Improving Convergence of CMA-ES Through Structure-Driven Discrete Recombination

Tim Brys and **Ann Nowé**
$\{tim.brys, ann.nowe\}$ $@vub.ac.be$
Computational Modelling Lab, VUB
Pleinlaan 2
1050 Brussels, Belgium
Tel: +0032 2 629 12 27

Abstract

Evolutionary Strategies (ES) are a class of continuous optimization algorithms that have proven to perform very well on hard optimization problems. Whereas in earlier literature, both intermediate and discrete recombination operators were used, we now see that most ES, e.g. CMA-ES, use only intermediate recombination. While CMA-ES is considered state-of-the-art in continuous optimization, we believe that reintroducing discrete recombination can improve the algorithms' ability to escape local optima. Specifically, we look at using information on the problem's structure to create building blocks for recombination.

In evolutionary computation, a population of candidate solutions is evolved by applying mutation and recombination. Mutation alters elements of a single solution, while recombination combines elements from different individuals to create a new candidate solution. A typical recombination operator is the crossover operator, where a new solution is produced by essentially copying parts from different parents and glueing them together. When performed randomly, this process of crossover can disrupt optimized substructures present in a parent by only inheriting part of the substructure into the offspring. In the domain of GAs (genetic algorithms), research into detecting and using a problem's structure to improve the performance of crossover recombination is ongoing (Harik, Lobo, and Sastry 2006; Thierens and Bosman 2011; Pelikan, Hauschild, and Thierens 2011). On the other hand, in Evolutionary Strategies (ES) (Bäck, Hoffmeister, and Schwefel 1991; Beyer and Schwefel 2002), research has moved away from the concept of crossover or discrete recombination. See for example the state of the art Covariance Matrix Adaptation Evolution Strategy (CMA-ES)(Hansen 2006), which only uses intermediate recombination in its optimization, calculating the point of gravity (weighted average) of the best current candidate solutions.

When investigating infinite-valued SAT, or satisfiability in infinite-valued logics, which can be modelled as a continuous optimization problem over the domain $[0, 1]^n$, with n the number of variables, we applied the standard CMA-ES as a solver and have significantly improved upon the state of the art (Schockaert, Janssen, and Vermeir 2012) (our results are to be published). Still, on a number of instances, the algorithm regularly fails to converge to the global optimum.We hypothesise – and initial experiments confirm this – that such infinite-valued SAT problems have an inherent structure to them that could be used to improve the likelihood of convergence to the global optimum. We aim to incorporate a more informed discrete recombination operator into CMA-ES, by explicitly detecting correlations between variables and creating clusters of variables that can be exchanged between candidate solutions, improving the algorithms' ability to escape local optima. In the next sections, we explain how this structure can be derived from the covariance matrix used in CMA-ES and used in recombination.

Clustering variables

The CMA-ES algorithm relies on the adaptation of the covariance matrix of a multi-variate normal search distribution, from which new individuals are sampled. The covariance matrix is adapted to fit the search distribution to the contour lines of the function to be minimized. Thus, in this covariance matrix is captured an estimation of the correlations between variables, which can be used to derive a clustering of variables to be used in discrete recombination.

The first step to achieve this clustering is building a weighted graph that represents the structure of the problem. For that, we use the eigendecomposition of the covariance matrix, which is already calculated within the CMA-ES. The correlation between two variables is determined by looking at the magnitudes of the corresponding components in each eigenvector. For each eigenvector, we take the product of the magnitudes of the two components in question. These products are summed, each product weighted by the eigenvalue of the eigenvector. The result of this method is that variables that have a relatively large magnitude in the same, important eigenvectors, will be assigned a larger weight in the structure graph. Pseudocode is shown in Algorithm 1. The weights need to be normalized so that, if represented as a matrix, elements on the diagonal equal 1, i.e. full auto-correlation for variables. Normalization is performed as follows:

$$weight_{x,y} = \frac{weight_{x,y}}{\sqrt{weight_{x,x}} * \sqrt{weight_{y,y}}} \tag{1}$$

Algorithm 1 Correlation between variables/dimensions x and y

```
$weight_{x,y} = 0$
for $i = 1 \to N$ do
    $weight_{x,y} = weight_{x,y} + (eigenvector_i[x] * eigenvector_i[y] * eigenvalue_i)$
end for
```

Starting with this graph, we can cluster variables based on the strength of correlations. We start with the highest correlation in the graph and merge the two corresponding variables into one node. The correlations between the grouped variables and the others then need to be recalculated, by averaging the original correlations between each variable and the others. See Algorithm 2. We find the new highest correlation in the graph and repeat until the highest correlation drops below a threshold. The nodes in de resulting graph are then the most significant substructures of the problem and can be used as building blocks to be exchanged during recombination. This structuring, a set of disjunct sets, is referred to as the Marginal Product Structure (Thierens and Bosman 2011) in the context of GAs.

