GA-EDA: Hybrid Evolutionary Algorithm Using Genetic and Estimation of Distribution Algorithms

J.M. Peña, V. Robles, P. Larrañaga, V. Herves, F. Rosales, and M.S. Pérez

Universidad Politécnica de Madrid, Madrid, Spain
{jmpena, vrobles, vherves, frosal, mperez}@fi.upm.es
Universidad del País Vasco, San Sebastián, Spain
ccplamup@si.ehu.es

Abstract. Evolutionary techniques are one of the most successful paradigms in the field of optimization. In this paper we present a new approach, named GA-EDA, which is a new hybrid algorithm based on genetic and estimation of distribution algorithms. The original objective is to get benefits from both approaches. In order to perform an evaluation of this new approach a selection of synthetic optimizations problems have been proposed together with two real-world cases. Experimental results show the correctness of our new approach.

Keywords. Genetic Algorithms and Heuristic Search

1 Introduction

Evolutionary techniques stand from the assumption that a restricted set of solutions could be evolved to improve solutions in an iterative process. The evolutionary process is driven by a fitness function, which measures how good each solution is.

Many pure (GAs and EDAs) techniques have been proposed as well as other combination of them, named hybrid algorithms. Examples of them are: ERA, which incorporates, simulated annealing. [Rodriguez-Tello & Torres-Jimenez, 2003]; GASAT that incorporates local search within the genetic framework [Hao & Lardeux & Saubion, 2002]; and an integrated Genetic Algorithm with Hill Climbing that solves the matrix bandwidth minimization problem [Lim & Rodrigues & Xiao, 2003]. Also, a hybrid algorithm based on the combination of EDA with Guided Local Search (GLS) for Quadratic Assignment Problems (QAP) [Zhang & Sun & Tsang & Ford, 2003]; another hybrid genetic algorithm that combines efficient local heuristic and aging mechanism for the hexagonal tortoise problem [Choe & Choi & Moon, 2003].

In this paper we present a new approach, named GA-EDA, which is a new hybrid algorithm based on genetic and estimation of distribution algorithms.

R. Orchard et al. (Eds.): IEA/AIE 2004, LNAI 3029, pp. 361-371, 2004.

2 Evolutionary Optimization Methods

Among the different evolutionary techniques the best known are Genetic Algorithms, although new approaches, like Estimation of Distribution Algorithms, have arise in the very last years.

2.1 Genetic Algorithms

Genetic Algorithms are heuristics search and optimization algorithms, highly parallel, inspired by the Darwinian principle of natural selection and genetic reproduction [Goldberg, 1989].

Genetic Algorithms begins with a population of individuals, each one representing a possible solution of a given problem. These individuals are represented as chromosomes. Chromosomes generally are sequences of bits, but often the problem demands one more complex representation. Any chosen representation, should to be able to represent the entire space search to investigate. Representation must be minimum since if it contains unnecessary information the size of the space search increase and therefore the efficiency of the GA decreases during the search.

Pseudocode for the GA approach:

$P_0 \leftarrow$ Generate M individuals (the initial population)
Repeat until stopping criterion is reached ($i = 1 \ldots n$):

- Selection:
    - $P_{intermediate} \leftarrow$ Select N individuals ($N \leq M$) from P_{i-1} according to some selection mechanisms
    - Select P individuals ($P \leq N$) from $P_{intermediate}$ that will be the progenitors
- Reproduction:
    - P individuals of $P_{intermediate}$ are selected and joined in pairs, and Q descendants are generated
    - Pintermediate $\leftarrow N + Q$
- Replacement:
    - $Pi \leftarrow$ Select M individuals from $P_{intermediate}$, generally the fittest.

