IMPROVING GENET AND EGENET BY NEW VARIABLE ORDERING STRATEGIES

VINCENT TAM
Department of Computer Science
University of Melbourne, 3052
Parkville, Australia.
E-mail: vtam@cs.mu.oz.au
http://www.cs.mu.oz.au/~vtam

PETER STUCKEY
Department of Computer Science
University of Melbourne, 3052
Parkville, Australia
E-mail: pjs@cs.mu.oz.au

Constraint satisfaction problems (CSPs) naturally occur in a number of important industrial applications such as planning and scheduling defeating many algorithmic search methods. GENET and it extended model, EGENET, are probabilistic neural networks which had some remarkable success in solving some hard instances of CSPs such as a set of hard graph coloring problems. Both GENET or EGENET does not employ any variable ordering strategy in its computation to guide the search. In this paper, we proposed several new variable ordering strategies for improving GENET and EGENET. We compared the efficiency of their improved versions using the conventional or new variable ordering strategies against the original GENET and EGENET on a number of randomly generated and hard instances of binary CSPs. The improved versions with variable ordering heuristics compares favorably with the original versions on these binary CSPs. Our work shed lights on several directions for future exploration.

1 Introduction

Many important industrial applications such as planning and scheduling can be formulated as constraint satisfaction problems (CSPs) involving a set of variables, each with its domain (a set of possible values), and a set of constraints (relations) defined on some subsets of variables. The task is to assign a value to each variable from its associated domain so that none of the constraints is violated. But there exists no known general algorithm to solve a CSP, generally NP-

complete, in polynomial time.

There are two main approaches to solve CSPs. The complete search methods such as the enumerative search method are algorithmic and usually slow on most real-life applications. Thus, they need to be improved with different techniques such as chronological backtracking, conflict-directed back-jumping, consistency techniques such as AC-4, variable ordering heuristics (VOHs) or their combinations. Among these improving techniques, VOHs direct the search to a more promising subtree by altering the order in which the variables are instantiated. Besides, there are stochastic search methods, such as artificial neural networks (ANNs), genetic algorithms (GAs) and simulated annealing, which work by relaxation-based search to allow variable assignments violating some constraints during the search. They cannot guarantee to find a solution if one exists. Both GENET[1] and EGENET[2] are probabilistic ANNs for solving general CSPs. They use a convergence procedure based on min-conflict heuristic to find variable assignments representing local minima in terms of constraint violation. If the local minima are not solutions, a heuristic learning rule will then be used to escape from them.

Complete search methods guarantee the finding of all solutions while stochastic search methods are fast to find a single solution on hard instances of CSPs. Thus, there were some recent works done on integrating the techniques used in these two approaches. Stuckey and Tam proposed different integration models[5] to show how the two different solvers can work together to improve the derivation steps in the constraint logic programming systems. Also, they suggested a new kind of lazy constraint consistency[4] integrated into the EGENET computation to detect inconsistent problems with minimal cost. Riff Rojac[3] integrated constraint propagation techniques into GAs for solving CSPs. In this paper, we proposed several new variable ordering strategies suitable for the GENET or EGENET computation. Also, we evaluated the advantages gained from such improvement. Up to our knowledge, our work is the first attempt to study the possible advantages of VOHs to stochastic search methods in the context of GENET or EGENET computation.

This paper is organised as follows. Section 2 briefly describes the original GENET and EGENET. Section 3 reviews some conventional VOHs for complete search methods and the new VOHs for improving GENET and EGENET. We discuss the preliminary experimental results of the original and improved GENET and EGENET some hard instances of CSPs in section 4. Lastly, we conclude our work in section 5.

