Evolvability Analysis of the Linkage Tree Genetic Algorithm

Dirk Thierens[1] and Peter A.N. Bosman[2]

[1] Institute of Information and Computing Sciences
Universiteit Utrecht, The Netherlands
D.Thierens@uu.nl
[2] Centre for Mathematics and Computer Science
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
Peter.Bosman@cwi.nl

Abstract. We define the linkage model evolvability and the evolvability-based fitness distance correlation. These measures give an insight in the search characteristics of linkage model building genetic algorithms. We apply them on the linkage tree genetic algorithm for deceptive trap functions and the nearest-neighbor NK-landscape problem. Comparisons are made between linkage trees, based on mutual information, and random trees which ignore similarity in the population. On a deceptive trap function, the measures clearly show that by learning the linkage tree the problem becomes easy for the LTGA. On the nearest-neighbor NK-landscape the evolvability analysis shows that the LTGA does capture enough of the structure of the problem to solve it reliably and efficiently even though the linkage tree cannot represent the overlapping epistatic information in the NK-problem. The linkage model evolvability measure and the evolvability-based fitness distance correlation prove to be useful tools to get an insight into the search properties of linkage model building genetic algorithms.

1 Introduction

Linkage learning genetic algorithms aim to identify interacting or dependent problem variables that contribute to highly fit solutions. The goal is to build a linkage model of these interactions and use this model to generate, with high probability, new highly fit solutions.

To better understand how this class of algorithms searches for optimal solutions we introduce two measures: the linkage model evolvability and the evolvability-based fitness distance correlation. These measures have their origin in the evolvability measure and fitness distance correlation coefficient for general evolutionary algorithms. We exploit the particularities of linkage models to design more informative evolvability and correlation measures.

The paper is organized as follows. In the next section we describe the linkage tree learning and the optimal mixing evolutionary algorithm. Section 3 introduces the linkage model evolvability measure and the evolvability-based fitness distance correlation. In Section 4 we compute these measures for the LTGA on a deceptive trap function and a nearest-neighbor NK-landscape problem. Finally, Section 5 concludes the paper.

2 Linkage Tree Genetic Algorithm

We give a short review of the LTGA, for more details we refer the reader to [9].

The linkage tree (LT) is a hierarchical linkage model obtained by a bottom-up agglomerative hierarchical clustering algorithm starting from the set of problem variable singletons [2,8]. For a problem of length ℓ the linkage tree has ℓ leaf nodes (the clusters having a single problem variable) and $\ell-1$ internal nodes. The similarity measure is for instance the normalized mutual information between two subsets of variables. An efficient method to compute the similarity measure is the average linkage clustering or unweighted pair group method with arithmetic mean (UPGMA) [6,9]. Given a population of size N the LT can be built in $\mathcal{O}(N\ell^2)$ time using the reciprocal nearest-neighbor chain algorithm.

The LT specifies a set of problem variable subsets which are a specific example of the more general family of subsets (FOS). The class of linkage model building GAs we consider here specify their linkage model by this FOS model. Mathematically, the FOS model is a subset of the power set of the problem variables. This FOS model is used by the Gene Pool Optimal Mixing Evolutionary Algorithm (GOMEA) to generate new solutions [9]. Each subset of problem variables is used as a crossover mask. Each solution of the population is iteratively used as parent solution. For each parent solution the entire FOS set is traversed: for each problem variable subset in the FOS model a random solution is picked from the population as donor. The donor's values of the problem variables specified by the subset - or crossover mask - are copied to the parent solution. This new solution is evaluated and when it is an improvement of the parent, the offspring replaces the parent. Next , the traversal of the FOS model continues with the new solution. If there was no fitness improvement the FOS model is further traversed using the parent solution. New solutions are thus only accepted when they have a better fitness value than the parent solution. When the tree is completely traversed, the current parent solution is copied to the next generation's population. This tree traversal process is done for each solution in the current generation. The LT has $2\ell-2$ subsets in the tree (the top node is ignored because it contains all problem variables), so in one generation there are at most $N(2\ell-2)$ offspring generated and evaluated. This value is an upper boundary because before evaluating a new solution we always first check whether the generated solution is really different from the parent solution.