2.2 Estimation of Distribution Algorithms

EDAs [Larrañaga & Lozano, 2001] [Mühlenbein, 1998] are non-deterministic, stochastic heuristic search strategies that form part of the evolutionary computation approaches, where a number of solutions or individuals are created every generation, evolving once and again until a satisfactory solution is achieved. In brief, the characteristic that differentiates most EDAs from other evolutionary search strategies such as GAs is that the evolution from a generation to the next one is done by estimating the probability distribution of the fittest individuals, and afterwards by sampling the induced model. This avoids the use of crossing or mutation operators, and the number of parameters that EDAs require is considerably reduced.

In the pseudocode of a generic EDA algorithm, we can distinguish four main steps:

1. At the beginning, the first population D_0 of *M* individuals is generated.
2. A number *N* (*N* <= *M*) of individuals are selected, usually the fittest.
3. The *n*–dimensional probabilistic graphical model that better expresses the dependencies among the *n* variables is induced.
4. A new population of *M* new individuals is obtained by simulating the probability distribution learnt in the previous step.

2.3 Comparative Results

There are several studies of the properties and qualities of these two approaches for different problems (see chapters 13,16 and 17 of [Larrañaga & Lozano, 2001]). One of the most important results obtained on these and similar studies is that none of them outperforms the other for all the possible problems. There are cases in which GAs converge slower to the solution and there are other cases in which EDAs fall in a local optimum. Sometimes the absolute optimum is obtained only by one of these algorithms. The reason depends on characteristics of the very problem, and only for few specially designed problems is possible to predict whether GAs or EDAs are going to perform better.

3 Hybrid GA-EDA Algorithm

On this paper we propose a new algorithm based on both techniques. The original objective is to get benefits from both approaches. The main difference from these two evolutionary strategies is how new individuals are generated. These new individuals generated on each generation are called *offspring*. On one hand, GAs uses crossover and mutation operators as a mechanism to create new individuals from the best individuals of the previous generation. On the other, EDAs builds a probabilistic model with the bests individuals and then sample the model to generate new ones.

Participation Function Our new approach generates two groups of offspring individuals, one generated by the GA mechanism and the other by EDA one. $Population_{p+1}$ is composed by the best overall individuals from (i) the past population ($Population_p$), (ii) the GA-evolved offspring, and (iii) EDA-evolved offspring.

The individuals are selected based on their fitness function. This evolutionary schema is quite similar to Steady State GA in which individuals from one population, with better fitness than new individual from the offspring, survive in the next one. In this case we have two offspring pools. Figure 1 shows how this model works.

