Evolutionary Computation in Constraint Satisfaction and Machine Learning

Jano I. van Hemert

Leiden Institute of Advanced Computer Science, Leiden University
jvhemert@cs.leidenuniv.nl

1 Introduction

At first sight the two problem areas constraint satisfaction and machine learning do not appear to be similar. The first has a clear definition of its problem domain and a crisp definition of solutions. The second is a much broader defined problem domain, which leads to many objectives to be solved.

Many research areas have focused there attention on constraint satisfaction, operating research, ant colonies, evolutionary computation and, most notable, constraint programming. Although the problems that are being studied share the same goal, which is to satisfy a set of constraints, their precise definition varies.

Among these problems we find numerous well known ones such as, k-graph colouring, 3-SAT and n-queens. For any of these problems we can transform them to a binary constraint satisfaction problem without loss of generality. This holds for any finite constraint satisfaction problem, and it means that we can solve a problem by solving a corresponding binary representation [14], which means that every constraint only restricts the assignment of values to two variables. In practice, however, we need to be careful as different forms of transformation are possible, each with its own impact on the performance of the applied solving algorithm.

In contrast to the field of constraint satisfaction the field of machine learning tries to cope with many different aspects. There we find problems such as clustering, classification, association rules and symbolic regression. Often these are lifted out one at a time resulting in specific techniques to solve them. It is uncommon to find a classifying technique this is capable of clustering too. Techniques from Evolutionary Computation may change this practice.

Basically machine learning is one of the last steps in the process of data mining. It is the moment where we transform raw data into knowledge, which makes it an important step as this forms the main idea behind data mining.

Here we focus on those problems from the two areas that have been solved with evolutionary computation. A short overview of these follows next. Thereafter we give an overview of the empirical research we have performed over the past five years. Then follows a short discussion about the results, after which we conclude with the research currently in progress.

2 Empirical research

We live in a real world and an important aspect is that we need solutions to problems. Sometimes this means we do not need or want to know exactly how a solution is created, which justifies the use of empirical research to some extend. Many companies are interested in new techniques solely on the basis of empirical research. In the same light Evolutionary Computation is becoming more and more popular while at the same time the interest in theory and analytic research seems to decrease. Here we will give an overview of empirical research performed on the problems we have discusses in the previous section.

Problems often exist before a solution is provided, but in some cases the reverse happens. So it seems to be with evolutionary algorithms as in the past decennia people have been searching for problems that we could solve with Evolutionary Computation. The problems with which people experimented have been solved previously to some success with other techniques.

In constraint satisfaction these established techniques consist of methods that have been derived from chronological backtracking [8] and methods that are based on simple heuristics that are influenced by a stochastic process, such as DSatur [2]. Here we will look at backtracking (BT) and forward checking with conflict-directed back-jumping (FC-CBJ) on solving binary constraint satisfaction problems and we will look at DSatur on solving k-graph colouring.

Machine learning has many different techniques for every aspect, here we restrict ourselves to some of the techniques that exist for solving classification and symbolic regression problems. Results of these techniques have been reported in Statlog [11] where it concerns classification. For symbolic regression we have used cubic regression and splines under tension [4] in our comparisons.

In [7] we measured the performance of two evolutionary algorithms, Stepwise Adaptation of Weights (SAW) and the Grouping Genetic Algorithm and one established technique called DSatur [2] when solving randomly created instances of k-graph colouring problems with $k = 3$. The tests sets consisted of instances created randomly in such a way that we look at a range of edge connectivities that overlaps the phase transition as predicted by Cheeseman et al. [3]. We found that the two evolutionary algorithms easily outperformed the established DSatur technique.

We have performed the same testing procedure for solving random binary constraint satisfaction problems [10]. The test set has been created using model B [12] and again we created a test set that overlaps the transition phase as predicted by Smith [13]. Here the roles are reversed, the two established techniques, BT and FC-CBJ, outperform the two evolutionary techniques, SAW and the Microgenetic Iterative Descent method on most of the test set except for the instances that take the least effort to solve.

To further investigate these results we set up a scale-up experiment where the number of variables in the constraint satisfaction problem is increased [10]. Again the two established techniques outperform the evolutionary algorithms.

In [9] the SAW technique was first applied in genetic programming, after further investigation we reported a comparison in [5] based on data in the Statlog libraries where we noticed that adaptive genetic programming is only a fair player on the list of techniques tested.

We then turned to symbolic regression using the same adaptive SAW technique in genetic programming [6]. For this real valued domain we were confronted once more with the fact that most problems were solved much faster and much more accurate with simple techniques such as regression and splines. These simple techniques, however, do not provide a symbolic model.

3 Discussion

When we look at the quick summing up of results in the previous section we get a grim picture of the performance of evolutionary algorithms compared to the other established techniques that were mentioned. Especially the harder problems seem to form an obstacle for evolutionary algorithms. Even worse, but less surprising, unsolvable problem instances form an even bigger problem as most evolutionary algorithms have almost no way of determining that they should give up.

