A Generalized Approach to Handling Parameter Interdependencies in Probabilistic Modeling and Reinforcement Learning Optimization Algorithms

Victor V. Miagkikh and William F. Punch III
Genetic Algorithms Research and Application Group (GARAGe)
Department of Computer Science and Engineering
Michigan State University
3115 Engineering Building
East Lansing, MI 48824
Phone: (517) 353-3541
E-mail: {miagkikh,punch}@cse.msu.edu

Abstract - This paper generalizes our research on parameter interdependencies in reinforcement learning algorithms for optimization problem solving. This generalization expands the work to a larger class of search algorithms that use explicit search statistics to form feasible solutions. Our results suggest that genetic algorithms can both enrich and benefit from probabilistic modeling, reinforcement learning, ant colony optimization or other similar algorithms using values to encode preferences for parameter assignments. The approach is shown to be effective on both the Asymmetric Traveling Salesman and the Quadratic Assignment Problems.

Introduction

There has been a recent upsurge of interest in a family of search algorithms that store past experience not only as the best solutions generated, but also abstract representations of the decision processes employed. This interest is provoked by a number of factors. First, since solution memory is always limited, search algorithms which store only best solutions must necessarily ”forget” valuable experience. Second, abstractions of the decision process itself, such as decision probability estimates, sometimes better reflects important properties of the search space. There are a number of different approaches to extraction, representations and use of the information obtained in course of search. For example, cultural algorithms [Reynolds (1991)] employ version spaces for this purpose. The STAGE algorithm [Boyan and Moore (1998)] uses value-function approximation to predict the potential location of the optimum.

In this paper we would like to concentrate on a significant problem within a group of algorithms which use quantitative information to estimate the expected quality of free parameter assignments. This group includes algorithms such as:

1) population based incremental learning (PBIL) proposed by Baluja (1994) and other algorithms which use probabilistic modeling of the search space [De Bonet *et al* (1996), Baluja&Davies (1998), Pelikan *et al* (1999), Miagkikh& Punch(1999a)];

2) reinforcement learning applications to optimization problem solving [Zhang&Dietterich (1996), Crites&Barto(1996), Singh&Bertsekas (1996)], etc.;

3) ant colony optimization [Dorigo *et al* (1996)].

These algorithms collect explicit search statistics in the form of probability estimates, action-values, pheromone or other such values which indicate how likely each parameter assignment will contribute to a higher quality solution. For example, PBIL attempts to generalize information contained in a genetic algorithm (GA) population. PBIL creates a probability vector (in this case, of binary-valued chromosomes) where each component corresponds to the probability of setting the associated GA chromosome bit to 1. Unfortunately, this representation does not capture parameter interdependencies or *building blocks* encoded in the GA population. One way to account for such parameter dependencies is to use conditional probabilities [Miagkikh&Punch (1999a)] and organize those in chains [De Bonet *et al*(1996)], dependency trees [Baluja&Davies (1998)], or Bayesian networks [Pelikan *et al* (1999)] in order to build a more precise model of the search space. The major problems with these approaches are both the fundamental impreciseness of the models as the result of using low order (pairwise) conditional probabilities, and intensive the computational efforts required to maintain the models.

Handling parameter interdependencies is one of the major complications in applications of statistical reinforcement learning (RL) [Sutton&Barto (1997)] to optimization. From the viewpoint of RL, the goal of the search algorithm is to generate solutions with high expected fitness based on previous feasible solutions. RL algorithms are based on *Markov Decision Processes* (MDPs) theory. Typical RL algorithms, e.g. SARSA or Q-learning [Watkins&Dayan (1992)], look for an optimal *policy,* which is a mapping from MDP states to actions which maximizes the expected value of reward obtained on an MDP path. The *Action-value function* for some policy is defined on all possible state-action pairs of the MDP and maps them to an expected reward obtained by starting at a given state and taking only policy-dictated actions. In the majority of RL optimization applications [Zhang& Dietterich (1996), Crites&Barto(1996), Singh& Bertsekas (1996)], action-values reflect only the assignment of particular free parameters without taking into account how other assignments are made. Thus, states are no longer Markovian, and such an RL algorithm cannot keep track of parameter interdependencies. For more details and application to the Quadratic Assignment Problem (QAP) and the Traveling Salesman Problem (TSP) please see [Miagkikh&Punch(1999b)].

