Predictive measures for problem representation and genetic operator design

Dirk Thierens

institute of information and computing sciences, utrecht university
technical report UU-CS-2002-055
www.cs.uu.nl

Predictive measures for problem representation and genetic operator design

Dirk Thierens

Institute of Information and Computing Sciences
Utrecht University, The Netherlands

Abstract. The design of a problem representation and of the genetic operators acting on this representation are a very important task when applying genetic algorithms. To assist the user in this design task a number of predictive measures have been proposed in the literature. Here we discuss some issues in the use of these predictive measures. We identify the properties which a predictive measure should possess, and introduce an alternative procedure. The results represent a small step towards a predictive tool assisting the GA designer. The paper's main goal is to stimulate the discussion about the necessary ingredients of such a tool at the GECCO Workshop on Representations. To keep the discussion concrete we compute some predictive measures of different GA problem solving approaches for the Bin Packing problem.

1 INTRODUCTION

The efficient use of genetic algorithms (GAs) generally requires a well chosen problem encoding - or genotype representation - and accompanying genetic operators. It is important for the search efficiency that the operators generate genuinely new solutions from the current set in order to explore new parts of the search space. These new solutions should also - with high probability - have a higher fitness value than randomly generated solutions. These goals can only be accomplished when the generation of new solutions also preserves - with high probability - some of the information present in the parent solution which contributes to its fitness value. This requirement is rather easy to achieve with exploration operators that make only a small change to an individual solution. For algorithms such as local greedy search, simulated annealing, tabu search, and mutation-only evolutionary search, the representation and operator design problem is therefore not a major challenge, although its importance should never be underestimated. For recombinative genetic algorithms however, the design is a more substantial challenge. Due to the large exploratory steps performed by recombining two solutions, it is more difficult to ensure that useful fitness information contained in the parent solutions is transmitted to the children with high probability. This more challenging design task is the price we have to pay for the potentially more efficient search by the use of recombination operators.

The obvious question any (novice) GA designer asks himself is "how should one proceed to design such a representation and/or genetic operator". Unfortunately, there is no simple answer to this question, but there is enough known about the search characteristics of GAs, that some guidelines for the design of competent genetic algorithms can be given [15]. In spite of this knowledge though, it often remains necessary to explore and compare different alternatives for the representation and the operators. The straightforward - and most accurate - approach to achieve this, is to actually run the different GA alternatives. An important drawback is the computational cost, and several researchers have looked into cheaper mechanisms to evaluate alternative representations and operators. In the next section we will first discuss the properties such a mechanism should possess and look at some predictive measures found in the literature. Section 3 introduces an alternative mechanism for predicting the performance of different representations and/or genetic operators. In section 4 we illustrate these measures on the bin packing problem. Finally section 5 ends with a discussion.

2 PREDICTIVE MEASURES

The goal of computing predictive measures is to be able to tell how well certain design choices, such as solution representation and genetic operators, perform as compared to alternative choices without actual running the different GAs. Naturally, the computational effort spend in computing the predictive measures should be smaller than running the GAs. It might well be possible that certain measures that do require more computational effort than the actual GA runs, provide useful insights to help designing better problem encodings and genetic operators, but they should be looked upon as analysis tools not predictive tools.

In addition to the low computational cost requirement, there are a number of properties that should be fulfilled. First of all, the measure should be generally applicable. Predictive measures that can only be used for binary representations are less suitable since especially for non-binary representations there is a need for predictive measures. Examples of such representations are ordering problems, problem specific encodings for combinatorial problems, genetic programming, neural networks, directed acyclic graphs, ... The list can easily be extended, in fact any rich representation where it is not immediately clear how individual solutions can be recombined, or even mutated, is a candidate.

A second requirement is that one should not need to know the global optimum of the search space. For theoretical analysis on artificial problems this knowledge can prove to be very useful [10]. However, for all practical purposes, predictive measures that require such knowledge - or even an estimate - have limited value in practice. For instance predictive measures as the fitness distance correlation [8] [3], or the progress rate [4], all require to know the distance to the optimal value.

