with a single skill, multiple skills, or individual skills. Several constraints can apply to the personnel environment such as limited availability of the nurses, or restrictions on the sequences of assignments to individual nurses. Work characteristics are determined by the coverage constraints and shift types. The nurse rostering problems can involve a limited number of shift types, or multiple shift types, which can be overlapping as well. Coverage constraints can be given as a fixed value or as a range. They can also apply to time intervals. Coverage constraints can be fluctuating over the schedule horizon. Optimisation objectives of the nurse rostering problems can involve personal and coverage constraints, as well as the number of nurses. The problems can be defined as exact, optimisation, and multi-objective optimization problems.

The NRP has been the subject of intensive study and an overview of the state of the art is presented by Burke et al. [2]. Some real world benchmark problems can be found in [4].

3 Experimental setup

We generalise and apply the proposed strategy (see Section 1) and build empirical hardness models for a set of algorithms for the NRP. We are interested both in approximate solutions and in the optimal solution. Therefore we initially work with a limited planning horizon so that a complete solver can still find the optimal solution in a limited amount of time. Later on we intend to do the same for larger real world instances.

Instance distribution. The first step in the strategy is to define the problem instance distribution. We focus on the $SA|DV\{2,3\}|P$ problems in the categorisation of [3]. We consider sequence constraints (S) and availabilities (A) with a determined (D), fluctuating (V) coverage in a 2- or 3-shift structure. The objective is to satisfy personnel constraints (P). Coverage is considered a hard constraint (can not be violated) while sequence constraints and availabilities can be violated at a certain cost. We limit the planning horizon to 14 days and 6 nurses are available. Sequence constraints, coverage and availabilities vary.

Algorithm selection. The second step is to select one or more algorithms. In order to find optimal solutions, we designed in integer program representation. The commercial software package CPLEX contains a state of the art solver for such integer programs. We let this solver compute the optimal solutions. CPLEX execution time and the optimal value for the objective functions are the performance characteristics that we will try to predict.

We also use a variable neighbourhood search algorithm to find quick approximate solutions [1]. The framework is generalized in the sense that we specify which performance criteria should be considered. We look at:

- CPLEX runtime
- quality of the optimal solution (the accumulated costs associated to the constraint violations of the optimal solution (positive integer))
- quality of the approximate solution obtained by the metaheuristic
- the optimality gap (measuring the difference between the metaheuristic result and the optimum)

Feature set. This third step is highly important for the success of the strategy. Good, valuable features that influence the performance have to be found. We follow two different approaches here. We look at the specific domain of nurse rostering and develop a set of features which intuitively relate to the empirical complexity of the instances. The second approach is to develop an efficient translation scheme and to represent the NRP instances as SAT instances. Translating NRP constraints into SAT clauses is an important idea in this procedure. It allows using an extensive set of SAT features, already proven successful in runtime prediction for complete SAT solvers. Doing so, we hope to capture hidden problem structures that can not be easily represented using the domain specific features. Details of the translation can be found in [7].
The domain specific set contains 6 features expressing structural properties of the problem instances:

- maximum and minimum number of assignments
- maximum and minimum number of consecutive working days
- maximum and minimum number of consecutive free days

We add ratios of these features, somehow representing the hardness of feasibility and tightness of the constraints:

- hardness: availability / coverage
- tightness: max / min ratios (the smaller this ratio, the less freedom for solving)

After translating the instance to SAT, we can use (a subset of) an existing feature set [6]. This set contains features in the following groups:

- problem size: number of clauses, number of variables, clauses-to-variables ratio, ...
- problem structure: different graph representations of the instances lead to various node and edge degree statistics
- balance based: the fraction of positive literals per clause, fraction of positive occurrences of variables in clauses, ...
- proximity to Horn formulae: measuring how similar the formula is to a Horn formula (Horn formulae form a special class of SAT formulae that are easier to solve)

Sampling and measuring. In this phase we sample the problem instance distribution to generate a training set of instances. We run both algorithms and determine the performance criteria. All feature values are calculated.

Feature elimination. Some features are uninformative (no distinction in the values), other features are strongly correlated. We reduce the set of features until we have a set with only informative, non-correlated features.

Model learning. The models are built using statistical regression techniques, based on the good results obtained by [8]. Other machine learning techniques are also possible.

4 Results

The results of the experiments are twofold. On the one hand we have the models based on the domain specific feature set and on the other hand the models based on the SAT features.