The reason for calling this operation optimal mixing is due to its inception in the original LTGA [8] where a two-parent crossover operator was used following the LT with intermediate evaluations to check for improvements. Given only two parents, traversing a FOS while performing crossover then ensures that all building blocks as described in the FOS wind up in one of the two parents and mixing therefore can be said to be optimal. However, it was recently shown that considering a randomly selected new donor parent for each element in the FOS while performing crossover, which was called Gene-pool Optimal Mixing (GOM) is overall a more efficient manner of mixing because this removes the covariance in mixing that is inherent in using the same two parents for the entire crossover operation [9].

The use of OM allows making use of the linkage information as provided by all subsets in the FOS. OM further strongly increases the selection pressure in a building-block-wise manner instead of on a entire solution basis, the latter of which is a source of noise in deciding well between building blocks that increases the population size. It is through OM that the LTGA is capable of working with very small population sizes compared to most linkage learning EAs of the Estimation of Distribution Algorithm (EDA) type.

3 Linkage Model Evolvability Analysis

When applying a genetic algorithm - and more generally, any metaheuristic search algorithm - to a specific problem, we often would like to get a clear picture of how the search dynamics proceeds. We would like to possess a few measures that can give us a better understanding of what is going on during the search. Ideally, these measurements would help us explain why a certain algorithm succeeds in finding good solutions while others do not. In this paper we will use four measurements: two existing and two that we define specifically for linkage model building GAs where the model is a family of subsets (FOS).

Hamming Distance. The first and most obvious measure to trace the convergence progress is the minimum and median hamming distance from the solutions in the current population towards the global optimum.

Evolvability. The ultimate goal of search operators like crossover or mutation is to create new solutions that have a better fitness than their parent(s). The probability that this occurs has been called *probability of success* in the evolution strategy literature, while Altenberg called this the *evolvability* [1]. For an evolutionary algorithm to be successful it is important that the evolvability remains positive when the search has reached the higher regions of the fitness landscape, or when the parents have high fitness values. To compute this we split up the higher fitness values in bins and count the frequency of generating fitter offspring when the parent has a fitness value within the fitness bin's range. The parents are taken from actual runs, not from random samples. This is important as the key question here is whether the evolvability remains positive as the search approaches the most fit solutions in the search space.

Linkage Model Evolvability. The evolvability as a function of the parental fitness does not give any insight in the contribution of the different masks in the linkage FOS model. To investigate their importance in the search process we calculate the evolvability as a function of the size of the masks - or FOS subsets - of successive linkage trees during an actual LTGA run. For the benchmark functions used in this paper, the successive application of five linkage trees is sufficient to reliably find the global optimum, at least if the model learning is done properly. In an actual LTGA run each mask of the tree is used N times (N being the population size): for each solution in the population the linkage tree is traversed and a random solution is picked as donor. Counting the evolvability based on these offspring represents a limited sample of the actual evolvability potential of the linkage tree for the current population. In order to get a less noisy measurement we have calculated the evolvability by taking each solution in the population as the donor solution, as opposed to a single random solution. This way each mask is used N^2 times and the number of improvements are counted. We also compute the relative number of improvements as the percentage of offspring more fit than their parent (excluding the donor) of all the offspring generated by a mask of a given size using a particular linkage tree. It is important to note that this calculation of the linkage masks' evolvability has no influence on the actual LTGA run. The offspring generated during this calculation are not used in the actual LTGA run.