A number of predictive measures have been proposed in the literature. Manderick et al. proposed to measure the fitness correlation coefficient between the fitness of the parent(s) and the fitness of the child(ren) [16]. This idea elaborated on the work of Weinberger on correlated and uncorrelated fitness landscapes [9]. The key assumption is that a high fitness correlation is indicative of the fact that genetic operator preserves useful fitness characteristics between parents and offspring, which in turn should lead to an efficient search. Formally, let F_p be the mean fitness of the parents, and F_c the mean fitness of the children. The *operator fitness correlation coefficient* ρ_{op} is defined as

$$\rho_{op} = \frac{cov(F_p, F_c)}{\sigma(F_p)\sigma(F_c)},$$

where $\sigma(F_p)$ is the standard deviation of the fitness of the parents, $\sigma(F_c)$ the standard deviation of the fitness of the children, and $cov(F_p, F_c)$ the covariance between the parent fitness and offspring fitness. An estimate of this value can easily be computed by generating a large set of parent-offspring groups, and calculating the fitness values.

A second predictive measure is the *evolvability* or *expected improvement* which measures the ability of the evolutionary algorithm to produce new offspring with fitness better than existing individuals ([1],[2],[4],[6],[7],[11]). For a fitness function F (assuming maximisation) and a variation operator Γ, we have:

$$E_{imp} = E[F(\Gamma(x)) - F(x)].$$

A related measure is the *probability of improvement:*

$$P_{imp} = Prob[F(\Gamma(x)) > F(x)].$$

3 TREE COMPETITION

Manderick et al. computed the crossover fitness correlation coefficient ρ_{op} from a sample of randomly created parents. It should be noted though that it is reasonable to expect the statistical values to be dependent on the fitness of the parents. After all, it might be much more difficult to transmit the fitness determining pieces of the parent encoding to the offspring when the parents have high fitness, as opposed to simply randomly created parents. Unfortunately this creates a

dilemma: generating highly fit parents would require running the GA to find them, but this is exactly what we want to avoid by using the predictive measures. Here we generate highly fit parents by holding a tournament selection of size s. The parents chosen for computing the statistics are the best solutions within a randomly generated set of s solutions. This requires a large number of samples though. In addition, it does not take into consideration that the characteristics of for instance the best out of 8 individuals might be very different from the characteristics of the best individuals winning three consecutive tournament and offspring generation cycles.

Here we propose a scheme which should at least partially resolve some of the issues. We would like to emphasise though that this only represents some preliminary thoughts about how a predictive set of tools could be constructed. The predictive measures discussed so far all try to capture the relevant information in a single number. Recall that the goal of computing predictive measures is to allow making choices between alternative representations and genetic operators without actually running all the GAs. It is an open question whether such informative measures can be constructed. One way to get closer to our goal might be to abandon the idea of a single measure but instead focus at some simple graphs that could help the GA designer in making his choices.

As a first attempt we look at the following procedure:

1. Randomly generate a population of individuals of size 2^k with k typically between 9 and 12.
2. Create pairs and recombine them to create their offspring
3. Evaluate the fitness values and determine the best individual of each family of four.
4. Collect statistics for each family and its winning individual
5. The winning individuals go to the next level, thus forming a population of size 2^{k-1}.
6. Repeat the cycle until only one individual remains

At each cycle the number of individuals is thus reduced by a factor 2. This exponential reduction ensures that the computational cost remains low, while we still get an idea of the recombinative behaviour at later generations.

4 BIN PACKING

To illustrate the use of the use of predictive measures we have run some experiments with the Bin Packing problem. The Bin Packing problem is well suited for this purpose since it is a hard problem where a number of different representations and crossover operators have been proposed in the literature. In addition all efficient GA approaches are hybrid - this is, they use a domain specific heuristic to help Hybrid GAs are quite common in real world combinatorial optimisation problems.

In the Bin Packing problem we have a set of items $I = \{I_0, I_1, \ldots, I_{n-1}\}$ each having a particular size $S(I_i)$. The task is to partition the set of items in a set of exhaustive and mutually exclusive set of subsets (also called the bins) such that the sum of the sizes of the items in each subset does not exceed a fixed capacity C. The optimal solution is the partitioning or grouping that needs the smallest number of subsets. Counting the number of subsets is however a very uninformative measure since most solutions will rapidly consist of the optimal number of bins plus one. A more smoother fitness function is obtained by adding for each bin the squared of the ratio of the bin filling and its total capacity:

$$F(BP(I)) = \frac{1}{b} \cdot \sum_{i=1}^{b} \left(\frac{\sum_{j=1}^{g} S(I_j)}{C} \right)^2$$

The Bin Packing problem has been studied by a number of GA researchers [5], [12], [13]. Falkenauer made a very extensive study, and he argued that Bin Packing should be looked upon as a specific type of problem - namely, a grouping problem - that should benefit from using a problem encoding and genetic operators designed in accordance with the grouping characteristics [5]. He distinguished

three different types of genotype representations with their corresponding crossover operator: standard representation, order representation, and group representation. His experiments showed that the group encoding had the best performance, closely followed by the ordering representation, while the standard encoding was the worst. The interesting question to consider here is whether this information could also be learned by the use of predictive measures as opposed to actual running all the experiments.