We generated a set of 500 training instances, sampling from the defined distribution, and calculated all feature values for both the NRP features and the SAT features. As it turned out, the NRP feature set did not contain strongly correlated or uninformative features, so this set did not have to be reduced. The SAT feature set however did contain some correlated and univalued features. After elimination, 10 out of the 37 features are left:

- clauses-to-variables ratio
- variable-clause graph. This bipartite graph has on the one side a node for every variable, and on the other side a node for every clause in the problem. There are edges between clause nodes and variable nodes when the variable is contained in the clause.
  - clause node degree maximum over all clause nodes
  - clause node degree minimum over all clause nodes
  - clause node degree variation over all clause nodes
- clause graph. This graph representation contains a node for every clause and an edge between two nodes means that both clauses contain at least one common variable.
  - node degree mean over all nodes
- balance
  - the fraction of positive literals per clause, variation over all clauses
  - the portion of unary clauses in the SAT formulation
  - the portion of ternary clauses in the SAT formulation
- proximity to Horn formulae
  - the number of occurrences of a variable in a Horn clause, mean over all variables
  - the number of occurrences of a variable in a Horn clause, variation over all variables

The feature set was developed for runtime prediction for abstract problems in SAT competitions. We demonstrate that the same features can be used for other predictions on instance distributions generated from NRP instances.

We use multivariate linear regression to learn mappings from the feature space onto the desired performance criteria, minimizing the root mean squared error on the learning set. The regression results are shown in the following subsections.

4.1 CPLEX runtime

The linear regression models for the CPLEX runtime are not very accurate. This is due to very high fluctuations in the runtime. Most instances in the training set were solved very quickly, while a small amount of instances took several orders of magnitude more time. The linear models did not fit this data very well. Using the logarithm of the runtime did not result in higher accuracy.

Based on these observations, we can argue that we do not really need a predictor for the exact runtime. Instead it can be more useful to have a classifier that predicts whether an instance can be solved very quickly (less than one second) or rather slow (more than a few hundred seconds). We did not perform any experiments with this kind of classifiers (but we plan to do this in the future).

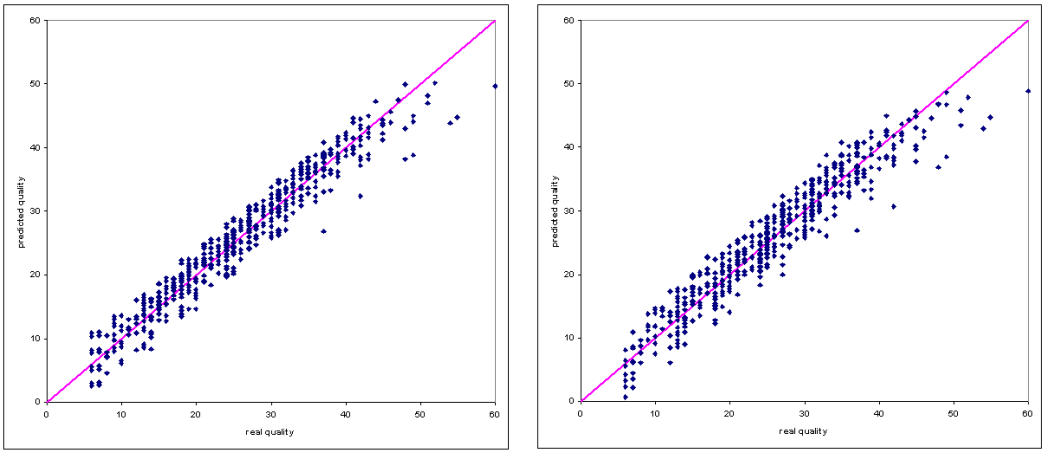
4.2 quality optimal solution

The linear regression models that are built to predict the solution quality of the optimal solutions are very accurate. The model based on NRP features has a correlation coefficient R^2 of 0.94. The model based on the SAT feature set does even better with an R^2 of 0.99.

Figure 1 shows a plot of the real optimal objective function values versus the predicted values of the training instances, (a) for the model based on NRP features, and (b) for the model based on SAT features.

We can see that both models fit the training set very well. The models slightly underestimate the accumulated penalty for instances where the optimal value is rather high.

Figure 1: Optimal objective value versus predicted optimal value.