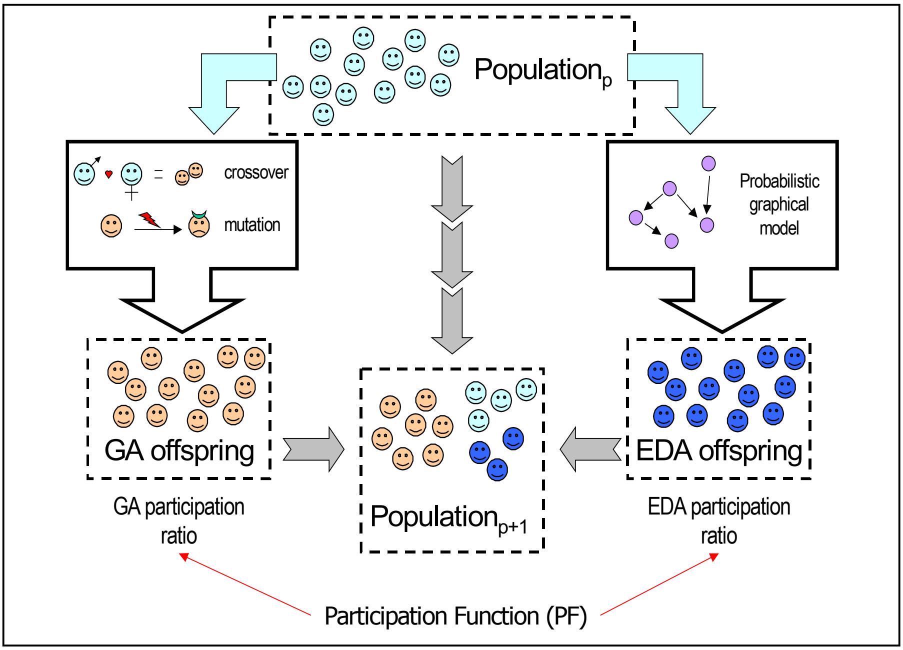


Fig. 1. Hybrid Evolutionary Algorithm Schema

On this new approach an additional parameter appears, this parameter has been called *Participation Function* (PF). PF provides a ratio of how many individuals are generated by each mechanism. In other words, the size of GA and EDA offspring sets. The size of these sets also represents how each of these mechanisms participates on the evolution of the population. These ratios are only a proportion for the number of new individuals each method generates, it is not a proportion of individuals in the next population, which is defined by the quality of each particular individual. If a method were better that the other in terms of how it combines the individuals there would be more individuals from this offspring set than the other.

The following alternatives for Participation Functions are introduced:

Constant Ratio (*x% EDA / y% GA*)
The percentage of individuals generated by each method is constant during all the generations. For example, *30%* of the individuals are generated by GA crossover and mutation and *70%* by the EDA probabilistic graphical model.

Incremental Ratio (*EDA*++ and *GA*++)

The partition ratio for one of the mechanism increases from one generation to the other. There are two incremental Participation Functions, GA Incremental Function and EDA Incremental Function. The ratio is defined by the formula[1]:

$$ratio = \frac{gen}{M + gen}$$

Alternative Ratio (*ALT*)

On each generation it alternates either GA or EDA generation method. If the generation is an even number GA mechanism generates all offspring individuals, if it is an odd number is the EDA method.

Dynamic Ratio (*DYNAMIC*)

As a difference with the previous Participation Functions that are static (and deterministic), we also propose a dynamic adaptative function. The idea is to have a mechanism that increases the participation ratio for the method that happens to generate better individuals. This function is evaluated each generation considering the possibility to change the participation criterion (defined by the `ratio` array).

This function performs according to the following algorithm:

```
diff=(MAX(avg_score[GA],avg_score[EDA])-base)/
     (MIN(avg_score[GA],avg_score[EDA])-base);
if(avg_score[GA]>avg_score[EDA]){
 ratio_inc=ratio[EDA]*ADJUST*diff;
 ratio[GA]  += ratio_inc;        ratio[EDA]= 1.0 - part[GA];
}
else if(avg_score[GA]<avg_score[EDA]){
 ratio_inc=ratio[GA]*ADJUST*diff;
 ratio[EDA] += ratio_inc;        ratio[GA] = 1.0 - part[EDA];
}
```

Where `avg_score` is an array of the average fitness score of the top *25%* of the individual generated by each of the offspring methods[2]. As the best fitness score is monotonically increasing this value is always greater than 1. `ADJUST` is a constant that defines the size of the steps of the dynamic update (*5%* in our experimentation).

This algorithm starts with *50%/50%* ratio distribution between the two methods. On each generation the best offspring individuals from each method are compared and the wining method gets a *5%* of the opposite method ratio (scaled by the amount of relative difference between the methods, `diff` variable). This mechanism provides a contest-based dynamic function, in which methods are competing to get higher ratio as they generate better individuals.

---

[1] *gen* is the number of the generation and *M* is called the Mid-point that represents at which generation the ratio is *50%/50%*. Function is 0 at the first generation and never reaches 1.

[2] `base` is the best fitness score obtained in the first generation (used to scale fitness values).

4 Evaluation of the New Algorithm

The hybrid algorithm proposed is composed by the simplest versions of both GA and EDA component. In this sense a single bit-string chromosome has been used to code all the problems. GA uses “Roulette Wheel” selector, one-point crossover, flip mutation and uniform initialization. EDA uses UMDA probabilistic model. The overall algorithms generate an offspring twice the size of the population (this offspring is then divided between two methods depending on the ratios provided by the Participation Function). The composition of the new population is defined by a deterministic method, selecting the best overall fitness scores from the previous population and both offspring sets.