We have calculated predictive measures for the triplet Bin Packing problem. This particular Bin Packing problem has been created to form a difficult benchmark problem. The idea is to draw item sizes in the range of $[250 \ldots 500]$ which have to be packed in bins with capacity 1000. A well-filled bin should contain one large item and two small items. The difficulty is due to the fact that two large items or three small items can be put together in one bin, but this will lead to a suboptimal solution. Instances of known optima can be generated as follows: first, an item is created with a size s_1 drawn uniformly at random within the interval $[380 \ldots 490]$. The second item is generated with a size s_2 drawn uniformly at random within the interval $[250 \ldots 1000 - \frac{s_1}{2}]$. The third item size s_3 is put equal to $1000 - s_1 - s_2$, therefore completing the bin to full capacity. It is easy to see that there is one large item, $s_1 \in [380 \ldots 490]$, and two small items, $s_2 \in [250 \ldots 310]$ and $s_3 \in [200 \ldots 370]$. To calculate the predictive measures we generated 50 triplets resulting in Bin Packing problem sets of 150 items.

4.1 MULTI-START FIRST FIT HEURISTIC

The GAs applied to Bin Packing all make use of a bin filling heuristic to help constructing a solution. It is therefore natural to consider how well one should do if only the heuristic was used. Fair comparisons can easily be made by simply generating the same number of children as with the GA, but instead of constructing the children from the parents one simply generates two solutions without considering the parents. Of course this will only be meaningful if the heuristic used has a randomised component, there is little value in generating the same solution over and over again. Here we use the First Fit heuristic that - given an ordering of the items - will simply place the items in the first bin where there still enough room for the item to be added. Given a random permutation of the items the First Fit heuristic will generate a different solution.

The following table shows the 'crossover correlation coefficient' for randomly generated parents ($s = 1$), and for parents that are the best out of a group of 10 ($s = 10$). Naturally, $\rho_{op} = 0$ since we not are not inheriting anything of the parents. The second table shows the percentage of improvements and the average fitness improvement. Note that for $s = 1$ the percentage is not 50% since this measure looks at the number of children (1 or 2) that are better than the best (out of 2) parents.

s	$\overline{F}_P$	$\overline{F}_C$	σ_P	σ_C	Cov_{PC}	ρ_{op}
1	0.80	0.80	0.013	0.013	0.000002	-0.01
10	0.83	0.80	0.008	0.013	0.0000005	0.00

Best of s	% improv.	expected improv.
1	34 %	0.0001
10	4 %	-0.025

Finally, figure 1 shows the best, median, and worst fitness for the tree-based procedure. This figure serves as a baseline for the following experiments, and give an indication how much is actually gained by the recombining the parents.

4.2 STANDARD REPRESENTATION

In the standard encoding the grouping is represented by a character string of fixed length n. Each character represents a particular group and the position in the string refers to the corresponding item: the group item I_i belongs to is specified at the i^{th} position in the string. For instance, the

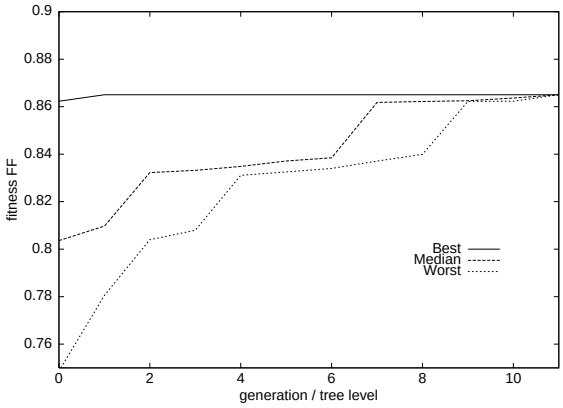

Fig. 1. Best, median, and worst fitness value for the First Fit heuristic (no recombination) for increasing number of elitist tournaments.

genotype string "BBACCAABCA" represents the subsets $\{I_2, I_5, I_6, I_9\}$ corresponding to character A, $\{I_0, I_1, I_7\}$ corresponding to character B, and $\{I_3, I_4, I_8\}$ corresponding to character C. Recombination can be done with a standard crossover operator such as two-point crossover.