(a) based on the NRP feature set

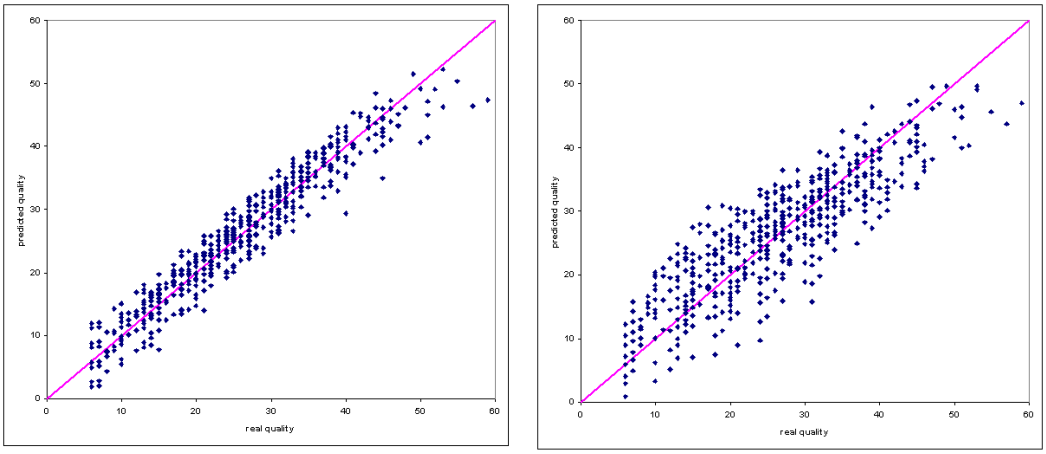
(b) based on the SAT feature set

4.3 quality approximate solution

The next performance criterion we are interested in is the quality of an approximate solution, obtained by a metaheuristic. The models that map the feature space onto this criterion are also quite accurate. There is some loss in accuracy, but this is because the metaheuristic is not always able to get the maximum out of an instance. For many instances of the given distribution the metaheuristic does reach the optimal solution, for a much smaller amount of instances this optimal value is only approximated. This can be thought of as noise in the data. It leads to less accurate models than when exact information is available.

The correlation coefficient R^2 of the model based on NRP features is 0.93. The model based on SAT instances achieves a slightly larger R^2 of 0.96. Figure 2 shows a plot of the actual metaheuristic objective function values versus the predicted values, (a) for the model based on NRP features, and (b) for the model based on SAT features.

Figure 2: Metaheuristic obtained objective value versus predicted metaheuristic value.



(a) based on the NRP feature set

(b) based on the SAT feature set

4.4 quality gap

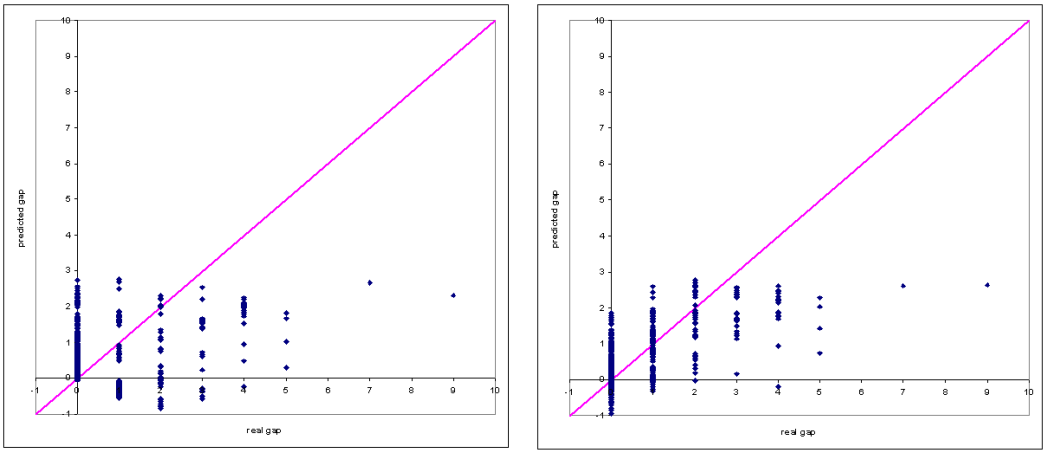
Predicting the quality gap based on the feature sets did not work so well. The correlation coefficients R^2 are only 0.40 and 0.45 for the models based on NRP features and SAT features respectively. Figure 3 shows a

plot of the predicted quality gap versus the real quality gap, (a) for the model based on NRP features, and (b) for the model based on SAT features.

It might seem strange that the gap is not well predicted while both the real optimal value and the approximation can be predicted accurately. However in many cases the quality gap is 0 (meaning that the metaheuristic found the optimal solution) and only in 5% of the training instances the metaheuristic value is more than a value 3 away from the optimal value. Considering that the quality values in the dataset lay between 6 and 60, we can argue that the metaheuristic approximates the optimal value very well. The quality gap itself is thus of less importance for this metaheuristic on this set. There is never a big difference between the optimal value and the approximation.