Ant Colony Optimization (ACO) [Dorigo *et al* 1996] is an optimization technique based on mimicking the foraging behavior of ants. It uses a heuristic value called *pheromone* to indicate the desirability of a particular choice of assignment for a free parameter. The more a particular assignment is used in good solutions, the more likely it will be chosen in future solutions. This simulates the autocatalytic process of pheromone deposition as found in studies of ant colonies. Despite the fact that ACO was inspired by the behavior of real ants, the form of the ACO update rule and the solution creation process closely resembles those used in mainstream RL and competitive learning algorithms. Like other RL optimization approaches, ACO also has problems accounting for parameter interdependencies since the pheromone represents the utility of one particular assignment, independent of other assignments.

Inability to track which *combinations* of parameter assignments lead to overall improvements in solutions is a significant disadvantage of the above-mentioned algorithms and also diminishes their performance in the presence of epistasis.

Approach

Let us first consider how the problem of parameter interdependencies is handled in GAs. GAs account for combinations of parameters by their co-occurrences in population of chromosomes. The desirability of those combinations is implicitly encoded by the frequencies of schemata in the population. Combinations of parameter assignments leading to overall improvements are called *buildings blocks* in GA theory and is one of its fundamental concepts. Of course, GAs have their own problems, e.g. schema disruption, genetic drift, etc. However, their designed ability to handle parameter interdependencies is one of GA’s major strengths. Thus, the question arises, can GA's mechanism of handling parameter interaction be used in statistic search algorithms, and can GA in turn benefit from such modifications?

For example, could a population of PBIL probability vectors achieve this purpose? A population of probability vectors can indicate the feasibility of assignments of some allele given preferences for other parameter assignments. Thus, similar to a population of solutions, a population of assignment probabilities *has* representational power to account for interdependencies [Miagkikh&Punch (1999b)]. The transition from the evolution of actual values of free parameters to the evolution of probabilities or other values indicating preferences of assignment makes the whole concept of building blocks probabilistic and requires development of special theory of their operation. This is the topic of present research that we hope to publish shortly.

The idea of evolving probabilities instead of solutions was originally proposed by Baluja and Caruana(1995) as a way of avoiding premature convergence in PBIL simulating parallel GAs. Unfortunately, it was neither experimentally tested nor supported by theory. Recently, the authors proposed evolving a population of both solutions and action-value matrices which are updated by a reinforcement learning algorithm. The approach was shown to be effective in the ATSP [Miagkikh&Punch(1999b)] and the QAP [Miagkikh&Punch(1999a)]. At the same time, Palazari and Coli (1999) proposed evolving a population of weights controlling the probability of free parameter assignments and applied it to the problem of mapping a dependency graph onto a mesh of processors and finding the sequence with minimal autocorelation. These applications in the contexts of both RL and weight controlled probabilities can be seen as special cases of more general approach, which we would like to outline further.

Let *M*(*i*,*j*) be a *n*×*a preference matrix*, where *n* is the size of the problem and *a* is the size of the chromosome encoding alphabet. Each entry is a degree of desirability of choosing the *j*-th symbol of the alphabet for the *i*-th allele in chromosome. Let *P* be a population consisting of complex entities: a solution coupled with a preference matrix. This coupling was termed in [Miagkikh&Punch(1999b)] an *agent* for lack of better name. The nature of the entries in the preference matrices can be probabilities, action-values, pheromone or some heuristic weights depending on which rule is used to update the matrices. The approach can be represented as a high-level template, shown in pseudocode of the Figure 1.

```
1.  Initialize the instances of P with solutions (using some heuristic) and preference matrices (unbiased or biased according to some heuristic);
2.  Evaluate solutions in P. Assign fitness to each complex instance in P according to some rule. Fitness of the solution is one such rule.
3.  Select k instances from P for the mating pool S. Reset the set of generated instances G.
4.  Chose two instances from S;
5.  Crossover the preference matrices. (Other operators could be applied, such as evolutionary programming-style operators or even none at all )
6.  Create an offspring solution using parental solutions and preference matrices chosen in the step 4, or parental solutions and offspring preference matrix obtained in 5;
7.  Evaluate the offspring solution. Update the offspring matrix using RL, PBIL, ACO or other update rule. (This is the place for online update. Alternatively, offline updates are possible in the step 10);
8.  Repeat steps 6 and 7  l times and select the best generated solution. l is allowed to be zero;
9.  Calculate the fitness of the complex instance in the population according to some rule. Fitness of the best solutions in the step 8 is one of the possibilities.
10. If online updates in step 7 were not made,  update the parental preference matrices or the offspring preference matrix using RL, PBIL, ACO or other update rules and the best solution obtained in 8;
11. Place the complex generated instance in G.
12. Remove parental instances from S.
13. Repeat steps 4-11 till S is empty;
14. Combine P with G and restore P to original size using some replacement strategy;
15. Increase the generation counter. Repeat steps 3-14 until the termination criterion is met.
```