The standard encoding as used above is a very redundant coding. For instance the string "BBACCAABCA" represents the same solution as "AABCCBBACB" or as "DDCAACCDAC". Recombining them however does most likely lead to different offspring. In previous work on the encoding of multi-layer perceptron neural networks we have encountered a similar problem [14]. It was shown there how the elimination of these types of redundancy improve the inheritance of useful information. The redundancy seen here can easily be eliminated by transforming all character encoding to one canonical form. This can be done for instance by demanding that the characters are used in alphabetical order and no letter should be passed over. In the above example the canonical encoding is thus "AABCCBBACB".

The standard encoding and two-point crossover does not prevent the offspring from violating the bin capacities. In principle one could use a penalty function to steer the GA towards valid solutions, but it should be clear that in practice this is a very inefficient method. An alternative is to remove the bins that are above capacity and fill in the missing items with the first fit heuristic. As one could expect two-point crossover keeps generating too many bins violating the capacity constraint, and the whole approach highly resembles the multi-start use of the first fit heuristic. To save space we do not show the table and figure here.

4.3 ORDER REPRESENTATION

A second type of problem encoding for the Bin Packing problem is to use a fixed length list of numbers from 0 to $n-1$ where different solutions are represented by a permutation of the numbers. The actual partitioning is computed by applying a grouping heuristic - such as the First Fit heuristic - to the list of numbers. For instance, the genotype (0 7 1 2 9 5 6 8 4 3) might lead to the subsets $\{I_0, I_1, I_7\}$, $\{I_2, I_5, I_6, I_9\}$, and $\{I_3, I_4, I_8\}$, depending on the heuristic used and on the item sizes and capacity of the bins. Recombination can be done by any permutation based crossover operator. Here we have used the $C1$-ordering crossover: a random crosspoint is chosen and all items before the crosspoint are copied to the two offspring [13], [12]. The remaining items are filled in the order they occur at the other parent.

The crossover correlation coefficient clearly indicates that information about the fitness function is actually transmitted from the parents to the children. Surprisingly, the percentage of offspring that has higher fitness than the best parent is actually lower for randomly generated parents, but when the parents have higher fitness (winning a tournament of size $s = 10$), the percentage of improvement becomes more favourable than without recombination.

s	$\overline{F}_P$	$\overline{F}_C$	σ_P	σ_C	Cov_{PC}	ρ_{op}
1	0.80	0.80	0.013	0.013	0.00009	0.54
10	0.83	0.82	0.008	0.012	0.00003	0.34

Best of s	% improv.	expected improv.
1	27 %	-0.001
10	16 %	-0.007

Figure 2 shows the best, median, and worst fitness for the tree-based procedure, while figures 4 and 5 compare the different schemes.

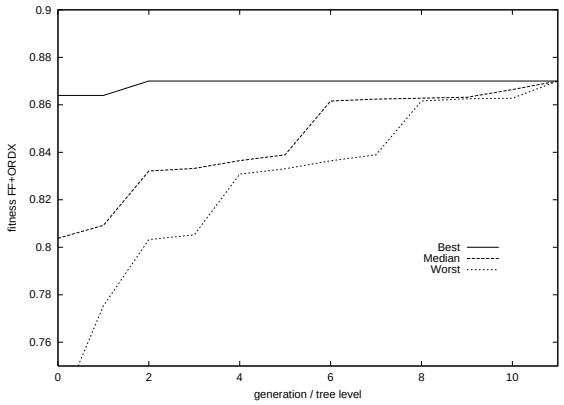

Fig. 2. Best, median, and worst fitness value for the Ordering representation, C1 crossover operator, and First Fit heuristic for increasing number of elitist tournaments.

4.4 GROUP REPRESENTATION

Finally, the grouping representation tries to stay as close to the characteristics of the underlying problem as possible. The genotype is a variable length list of lists representing directly the grouping. For instance, a possible representation of the subsets $\{I_0, I_1, I_7\}$, $\{I_2, I_5, I_6, I_9\}$, and $\{I_3, I_4, I_8\}$ is simply $((I_0\ I_1\ I_7)\ (I_2\ I_5\ I_6\ I_9)\ (I_3\ I_4\ I_8))$. The crossover operator however is now more complicated. Two crossing sites are chosen at random in both parents. The items between the crosspoints are inserted at the first crossing point of the other parent. Some of the items now appear twice, therefore the non-inserted groups that contain an item which is also present in the inserted groups are removed from the offspring. The missing items are then reinserted by the First Fit heuristic.