All experiments have been run ten times, and the values in the figures of the experimental result section are the average of these executions. As Participation Functions for the hybrid approach we have tested the next ones: *75% EDA/ 25% GA*, *50% EDA / 50% GA* and *25% EDA / 75*% GA as constant ratio functions, as well as *EDA++*, *GA++*, *ALT*, and *DYNAMIC* functions.

4.1 Description of the Problems

Four classes of problems were empirically tested on our new hybrid approach, two artificial problems (*4-bit fully deceptive function* and *240 bit Holland royal road*) and two real problems (*SAT problem* and *feature subset selection* problem).

4-Bit Fully Deceptive fuNction

Deceptive trap functions are used in many studies of GAs because their difficulty is well understood and it can be regulated easily [Deb & Golberg, 1993]. We have used the 4-bit fully deceptive function of order 2, defined by Whitley and Starkweather in their paper GENITOR II [Whitley & Starkweather, 1990]. The problem is a 40 bit long maximization problem, and is comprised of 10 sub-problems, each 4 bits longs.

240 Bit Holland Royal Road

The Royal Road functions were introduced in [Mitchell et al., 1992]. They were designed as functions that would be simple for a genetic algorithm to optimize, but difficult for a hillclimber. In [Holland, 1993], Holland presented a revised class of Royal Road functions that were designed to create insurmountable difficulties for a wider class of hillclimbers, and yet still admissible to optimization by a GA.

The Holland Royal Road function takes a binary string as input and produces a real value. The function is used to define a search task in which one wants to locale strings that produces high function values. The string is composed of 2^k non-overlapping continuous regions, each of length $b+g$. With Hollands’s defaults, $k=4$, $b=8$, $g=7$, there are 16 regions of length 15, giving and overall string length of 240. Each region is divided into two non-overlapping pieces. The first, of length b, is called the block, and the second, of length g, is called the gap. In the fitness calculation, only the bits in the block part of each region are considered.

SAT Problem

The goal of the satisfiability (SAT) problem [Rodriguez-Tello & Torres-Jimenez, 2003] is to find an assignment of truth-values to the literals of a given boolean formula, in its conjunctive normal form, that satisfies it. In theory SAT is one of the basic core NP-complete problems. In practice, it has become increasingly popular in different research fields, given that several problems can be easily encoded into propositional logic formula such as planning, formal verification, knowledge representation and so on.

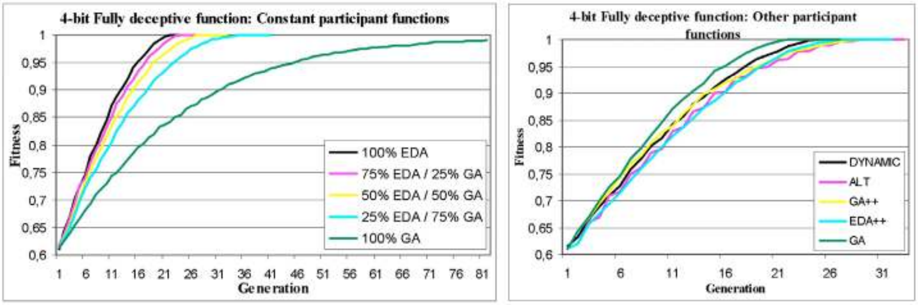
In GAs and EDAs the SAT problem is represented using binary strings of length n in which the *i-th* bit represents the truth-value of the *i-th* propositional variable in the formula. The fitness function used is the fraction of clauses satisfied. To test the algorithm developed the SAT instances *4blocksb.cnf* was used, since they are widely-known and easily available from the SATLIB benchmark[3].