Algorithm 2 New correlation between grouped x,y and rest

```
for $a \in nodes \setminus \{x, y\}$ do
    $weight_{(x,y),a} = \frac{nrVars(x)*weight_{x,a}+nrVars(y)*weight_{y,a}}{nrVars(x)+nrVars(y)}$
end for
```

Structure-driven discrete recombination

The implementation of the discrete recombination that we evaluate in this preliminary research, is quite simple. We run a number of CMA-ES populations in parallel, and each generation, we allow discrete recombination, based on the problem derived structure, between populations. This recombination applies to the following CMA-ES algorithmic parameters: the mean, the covariance matrix and the evolution path. The two last ones are recombined to keep intact as much as possible of the meta-information calculated by CMA-ES. Recombination of the mean and evolution path is trivial, as it simply involves exchanging the elements of the corresponding dimensions. The recombination of the covariance matrix is less simple, as it encodes information between variables that may or may not be kept together during recombination. The way the matrix crossover is implemented is as follows: if variables i and j come from the same parent, the covariance information for element (i, j) in the child covariance matrix comes from the corresponding parent. If the variables come from different parents, the average of the covariance information from both parents is calculated for the child.

This method is able to detect the structure of simple toy examples, functions built over 10 variables by summing two-dimensional gaussian distributions over several of the variables, creating correlated substructures, and achieves convergence in about 9% less function evaluations compared to the same number of CMA-ES populations running in parallel, without discrete recombination.

On the Michalewicz benchmark function, see Equation 2, our method detects that it is dimension-wise decomposable and applies uniform crossover. This results in a halving of the number of function evaluations required to reach the VTR (value to reach), and more importantly, it reaches this value in 93 out of 100 runs, compared to 48 without discrete recombination.

$$-\sum_{i=0}^{l-1} \sin(x_i) \sin^{20}\left(\frac{(i+1)x_i^2}{\pi}\right) \tag{2}$$

Conclusions and future work

Using the extension to CMA-ES as explained in the previous sections, we are able to detect significant correlations between variables, and build an approximate model of the problem's internal structure. We are currently conducting experiments to evaluate the effect of discrete recombination on infinite-valued SAT problems. We also intend to investigate whether discrete recombination can be introduced into CMA-ES in other ways.

Additional material concerning this paper, including results of experiments currently being conducted, can be accessed at the url: http://como.vub.ac.be/tbrys/AAAI2012.

References

Bäck, T.; Hoffmeister, F.; and Schwefel, H.-P. 1991. A survey of evolution strategies. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, 2–9. Morgan Kaufmann.

Beyer, H.-G., and Schwefel, H.-P. 2002. Evolution strategies – a comprehensive introduction. *Natural Computing* 1:3–52.

Hansen, N. 2006. The CMA evolution strategy: a comparing review. In Lozano, J.; Larranaga, P.; Inza, I.; and Bengoetxea, E., eds., *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*. Springer. 75–102.

Harik, G.; Lobo, F.; and Sastry, K. 2006. Linkage learning via probabilistic modeling in the extended compact genetic algorithm (ecga). In Pelikan, M.; Sastry, K.; and CantúPaz, E., eds., *Scalable Optimization via Probabilistic Modeling*, volume 33 of *Studies in Computational Intelligence*. Springer Berlin / Heidelberg. 39–61.

Pelikan, M.; Hauschild, M. W.; and Thierens, D. 2011. Pairwise and problem-specific distance metrics in the linkage tree genetic algorithm. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, 1005–1012. New York, NY, USA: ACM.

Schockaert, S.; Janssen, J.; and Vermeir, D. 2012. Satisfiability checking in lukasiewicz logic as finite constraint satisfaction. *JOURNAL OF AUTOMATED REASONING*.

Thierens, D., and Bosman, P. A. N. 2011. Optimal mixing evolutionary algorithms. In Krasnogor, N., and Lanzi, P. L., eds., *GECCO*, 617–624. ACM.