Figure 1: The presented approach in pseudocode

This template defines a family of algorithms in which particular instances were proposed and experimentally tested by [Miagkikh&Punch (1999a)] and by Palazari and Coli(1999). A special case, when the instances in the population consist only of preference matrices without any associated solution is also possible, but lacks certain advantages resulting from *combined* use of the preference matrices and solutions as shown by the authors in [Miagkikh&Punch(1999b)].

Results

Some examples of this approach were tested in the context of reinforcement learning and probabilistic modeling updates. In [Miagkikh&Punch(1999b)] we described a Q-learning [Watkins, P. Dayan (1992)] update rule run in an online mode (update preference matrices, right after new solutions are generated) as applied to the ATSP, and a Monte Carlo update rule for the QAP.

Table 1 shows the results on some instances of the ATSP from the TSPLIB [Reinelt (1991)] benchmark set using PBIL2 update [Baluja (1994)] in offline settings, with parameter *l*=10 and PBIL positive and negative learning rates equal 0.01. The rest of algorithm was the same as used in the RL context [Miagkikh&Punch(1999b)]. The meaning of the columns of Table 1 are as follows: Opt./BKS. – optimal or best known solution for this problem; GA+QL – our approach with Q-learning update described in [Miagkikh&Punch(1999b)], GA+PBIL – the approach with PBIL update, Best – the best result found in 10 runs; Average – average among the best solutions found in 10 runs; StdDev – standard deviation among solution; AS - the average fitness of the best solution obtained by the AS in Gambardella and Dorigo (1996); MMAS - the average fitness of solutions found by MAX-MIN AS described in Stützle and Hoos (1997); GA+LS the average fitness of solution obtained by GA with local search by Merz and Freisleben (1997). The best solution among the three techniques is bolded.

GA+QL always outperformed ACO-based approaches and showed slightly better performance in comparison to a memetic GA with local optimization (GA+LS). Comparing GA+QL and GA+PBIL, the first, Q-learning based approach definitely wins. The low standard deviation of GA+QL and GA+PBIL is also noteworthy, as is the scalability of both algorithms to larger problems.

Conclusion

The presented approach inherits the advantages of explicit search statistic algorithms such as encoding of abstractions of the problem solving process, and solves the problem of handling interdependencies sufficiently well at rather low computational cost.

One of possible further developments of this approach is the introduction of additional statistical information reflecting the degrees of certainty in the preference values. This would allow the introduction of a *probabilistic schema* and raises many interesting theoretical issues. We are presently working on this and related issues.

Table 1: ATSP Results

Banch-mark	Opt./ BKS.	GA+QL			GA+PBIL			AS	MMAS	GA+LS
		Best	Average	StdDev	Best	Average	StdDev			
Ft70	38673	38673	**38674.0**	8.50	38703	38703.0	0.0	39099.1	38707	38674.2
Ftv170	2755	2810	2824.7	16.13	2810	2834.7	23.2	2826.5	2807	**2762.2**
Kro124p	36230	36230	36263.3	59.78	36230	36322.8	195.6	36857.0	36655	**36231.5**
P43	5620	5620	**5620**	0.0	5620	**5620**	0.0	N/A	5623.8	5620.1
Ry48p	14422	14422	**14422**	0.0	14422	**14422**	0.0	14565.45	14494	14451.2
Rbg323	1326	1342	**1350.4**	5.68	1366	1376.4	9.8	N/A	N/A	N/A
Rbg443	2720	2720	**2722.8**	3.55	2720	2727.9	4.7	N/A	N/A	N/A

References

Baluja, S. (1994) "Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning", Carnegie Mellon University, Technical report, CMU-CS-94-163.

