On Empirical Memory Design, Faster Selection of Bayesian Factorizations and Parameter-Free Gaussian EDAs.

Peter A.N. Bosman[a]

[a] *Centre for Mathematics and Computer Science, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands,* Peter.Bosman@cwi.nl

The full version of this paper has been accepted for publication at the *2009 Genetic and Evolutionary Computation Conference — GECCO*.

1 Introduction

Estimation-of-distribution algorithms (EDAs) are optimization algorithms at the frontier of genetic- and evolutionary computation (GEC) research. Characteristic of EDAs is the iteration of selecting promising solutions, estimating a probability distribution from the selected solutions and subsequently generating new solutions by drawing samples from the estimated distribution. Probability distributions provide a principled way of modelling dependencies between problem variables. Contrary to classic GEC methods, this allows EDAs to successfully and automatically identify and exploit problem structures with respect to dependencies between problem variables. EDAs are therefore able to solve a much larger class of problems efficiently without requiring prior knowledge. In this paper we consider three ways of efficiency enhancement of EDAs: reducing the required population size, reducing the time to estimate the probability distribution and a restart-scheme to improve results in vastly multi-modal search spaces.

2 Empirical Memory Design

Selected solutions of subsequent generations typically have a lot in common. Re–estimating from scratch in each generation, as is usually done in EDAs, is therefore likely not the most efficient approach. A memory can be used to store and update information of previous generations. This reduces the population size required to estimate the distribution properly.

Let $\boldsymbol{\theta}$ be a subset of all distribution parameters. For learning, we use the typical memory–decay formula $\boldsymbol{\theta}(t) = (1-\eta)\boldsymbol{\theta}(t-1) + \eta\hat{\boldsymbol{\theta}}(t)$. Taking into account the decreasing contribution level of the distribution parameters estimated in generation t throughout the subsequent generations and the overall selection size from which the parameters must be derived, we obtain a function class with three parameters α_0, α_1 and α_2 and two variables $|\boldsymbol{\mathcal{S}}|$ (the generational selection size) and l (the number of problem variables) that is likely to allow for a good fit of η to the data, namely $\eta = 1 - \exp\left(\alpha_0|\boldsymbol{\mathcal{S}}|^{\alpha_1}/l^{\alpha_2}\right)$.

We applied empirical memory design for a Gaussian EDA, i.e. an EDA in which the normal distribution is used. The main parameter for which we built a memory is the covariance matrix. For a typical benchmark set of problems we determined for dimensionalities $l \in \{5, 10, 20, 40\}$ and population sizes $n \in \{10, 20, 40, 80, 160\}$ the best value for η as the combination that leads to the minimum average number of evaluations to reach the value–to–reach (VTR), averaged over 10 independent runs. With this data, non–linear regression was performed to determine the α_i parameters. An illustration of the regressed function for $\eta^{\boldsymbol{\Sigma}}$ and the underlying data for the covariance matrix is presented in Figure 1. Also shown is the ratio of the number of evaluations required if a memory is used compared to when no memory is used. An improvement is clearly obtained through the empirical memory design. Moreover, the required population size decreased dramatically from $17 + 3l^{1.5}$ to $10l^{0.5}$. For more extensive results, see the full version of this paper.

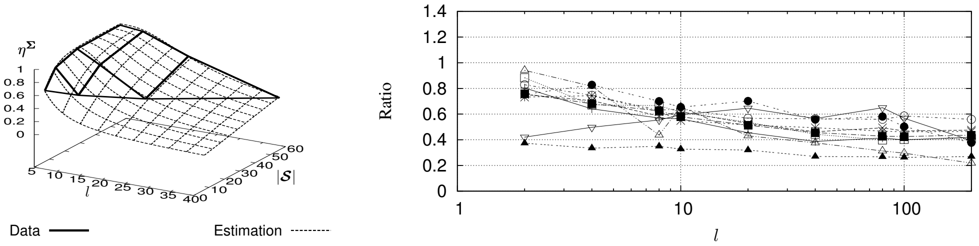

Figure 1: **Left**: regression results for the memory function for the covariance matrix. **Right**: ratio of evaluations for use of a memory versus no memory for all benchmark problems.

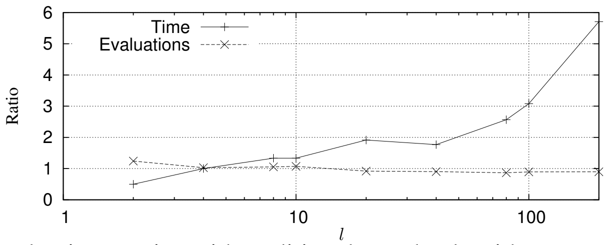

Figure 2: Time– and evaluations–ratios with traditional greedy algorithm versus new greedy algorithm.

3 Faster Selection of Bayesian Factorizations

To reduce the number of parameters to be estimated, Bayesian factorizations are commonly used in EDAs to model only a selection of all possible dependencies between problem variables. We will not discuss the details of Bayesian factorizations here, but Bayesian factorization selection in EDAs is often done with a greedy algorithm whose computational complexity is $\mathcal{O}(l^3)$ for the normal distribution. In this paper we propose a faster greedy algorithm that obtains results of a similar quality. To this end, the covariances are sorted and all dependencies are considered in the order of the magnitude of their covariance. This results in a running time of $\mathcal{O}(l^2\log(l))$ while the number of required evaluations to solve the benchmark problems remains almost the same as shown for experimental results in Figure 2.

4 Parameter–free Gaussian EDAs

Most parameter–free GEAs employ an exponential population–sizing scheme. If the population size increases, EDAs with the Gaussian distribution become more robust on slopes superimposed with many small deviations that make up local optima. A large enough population blurs out the local optima. Enlarging the population size may not always increase the probability of success however. If there are multiple local optima, but no underlying structure, using a larger population doesn't help as efficiently because the irregularities are too large to be smoothed out. We therefore propose a different scheme in which alternatingly the population size is increased and the number of parallel populations is increased. The parallel populations are initialized spatially separated and then used in independent parallel executions of the EDA. For all results, we refer the reader to the paper. For Michalewicz' function using the default population-increasing restart strategy the optimum was not found in 376 out of 1000 runs in 5 dimensions and 250 out of 1000 runs in 10 dimensions when $5 \cdot 10^6$ and $10 \cdot 10^6$ evaluations were allowed. With the alternating scheme that allows for parallel executions of the EDA, the optimum was found in *all* 1000 runs.

5 Conclusions

We have discussed various ways to improve the efficiency of EDAs. A memory can be used to reduce population size requirements. The approach taken here is empirical in nature, but also quite general and may therefore well be used in other EDAs. Greedy Bayesian factorization selection was sped up for Gaussian EDAs and a new scheme for real–valued parameter–free optimization was proposed. Overall, the results are encouraging. In future work we intend to validate the efficiency and effectiveness of EDAs for practical optimization problems; typically dynamic and/or multi–objective problems. For these cases, a small population size and a small generational running time are vital.