2 GENET and EGENET

GENET is a neural network simulator based on min-conflict heuristic to solve binary CSPs. In a GENET network, each variable in a CSP is represented by a cluster of nodes. Each node denotes a value in the domain of the variable , and has a state, *on* or *off*. When the state of the node denoting the value in the domain of the variable is *on*, it represents the assignment of the value to the variable . The assignment of values to all the variables denotes a network state $(X_1 = v_1, \ldots, X_n = v_n)$. Each binary constraint involving exactly the variables X and Y is defined as a set of illegal combinations, of the form $illegal(X = v, Y = u)$. The conjunction of constraints on X and Y gives an illegal constraint which is the union of their illegal combinations.

In EGENET, each variable in a CSP is represented by a variable node with its associated domain in the EGENET network. The state of a variable node is defined to be its current value assignment. Every (binary or non-binary) constraint in the CSP is denoted by a constraint node with its own set of illegal combinations in the EGENET network. In both GENET and EGENET network, every illegal combination $illegal(X = v, Y = u)$ is mapped to a positive penalty value, usually initialized to 1, by the *penalty()* function. The penalty value for any legal combination of variable assignments is always 0.

Initially, both GENET and EGENET randomly assign a value from its associated domain to each variable in the network. Then, GENET or EGENET executes the network convergence procedure in which each variable will be updated in parallel and asynchronously until there is no change in the value assigned to any variable (repair). In practice, a random permutation of variables is used for updating. To update a variable X, the input for each value v of X is calculated as the sum of penalty values of all the illegal combinations (or constraint nodes) involving variable X when $X = v$. The value with minimum input is the new value to be assigned to X. In the case of ties, if the old value has minimum input, then it remains to be assigned to X; otherwise one of the values with minimum inputs is chosen randomly. When there is no repair to any variable, the network state represents either a solution or local minima. In the latter case, a heuristic learning rule is activated to help the network escape from this local minima by penalizing the violated combinations (or constraint nodes). The heuristic learning rule in EGENET is left unspecified intentionally to allow the design of good learning rules by domain-specific knowledge.

3 Variable Ordering Strategies

Many complete search methods employ different VOHs to improve their efficiency on some real-life applications. The Minimal Width Ordering (MWO) heuristic is an example which gives a total and fixed ordering with minimum width for all the variables before the search starts. An ordering is MWO if the width, that is the maximum number of nodes adjacent and before to every node in a constraint graph, of that ordering is the minimal width of all possible orderings for a constraint graph. MWO works by ordering the less constrained variables to be labelled last to reduce the needs for backtracking. Besides, the fail-first principle (FFP), a dynamic ordering strategy, is a general heuristic to identify the most likely failed branches early in the search by usually sorting the variables in ascending order of their domain sizes. FFP is usually used in conjunction with forward checking (FC), a partial consistency techniques used to maintain the compatibility between the labelled and unlabelled variables.

There are several specially designed VOHs for GENET and EGENET. The max-degree-of-participation ordering (MDPO) sorts all the variables according to a descending order of degrees of participation of the variables. We define the degree of participation of a variable as the number of variables which share inhibitory connections with that particular variable in GENET, and the number of participating constraints in EGENET. The MDPO in GENET is the same of MDO. Another example is the max-expected-input ordering (MEIO). The expected input of a variable is the sum of weighted penalty values of all possible values for that variable. The weighting factor can be any arbitrary cost of assigning a particular value to the concerned variable in certain constraint. Besides, we propose the max-input ordering (MIO) which dynamically orders all the variables in GENET or EGENET according to a descending[a] order of inputs for the current variable assignment. Also, as an opposite to FFP, we proposed the success-first principle (SFP) since GENET or EGENET computation aims at successfully finding a solution by min-conflict heuristic. Our previous work suggested a new kind of constraint reasoning techniques, the lazy constraint consistency (LCC) techniques, most suitable for GENET or EGENET computation. Obviously, both FFP and SFP can be used in conjunction with LCC in GENET or EGENET computation.

---

[a] This is because we define positive penalty values for both GENET and EGENET.

4 Experimental results

For the graph-coloring problem, the task is to color each node of a graph so that no two connected nodes share the same color. We compared the original GENET against its improved versions on this problem.