Evolvability-Based Fitness Distance Correlation. Although the evolvability measures the potential of an evolutionary algorithm to keep finding new and better solutions it does not consider whether the population is actually converging towards the global optimal solution. An EA could well be capable of generating many better offspring during the search process but that is not a guarantee that it is actually getting any closer to the optimum. To measure the progress towards the optimum Jones and Forrest [4] introduced the concept of fitness distance correlation (FDC). As its name implies the FDC measures the correlation between the fitness value and the hamming distance towards the global optimal solution. A large negative correlation is seen as an indication that the GA would be guided towards the optimum by following a path of ever improving solutions. A large positive correlation is interpreted as a deceiving problem: following a path of better solutions would lead away from the optimal solution. The FDC is actually a search ignorant measure in the sense that it does not include any information about the capability of the genetic operators to generate improving solutions. FDC only looks at the representation and the fitness values of the solutions, not at the actual dynamics of the GA run. There is little to be gained from a high negative FDC measure if the genetic operators are unable to generate the high fitness solutions in the first place. Altenberg [1] discussed these limitations of a Hamming-distance based FDC and proposed two crossover-distance based FDC measures (XFDC). The XFDC measures aim to include the role of the genetic operators. The first XFDC defined the crossover distance as the number of discontinuities between 0s and 1s in a solution's bit-string. This measure is clearly only suitable for the specific test function used in that paper which is based directly on this number of discontinuities. A second more general crossover-based distance measure is computed by running crossover in reverse. Starting from the global optimum and its binary complement bit-strings are given a crossover distance of 1 when they are generated by a single application of the crossover operator. A second set of strings is given a distance of 2 by applying crossover to the previous set. Continuing this way a sequence of populations is generated with increasing crossover distance. The XFDC measure is now computed by calculating the correlation coefficient between the crossover distance and the fitness value. It is clear that both XFDC measures are only rough approximations of the actual GA dynamics. The main problem of getting a more accurate measure is the vast amount of different crossover events that are possible.

For the LTGA however, the number of possible crossover events is much smaller. In fact, when computing the linkage model evolvability, we are already looking at all possible outcomes for a specific linkage FOS model and a given population. The only thing we need to add is the correlation with approaching the optimal solution. Therefore we define the evolvability-based fitness distance correlation (EFDC) as the correlation between the Hamming distance between the offspring and the optimal solution and the amount of fitness gain whenever an improvement occurred during the calculation of the linkage masks' evolvability. The EFDC is a much more informative measure of the search dynamics than the FDC or XFDC measures.

In [3] the fitness distance correlation is computed for fixed neighborhoods that match the structure of the fitness function. The EFDC however is computed during the search and depends on the specific linkage tree built each generation, thus capturing more of the dynamics of the search process.

4 Experimental Analysis

To test whether the evolvability measures do indeed provide any insight in the search behavior of the LTGA we compare the linkage tree with a randomly build tree on 2 benchmarks. First, we consider the deceptive trap function [2]. This function is interesting for our purposes here because a linkage learning algorithm must be able to learn the structure of the problem in order to find the optimal solution. A randomly constructed tree will be unable to do this, and the interesting question is how this gets reflected in the evolvability measures. Our second test function is the nearest-neighbor NK-landscape [7]. This function is interesting because the overlap of the subfunctions cannot be represented by a linkage tree, and yet the LTGA is capable of consistently finding the optimal solution. The key question is whether the evolvability measures can help explain why this is the case.

Evolvability measures should give insight in the particular behavior of an actual GA run. We therefore compute the values on one single run, and not average them out over a whole set of runs. Of course, this run should be representative and we compared the results on different runs. As it turned out, all runs had basically the same behavior and similar evolvability values, so we only report here the results of one single run for both the benchmark functions.

4.1 Deceptive Trap Function

The deceptive trap function used here consists of 10 subfunctions of length 5 (stringlength $\ell = 50$), fitness value of the subfunctions is 5 for the optimum and 4 for the deceptive attractor. The population size is 100. The initial population is generated by performing a single-pass bit-flip local search on a population of random solutions.

Table 1 shows the Hamming distance between the optimal string and the best and the median solution in the population for successive linkage trees. When the linkage tree is learned using normalized mutual information, the population quickly converges to the optimal solution. Both the minimum and median Hamming distance are reduced by each new generation, and the optimal solution is generated at the fifth generation. The table also shows the Hamming distance when the linkage tree is built using random numbers as similarity measure instead of normalized mutual information. Clearly, without linkage learning the search algorithm does not make a lot of progress in finding the optimal solution. Only looking at the Hamming distance however does not make it clear whether the search is not making progress at all or it is simply going the right direction but at a very small pace.

Table 1. Hamming distance towards the global optimum for the deceptive trap function

distance	**Linkage tree**					**Random tree**				
	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
minimum	25	20	15	5	0	30	30	30	30	30
median	45	40	30	20	10	45	45	40	40	35

Table 2 shows the frequency of improvements as a function of parental fitness. The fitness is divided in 10 bins, spanning the fitness range from the solution with all deceptive attractors to the global optimum. Clearly, the search without linkage learning is not able to make any substantial progress towards the optimal solution. The probability of generating better offspring is nearly zero, and no offspring with a fitness value of 45 or higher are created. When linkage learning is done properly we see that better offspring are created in a consistent way, and the evolvability remains positive with increasing parental fitness values.