This leads to the question of how to improve matters. Not only regarding speed, but especially regarding the success rate, i.e. the percentage of problems instances solved. We can easily speed up evolutionary algorithms using parallel computing, but that would not increase the success rate. Thus the question remains why the accuracy of evolutionary algorithms on these problems is low. To answer this we are need to determine when and on what evolutionary algorithms become less accurate. More specifically, which parts of an evolutionary algorithm have a negative impact on the accuracy and for which problem instances does this hold.

4 Research in progress

To better test the techniques that we develop we use a different model for randomly generating binary constraint satisfaction problems. This model, called model E, behaves much better when we scale-up problems [1]. We are applying this new model in a large scale comparison of every evolutionary technique that solves constraint satisfaction problems we have seen over the past ten years.

With the same model we are testing an extended version of the Stepwise Adaptation of Weights technique that we have named Zoom-SAW. The preliminary results are promising and will be reported later when the whole experiment is finished.

At this moment in time most of the evolutionary techniques still fall behind established techniques. Besides attempts to change this situation by trying to create new innovative algorithms and testing them, we want to gain knowledge of what is going wrong in our techniques. We are, therefore, focusing on some aspects of our techniques and problems where we thinks a loss of performance might be present. Right now we are examining the $1/l$ mutation rate that is used quite often as we think this mutation rate is far too low. From the perspective of the problem we are investigating where exactly the phase transition in constraint satisfaction is found in relation to different solving techniques.

In the long run we hope to get a model that incorporates constraint satisfaction and machine learning in such a way that we can apply different techniques transparently on many different problems from both areas. We hope that we can create a model that is similar to the model of binary constraint satisfaction problems as there is much information, both empirical and theoretical, available on this model.

References

1. D. Achlioptas, L.M. Kirousis, E.K., D. Krizanc, M. S.O. Molloy, and Y.C. Stamatiou. Random constraint satisfaction a more accurate picture. In Gert Smolka, editor, *Principles and Practice of Constraint Programming — CP97*, pages 107–120. Springer-Verlag, 1997.
2. D. Brélaz. New methods to color vertices of a graph. *Communications of the ACM*, 22:251–256, 1979.
3. P. Cheeseman, B. Kanefsky, and W.M. Taylor. Where the really hard problems are. In J. Mylopoulos and R. Reiter, editors, *Proceedings of the 12th IJCAI-91*, volume 1, pages 331–337, Morgan Kaufmann, 1991. Morgan Kaufmann.
4. A. K. Cline. Six subprograms for curve fitting using splines under tension. *Commun. ACM*, 17(4):220–223, April 1974.
5. J. Eggermont, A. E. Eiben, and J.I. van Hemert. Adapting the fitness function in GP for data mining. In R. Poli, P. Nordin, W.B. Langdon, and T.C. Fogarty, editors, *Genetic Programming, Proceedings of EuroGP'99*, volume 1598 of LNCS, pages 195–204, Göteborg, Sweden, 26–27 May 1999. Springer-Verlag.
6. J. Eggermont and J.I. van Hemert. Adaptive genetic programming applied to new and existing simple regression problems. In J. Miller, M. Tomassini, P.L. Lanzi, C. Ryan, A.G.B. Tettamanzi, and W.B. Langdon, editors, *Genetic Programming, Proceedings of EuroGP-2001*, LNCS, pages 23–35. Springer-Verlag, Berlin, 2001.
7. A.E. Eiben, J.K. van der Hauw, and J.I. van Hemert. Graph coloring with adaptive evolutionary algorithms. *Journal of Heuristics*, 4(1):25–46, 1998.
8. S.W. Golomb and L.D. Baumert. Backtrack programming. *A.C.M.*, 12(4):516–524, October 1965.
9. J.I. van Hemert. Applying adaptive evolutionary algorithms to hard problems. Master's thesis, Leiden University, 1998.
10. J.I. van Hemert. Constraint satisfaction problems and evolutionary algorithms: A reality check. In A. van den Bosch and H. Weigand, editors, *Proceedings of the Twelfth Belgium/Netherlands Conference on Artificial Intelligence (*BNAIC'00*)*, pages 267–274, De Efteling, Tilburg, The Netherlands, 1-2 November 2000.
11. D. Michie, D.J. Spiegelhalter, and C.C. Taylor, editors. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, February 1994.
12. E. M. Palmer. *Graphical Evolution.* John-Wiley & Sons, New York, 1985. *An introduction to the theory of random graphs - an important tool in modeling networks of interactions.*
13. B.M. Smith. Phase transition and the mushy region in constraint satisfaction problems. In A. G. Cohn, editor, *Proceedings of the 11th European Conference on Artificial Intelligence*, pages 100–104. Wiley, 1994.
14. E. Tsang. *Foundations of Constraint Satisfaction.* Academic Press, 1993.