On the other hand, if we can not find a clear relationship between problem characteristics (features) and the quality gap for a given metaheuristic, the cause for this gap lies probably not in the instance itself. Random choices in the search process will probably have caused the metaheuristic not to find the optimum. The quality gap is then merely a result of random noise in the search process.

Figure 3: Real quality gap versus predicted quality gap.



(a) based on the NRP feature set

(b) based on the SAT feature set

5 Conclusions & further research

Overall we can conclude that the presented method is suitable for real world problems. Other performance criteria than runtime can be modelled.

For this problem instance distribution specifically, we can indeed learn accurate models that predict solution quality for both a complete solver (optimal solution) and a local search metaheuristic (approximate solution). The quality gap (the difference between an approximate solution and the optimal solution) was not so well modelled. We argue that this is a side effect of the quality of the metaheuristic. Because the metaheuristic approximates the optimal value very well, the quality gap is merely a consequence of random noise in the search process. The models that predict CPLEX runtime however were not at all accurate. This is due to very high fluctuations in the runtimes. Most of the instances in the training set were quickly solved by the complete solver (less than one second), while just a small number of instances consumed considerably more time when solved to optimality. We argue that in this case, it is of better use to have a classifier that predicts whether an instance can be solved quickly or not. In practice, for larger instances, we may assume that complete solvers will become infeasible. Most metaheuristics have a predefined stopping criterion, so prediction of runtime is not really necessary. In real world problems we are mainly interested in the quality of solutions.

The translation to SAT instances has also proven successful. The feature set that was originally developed to be used for runtime prediction of complete SAT solvers is also useful for modelling the performance of optimisation algorithms in a very different problem domain. We observe that the models learnt on SAT features even yield a higher accurary. This is partly a consequence of the size and quality of the NRP feature set. Only very basic properties and some aggregate functions of them are considered. The SAT feature set is

much more elaborate. But not only the feature set is of importance. The idea of translating nurse rostering instances into a general representation allows for a more abstract way of thinking about the problem. Representing the instances as SAT instances allows for a better exploration of hidden structures and not so easily expressed properties. Working with this abstract representation and a feature set that really looks into the deep structure of the instances almost naturally leads to better models. We want to advocate the usefulness of such a translation scheme for practical, real world problems. Generally applicable, efficient translation techniques may prove important in practical usage.

In the future we plan to apply this to real world problems. We expect to enable the prediction of the solution quality obtained by a metaheuristic. We plan to explore the benchmark nurse rostering instances of [4] with multiple solution methods, and as a result designing an algorithm portfolio. But the focus will not only be on nurse rostering. We aim at applying these results to other personnel scheduling and timetabling problems and eventually all kinds of constraint optimisation problems.

References

[1] Edmund K. Burke, Timothy Curtois, Gerhard Post, Rong Qu, and Bart Veltman. A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem. *European Journal of Operational Research*, 188(2):330–341, July 2008.

[2] Edmund K. Burke, Patrick De Causmaecker, Greet Vanden Berghe, and Hendrik Van Landeghem. The state of the art of nurse rostering. *J. of Scheduling*, 7(6):441–499, 2004.

[3] Patrick De Causmaecker and Greet Vanden Berghe. Categorisation of personnel rostering problems. Technical report, K.U.Leuven - KaHo Sint-Lieven, 2009.

[4] Tim Curtois. Nurse rostering problems and benchmarks, http://www.cs.nott.ac.uk/∼tec/nrp/.

[5] Kevin Leyton-Brown, Eugene Nudelman, and Yoav Shoham. Learning the empirical hardness of optimization problems: The case of combinatorial auctions. In *CP*, pages 556–572, 2002.

[6] Eugene Nudelman, Alex Devkar, Yoav Shoham, and Kevin Leyton-Brown. Understanding random sat: Beyond the clauses-to-variables ratio. In *In Proc. of CP-04*, pages 438–452.

[7] Burak Bilgin Patrick De Causmaecker Stefaan Haspeslagh, Tommy Messelis and Greet Vanden Berghe. Using a sat formulation to derive instance features for predicting algorithm performance in real world optimisation problems. Technical report, K.U.Leuven - KaHo Sint-Lieven, 2009.

[8] Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Satzilla: Portfolio-based algorithm selection for sat.