Computing the crossover correlation coefficient shows that the grouping representation transmits slightly more fitness related information from the parents to the children. The improvement measures give a more or less similar picture.

s	$\overline{F}_P$	$\overline{F}_C$	σ_P	σ_C	Cov_{PC}	ρ_{op}
1	0.80	0.80	0.013	0.014	0.00012	0.62
10	0.83	0.82	0.008	0.013	0.00004	0.40

Best of s	% improv.	expected improv.
1	27 %	-0.003
10	18 %	-0.007

Figure 3 shows the best, median, and worst fitness for the tree-based procedure. As before figures 4 and 5 compare the different schemes.

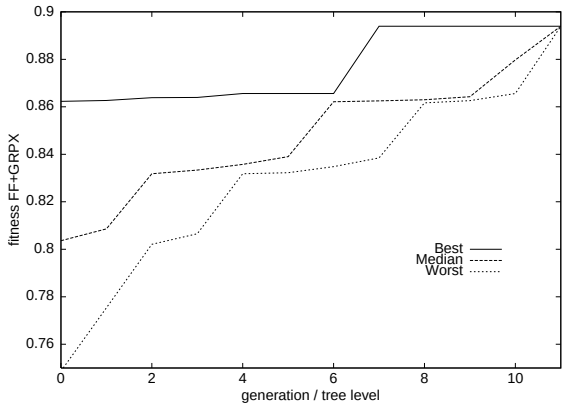

Fig. 3. Best, median, and worst fitness value for the Grouping representation, group crossover operator, and First Fit heuristic for increasing number of elitist tournaments.

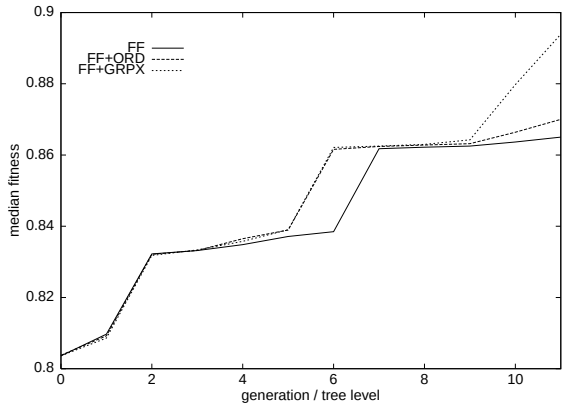

Fig. 4. Median fitness values for the three representations and operators.

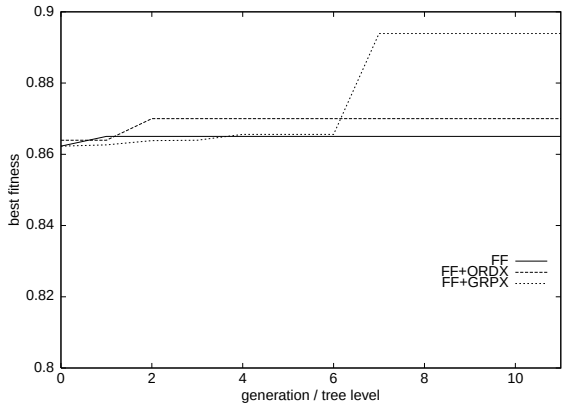

Fig. 5. Best fitness values for the three representations and their operators.

5 DISCUSSION & CONCLUSION

From the predictive measures considered so far it seems that the median fitness plot of the tree-based procedure and the crossover correlation coefficient give the most predictive information. In figure 6 we have plotted the percentage of recombinations where one of the offspring was better than the best parent for each 'generation' in the tree-based procedure. Figure 7 shows the average fitness improvement when an actual improvement did take place. Both figures show the benefit of crossover above the multi-start heuristic searcher. The grouping representation seems to have the edge above the ordering representation which is in agreement with Falkenauer's experiment.

One should be careful with interpreting the experimental results though. One thing that is certainly missing right now is the statistical significance of all these results. Whereas the median value is rather robust, significance tests should be applied, especially for the higher 'generations' in the tree-based procedure, since at these levels only a few individuals are still under consideration. Current work is underway to incorporate this.