Feature Subset Selection

Feature Subset Selection (FSS) is a well-known task in the Machine Learning, Data Mining, Pattern Recognition and Text Learning paradigms. FSS formulates as follows: *Given a set of candidate features, select the best subset under some learning algorithm*. As the learning algorithm we are going to use naïve Bayes [Duda & Hart, 1973] [Hand & Yu, 2001]. A good review of FSS algorithm can be found in [Liu & Motoda, 1998]. To test the FSS problem we will use the *chess* dataset from the UCI repository [Murphy & Aha, 1995], which has a total of 36 features and 699 instances.

Results: 4-Bit Fully Deceptive Function

Figure 2 shows the results for the 4-bit fully deceptive function using a population of 1000 individuals. Results for other size of populations are really similar.



(a) Constant Participation Functions (b) Other Participation Functions

Fig. 2. Results for the 4-bit Fully deceptive function

The best results are obtained with EDAs, while the worst are obtained with GAs. Using EDAs the maximum is reached in approximately 23 generations. On the other hand, using GAs, after 91 generations the maximum is never found. About our new hybrid approaches, we always reach the maximum, being the best Participation Function the dynamic one, which reaches the maximum in 27 generations and the worst Participation

---

[3] (http://www.satlib.org/benchm.html). *4blocksb.cnf* contain 24758 clauses, 410 propositional variables and is satisfiable.

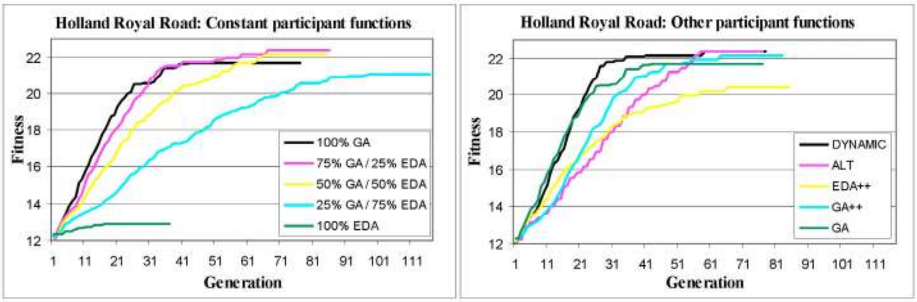
Function the constant ration with 25% EDA / 75% GA which reaches the maximum in 36 generations.

In conclusion, we always reach the maximum with our hybrid approaches and the bad results obtained with GAs only affect our hybrid in the number of generations required.

Results: 240 Bit Holland Royal Road

This problem is just the opposite of the previous one. As it is possible to see in Figure 3, with a population of 1000 individuals, the performance of GAs is much better than the performance of EDAs. With EDAs is only possible to achieve a fitness value of 12.91, while with GAs this value is 21.07. However, most of the hybrid approaches are better than GAs, being the best obtained value 22.37 with the DYNAMIC and the CONSTANT 75% GA / 25% EDA Participation Functions.

In conclusion, for the 240 bit Holland Royal Road problem our hybrid approach performs better than GAs and EDAs.

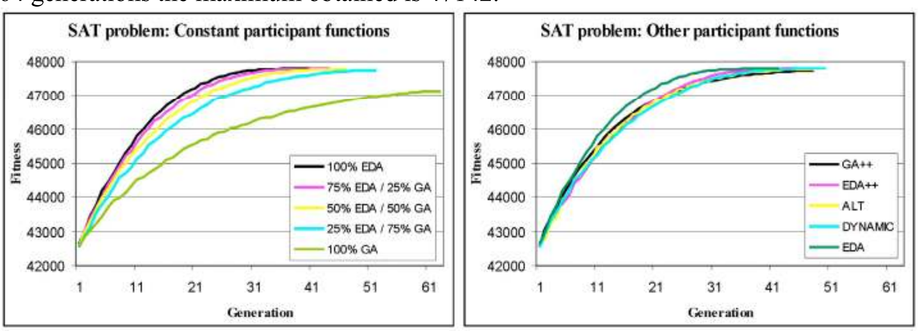


(a) Constant Participation Functions (b) Other Participation Functions

Fig. 3. Results for the Holland Royal Road problem

Results: SAT Problem

The experimental results obtained for the SAT problem are quite similar to the results of the 4-bit fully deceptive function (see Figure 4). The best results are obtained with EDAs, while the worst are obtained with GAs. Using EDAs the maximum (fitness = 47803) is reached in approximately 43 generations. On the other hand, using GAs, after 64 generations the maximum obtained is 47142.