Nodes	Colors	GENET	FIX1	MEIO
125	17	954.45s	1182.25s	748.75s
125	18	11.9s	7.15s	4.0s
250	15	2.6s	2.6s	3.8s
250	29	2172.85s	>10 hours	1916.8s

Table 1 : Original GENET versus improved GENETs with fixed random ordering (FIX1) and MEIO on a set of hard graph-coloring problem.

In general, FIX1 slows down the GENET search since the fixed random ordering does not exploit any domain-specific knowledge. The GENET with MEIO betters the original GENET since the most constrained variables possibly fixed up early in each convergence cycle to avoid many possible un-promising branches of the search tree. Also, we compared the original EGENET and its improved versions with MDPO, MWO and MIO on two hard graph-coloring problems. In the harder case, the improved EGENET with MIO required only about 1/4 of the solution finding time of the original EGENET because input value used in the GENET or EGENET computation can guide the search effectively.

The permutations generation problem is to construct all the permutations of a given range that admit the given monotonies and advances vectors. The benchmarks of the original and improved GENETs are shown as follows.

Range Size	GENET	LCC	LCC + FFP	LCC + SFP
9	0.210s	0.210s	0.155s	0.140s
10	0.325s	0.355s	0.300s	0.285s
20	6.470s	7.035s	5.570s	4.945s
30	50.61s	52.77s	73.42s	44.61s

Table 2 : Original GENET versus improved GENETs with LCC, LCC with FFP and LCC with SFP on permutations generation.

The GENET improved with LCC only performs worse than the original GENET. But the improved GENET with LCC and SFP performs the best among all the versions of GENET on these permutation generation problems since SFP considers variables with larger domain sizes after the possible pruning by LCC. Also, the improved version with LCC and FFP betters the original GENET on all except the last cases because relatively less values are removed by LCC on larger problems.

5 Conclusion

This paper represents the first attempt to study some useful VOHs for improving the performance of stochastic search methods, and to evaluate the possible advantages of specially designed VOHs for stochastic search methods such as GENET and EGENET. We proposed several new VOHs for GENET and EGENET, and built prototypes of the original GENET and EGENET and their improved versions with different VOHs to compare them on some hard binary CSPs. Our preliminary benchmarking results show that the FFP and SFP with LCC can drastically improve the GENET or EGENET on some instances of hard binary CSPs such as the permutation generation problems. For a set of hard graph-coloring problems, MEIO or MIO can significantly reduce the timing required for the original stochastic solver to find a solution.

There are several directions for future work. First, it is interesting to investigate the effects of those VOHs for GENET and EGENET on the other stochastic solvers. Second, we can study how our proposed VOHs for GENET and EGENET can be integrated in the different integration models proposed Stuckey and Tam. Lastly, the proposal of more VOHs for GENET or EGENET or other stochastic solvers will be interesting.

6 References

1. C. Wang and E. Tsang, 1991, "Solving satisfaction problems using neural-networks", in *Proceedings of IEE Second International Conference on Artificial Neural Networks.*
2. J.H.M. Lee, H.F. Leung and H.W. Won, 1995, "Extending GENET for non-binary CSP's", in *Proceedings of 7th International Conference on Tools with Artificial Intelligence*.
3. María Cristina Riff Rojas, 1996, "From Quasi-Solutions to Solution: An Evolutionary Algorithm to Solve CSP", in *Proceedings of Principles and Practice of Constraint Programming (CP96),* 367–381.
4. Peter Stuckey and Vincent Tam, 1997, "Extending E-GENET with lazy constraint consistency", (to appear) in *Proceedings of 9th International Conference on Tools with Artificial Intelligence.*
5. Peter Stuckey and Vincent Tam, 1997, "Semantics for using Stochastic Constraint Solvers in Constraint Logic Programming", (to appear) in *Journal of Functional and Logic Programming*.