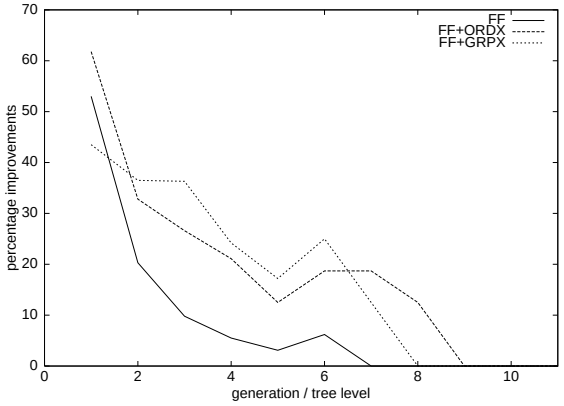

Fig. 6. Percentage of improvements at each tournament for the three representations and their operators.

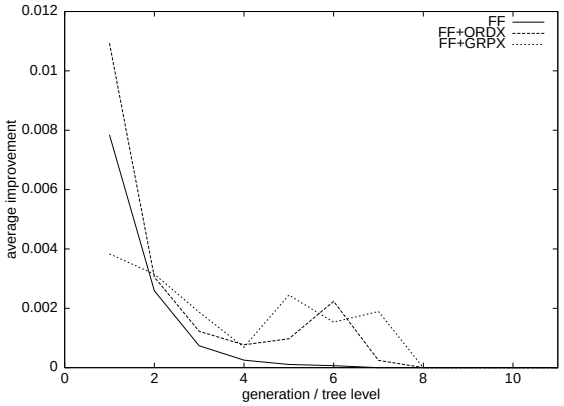

Fig. 7. Average improvement at the tournaments with an actual improvement for the three representations and their operators.

It should be clear that this work is in a preliminary phase and represents only a small step towards a fully functional tool to guide the GA designer in the representation and operator choices she has to make. We do believe however that such a tool will consist of a number of information rich graphs as opposed to a single number measuring one specific aspect. More studies are needed though to better understand their strengths and weaknesses, and to come up with a clear picture of what can and what cannot be concluded from specific estimates of particular predictive measures.

References

1. Lee Altenberg. The evolution of evolvability in genetic programming. In Kenneth E. Kinnear, editor, *Advances in Genetic Programming*, pages 47–74. MIT Press, Cambridge, MA, 1994.
2. Lee Altenberg. Evolving better representations through selective genome growth. In *Proceedings of the 1st IEEE Conference on Evolutionary Computation. Part 1 (of 2)*, pages 182–187, Piscataway N.J., 1994. IEEE.
3. Lee Altenberg. Fitness distance correlation analysis: An instructive counterexample. In Thomas Bäck, editor, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 57–64, San Mateo, CA, 1997. Morgan Kaufmann.
4. Hans-Georg Beyer. Towards a theory of evolution strategies: Self-adaptation. *Evolutionary Computation*, 3(3):311–347, 1995.
5. E. Falkenauer. A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics*, 2:5–30, 1996.
6. David B. Fogel and Adam Ghozeil. Using fitness distributions to design more efficient evolutionary computations. In *International Conference on Evolutionary Computation*, pages 11–19, 1996.
7. Christian Igel and Martin Kreutz. Using fitness distributions to improve the evolution of learning structures. In *1999 Congress on Evolutionary Computation*, pages 1902–1909, Piscataway, NJ, 1999. IEEE Service Center.
8. Terry Jones and Stephanie Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In Larry Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 184–192, San Francisco, CA, 1995. Morgan Kaufmann.
9. B. Manderick, M. de Weger, and P. Spiessens. The genetic algorithm and the structure of the fitness landscape. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 143–150, San Mateo, CA, 1991. Morgan Kaufmann Publishers.
10. B. Naudts and L. Kallel. A comparison of predictive measures of problem difficulty in evolutionary algorithms. *IEEE transactions on evolutionary computation*, 4:1–15, 2000.
11. Ingo Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. frommann-holzbog, Stuttgart, 1973.
12. Colin R. Reeves. Hybrid genetic algorithms for bin-packing and related problems. *Annals of OR*, 63:371–396, 1996.
13. D. Smith. Bin packing with adaptive search. In J.J. Grefenstette, editor, *Proceedings of an International Conference on Genetic Algorithms and their applications*. Lawrence Erlbaum Associates, 1985.
14. Dirk Thierens. Non-redundant genetic coding of neural networks. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pages 571–575. IEEE Press, 1996.
15. Steven van Dijk, Dirk Thierens, and Mark de Berg. On the design and analysis of competent GAs. forthcoming, 2002.
16. E.D. Weinberger. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics,*, 63:325–336, 1990.