(a) Constant Participation Functions (b) Other Participation Functions

Fig. 4. Results for the SAT problem

Our new hybrid approaches the best Participation Function is EDA++, which gives a fitness value of 48000. With the DYNAMIC, 25% EDA / 74% GA and 50% EDA / 50% GA Participation Functions the results are also good.

Feature Subset Selection

In the FSS problem GAs performance is better than EDAs performance. However, the hybrid solution using 50% EDA / 50% GA is better than both of them.

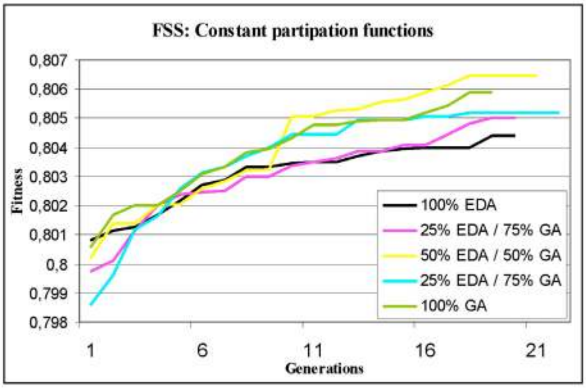


Fig. 5. Results for the FSS problem

4.2 Dynamic Participation Function: Evaluation

One of the most interesting aspects researched by this contribution is to know how the dynamic Participation Function performs for different kind of problems. This result provides an idea of how suitable is each of the methods for a specific kind of problem. And more useful, during the execution of the algorithm what is the performance base on the generation. Figure 6 shows the Percentage of GA participation in the Dynamic Participation Function for the four problems. In three of the four problems we can observe the same tendency, first we start to use the genetic algorithms, and after some generations the use of EDAs increase. This tendency is bigger in the problems in which GAs performs better than EDAs. However, it is necessary to remark that in the last generations, EDA algorithm always increases.

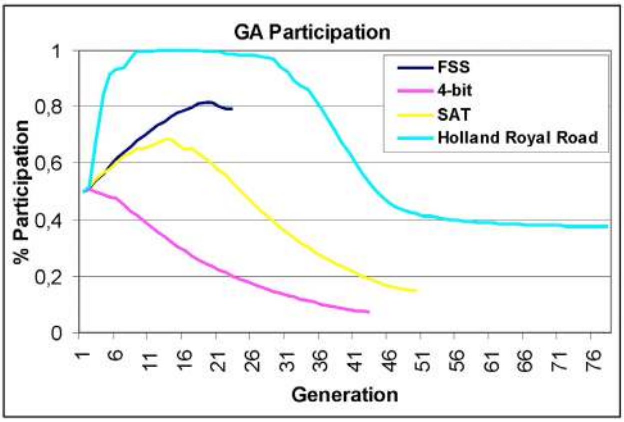


Fig. 6. Percentage of GA participation in the Dynamic PF

5 Conclusions and Future Work

In this paper we have proposed a new hybrid algorithm based on genetic and estimation of distribution algorithms. This new algorithm has been tested on four different problems: 4-bit fully deceptive function, Holland Royal Road, SAT problem and Feature Subset Selection. Although the hybrid algorithm proposed is composed by the simplest versions of both GA and EDA components and only works with bit-string individuals, the experimentation shows it is really promising and competitive. In most of the experiments we reach to the best of the values found by GAs or EDAs or even we improve them.

There is still a lot of further future work: Extend the implementation to support more sophisticated individual representations, for example with continuous genes, make new Participation Functions, make experimentation in more problems, implement a parallel version or use more complex GAs and EDAs in the hybrid solution.