Baluja, S. & Caruana, R. (1995) "Removing the Genetics from the Standard Genetic Algorithm" *in Proc. of the Int. Conf. on Machine Learning 1995 (ML-95*), Morgan Kaufmann, pp. 38-46.

Baluja, S. & Davies, S. (1998) “Fast Probabilistic Modeling for Combinatorial Optimization” *in Proc. of the 15th National Conference on Artificial Intelligence (AAAI-98)*, pp. 469-476, AAAI Press, July 26-30 1998.

De Bonet, J.,Isbell, C., Viola, P.(1996) “MIMIC: Finding Optima by Estimating Probability Densities",*Advances in Neural Information Processing*, MIT Press, vol. 9, p. 424.

J. A. Boyan and A. W. Moore (1998), Learning Evaluation Functions for Global Optimization and Boolean Satisfiability in Proc of the 15th National Conference on Artificial Intelligence (AAAI-98), AAAI Press, pp. 3-10.

R. H. Crites and A.G. Barto(1996), “Improving Elevator Performance Using Reinforcement Learning”, *in Advances in NIPS*, Vol. 8, pp. 1017-1023, The MIT Press.

M.Dorigo, V. Maniezzo, and A. Colorni (1996). “The Ant System: Optimization by a Colony of Cooperating Agents”, *IEEE Transactions on Syst.M&Cyber.,Part B*, IEEE Press, 26 (1):29-41.

L.Gambardella, M. Dorigo (1996). “Solving Symmetric and Asymmetric TSPs by Ant Colonies” in Proc. of *IEEE Conf. on Evolutionary Computation*, 622-627, IEEE Press.

P. Merz, B. Freisleben (1997). “Genetic Local Search for the TSP: New Results” in Proc. of the *1997 IEEE Int. Conf. on Evolutionary Computation*, 159-164, IEEE Press.

Miagkikh, V. and Punch, W. (1999a) "Global Search in Combinatorial Optimization using Reinforcement Learning Algorithms" *in Proc. of the 1999 Congress on Evolutionary Computation (CEC99)*, Vol. 1, pp. 189-196, IEEE, 6-9 July 1999.

Miagkikh, V. and Punch, W. (1999b) "An Approach to Solving Combinatorial Optimization Problems Using a Population of Reinforcement Learning Agents", *Genetic and Evolutionary Computation Conf. (GECCO-99)*, Vol. 2, pp. 1358-1365 Morgan Kaufmann, 13-17 July 1999.

Palazzari, P. and Coli, M. (1999) “Evolving Probabilistic Chromosomes in Genetic Algorithms*” in Proc. of the Genetic and Evolutionary Computation Conference(GECCO-99)*, Vol. 1, pp. 511-518, Morgan Kaufmann, 13-17 July 1999.

Pelikan, M. and Goldberg, D. and Cantu-Paz, E. (1999) “BOA: The Bayesian Optimization Algorithm” *in Proc. of the Genetic and Evolutionary Computation Conference(GECCO-99)*, Vol. 1, pp. 525-532, Morgan Kaufmann, 13-17 July 1999.

G. Reinelt (1991). “TSPLIB-A Traveling Salesman Problem Library”, *ORSA Journal on Computing*, 3(4): 376-384.

R. G. Reynolds (1991). “Version space controlled genetic algorithms”, *in Proc. of the Second Annual Conf. on AI Simulation and Planning in High Autonomy Systems*, IEEE Press, pp. 6-14.

S. Singh, D. Bertsekas (1996). “Reinforcement Learning for Dynamic Channel Allocation in Cellular Telephone Systems”. In Proc. of *Advances in NIPS*, 974-980, MIT Press.

T. Stützle, H. Hoos (1997). “MAX-MIN Ant System and Local Search for the Traveling Salesman Problem”, in *Proc. of 1997 IEEE 4th Int. Conf. On Evolutionary Computation*, 308-313, IEEE.

R. Sutton, A. Barto (1997). *Reinforcement Learning: An Introduction*. MIT Press.

C. Watkins, P. Dayan (1992). “Q-learning”. *Machine Learning*, 8:279-292, Kluwer Academic Publishers.

W. Zhang, T. Dietterich (1996). “High Performance Job-Shop Scheduling with a Time-delay TD(λ) Network” in Proc. of *Advances in NIPS*, 1024-1030, MIT Press.