Acknowledgements. The authors would like to acknowledge Raquel Hernández for her work programming and testing the first stages of this algorithm, as well as her useful comments.

References

[Choe & Choi & Moon, 2003] Choe, H. and Choi, S. and Moon, B. (2003). A Hybrid Genetic Algorithm for the Hexagonal Tortoise Problem, in GECCO 2003, Genetic and Evolutionary Computation Conference, Springer Verlag, 2003, number 2723 in Lecture Notes in Computer Science, pp. 850 - 861.

[Deb & Golberg, 1993] Deb, K., & Goldberg, D.E. (1993). Analyzing deception in trap functions. In Withley, L.D. (Eds.), *Foundation of Genetic Algorithms 2* (pp. 93-108). San Mateo, CA: Morgan Kaufmann.

[Duda & Hart, 1973] Duda, R. and Hart, P. (1973). Pattern Classification and Scene Analysis. *JohnWiley and Sons*, 1973.

[Goldberg, 1989] Goldberg, D. (1989). Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley, 1989.

[Hand & Yu, 2001] Hand, D.J. and Yu, K. (2001). Idiot's Bayes - not so stupid after all? *International Statistical Review*, 69(3):385–398, 2001.

[Hao & Lardeux & Saubion, 2002] Hao, J. and Lardeus, F. and Saubion, F. (2002). A Hybrid Genetic Algorithm for the Satisfiability Problem. 1rst International Workshop on Heuristics, Beinijg, 2002

[Holland, 1993] Holland, J. H. (1993). Royal Road functions. Internet Genetic Algorithms Digest, vol. 7, issue 22.

[Larrañaga & Lozano, 2001] Larrañaga, P. and Lozano, J.A. (2001). Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. *Kluwer Academic Publisher.*

[Lim & Rodrigues & Xiao, 2003] Lim, A. and Rodrigues, B. and Xio, F. (2003). Integrated Genetic Algorithm with Hill Climbing for Bandwidth Minimization Problem, in GECCO 2003, Genetic and Evolutionay Computation Conference, Springer Verlag, 2003, number 2724 in Lecture Notes in Computer Science, pp. 1594–1595.

[Liu & Motoda, 1998] Liu, H. and Motoda, H. (1998). Feature Selection for Knowledge Discovery and Data Mining. *Kluwer Academic Publisher*.

[Mitchell et al., 1992] Mitchell, M., Forrest, S. and Holland, J.H. (1992). The royal road for genetic algorithms: Fitness Landscapes and GA performance. In *Proceedings of the First European Conference on Artificial Life*. Cambridge, MA: MIT Press/Bradford Books, 1992.

[Mühlenbein, 1998] Mühlenbein, H. (1998). The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5:303–346, 1998.

[Murphy & Aha, 1995] P. M. Murphy, P.M. and Aha, D.W. (1995). UCI repository of machine learning databases. http://www.ics.uci.edu/~mlearn/

[Rodriguez-Tello & Torres-Jimenez, 2003] Rodriguez-Tello, E. and Torres-Jimenez, J. (2003) ERA: An algorithm for reducing the epistasis of SAT problems, in GECCO 2003, Genetic and Evolutionary Computation Conference, Springer Verlag, 2003, number 2724 in Lecture Notes in Computer Science, pp. 1283–1294.

[Whitley & Starkweather, 1990] Withley, D. and Starkweather, T. (1990). GENITOR II: a distributed genetic algorithm, *Journal of Experimental and Theoretical Artificial Intelligence*, 2, 189-214.

[Zhang & Sun & Tsang & Ford, 2003] Zhang, Q. and Sun, J. and Tsang, E. and Ford, J. (2003). Combination of Guided Local Search and Estimation of Distribution Algorithm for Quadratic Assignment Problems, in GECCO 2003: Proceedings of the Bird of a Feather Workshops, Genetic and Evolutionary Computation Conference, pp, 42 – 48.