The linkage model evolvability in Table 3 gives a more detailed picture of the evolvability in the linkage tree. The table only shows the masks where a fitness improvement takes place. For the linkage learning the masks' sizes are all multiples of five which reflects the building block length of the deceptive trap function. The percentage of improvements is quite high for all the masks in the table, indicating a very efficient search process.

For the random tree - this is, no linkage learning - there are very few fitness improvements, and the vast majority of them are achieved with crossover masks of length 48 and 49. For a stringlength of size 50 this basically means that the donor solution is better than the parent and the large masks are simply making an almost complete copy of the donor. Obviously, this does not lead to good novel solutions.

Table 2. Evolvability: frequency of improvements as a function of parental fitness

Fitness	**Linkage tree**					**Random tree**				
range	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
$[40, 41[$	0.29	0.27	0.26	0.22	–	0.03	0	0	–	–
$[41, 42[$	0.10	0.16	0.20	0.23	–	0	0	0	0	0.01
$[42, 43[$	0.01	0.05	0.15	0.21	0.17	0	0	0	0	0.01
$[43, 44[$	0.03	0.02	0.05	0.18	0.24	0	0	0	0	0
$[44, 45[$	0.01	0.02	0.03	0.12	0.19	–	0	0	0	0
$[45, 46[$	0	0.01	0.02	0.04	0.15	–	–	–	–	–
$[46, 47[$	–	0	0.01	0.03	0.14	–	–	–	–	–
$[47, 48[$	–	–	0	0.02	0.08	–	–	–	–	–
$[48, 49[$	–	–	–	0.01	0.02	–	–	–	–	–
$[49, 50[$	–	–	–	0	0.01	–	–	–	–	–

Table 3. Linkage model evolvability for the deceptive trap function

Linkage tree										
Mask size	Improvements					% improvements				
	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
5	10892	16808	21916	22648	16192	11%	17%	22%	23%	16%
10	7176	9474	8995	12726	7462	18%	24%	30%	32%	25%
15	5152	6032	10080	3661	5668	26%	30%	34%	37%	29%
20	2147	3318	3677	3846	–	22%	33%	37%	39%	–
30	–	–	3712	3931	3720	–	–	37%	40%	38%
35	3285	3663	–	–	–	33%	37%	–	–	–
40	–	–	–	–	3975	–	–	–	–	40%
45	–	–	–	4112	4002	–	–	–	41%	40%
Tot.	28652	39295	48380	50924	41019					

Random tree										
Mask size	Improvements					% improvements				
	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
22	269	–	–	–	–	1.3%	–	–	–	
29	–	62	–	–	–	–	0.01%	–	–	
39	–	–	130	–	110	–	–	0.01%	–	0.01%
41	–	–	–	40	–	–	–	–	0.00%	
42	–	–	–	40	–	–	–	–	0.00%	
43	–	–	130	–	–	–	–	0.01%	–	
44	–	–	–	–	310	–	–	–	–	0.03%
48	–	–	–	–	3351	–	–	–	–	34%
49	2095	–	–	–	–	21%	–	–	–	
Tot.	2364	62	260	80	3771					

Finally, Table 4 shows the EFDC measure. In case of linkage learning there is a perfect linear relationship between the amount of fitness improvement and the reduction in Hamming distance towards the optimal solution. This makes sense as the crossover masks exactly match the building blocks and fitness improvements of 1, 2, 3, ... correspond to Hamming distance reductions of 5, 10, 15, ... When there is no linkage learning there is usually also no correlation coefficient to compute since there are either zero or only one single pair of fitness improvement and Hamming distance value. Only in the first random linkage tree large masks can sometimes get a fitness improvement.

Table 4. Evolvability-based Fitness Distance Correlation for the deceptive trap function

Linkage tree					**Random tree**				
Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	–	–	–	–

4.2 Nearest-Neighbor NK-Landscape

The nearest-neighbor NK-landscape used here has stringlength $\ell = 50$, the subfunctions have length 5 bits, and the overlap is maximal - that is, 4 bits. The population size is $N = 200$.

Table 5. Hamming distance towards the global optimum for the NK-landscape

distance	**Linkage tree**					**Random tree**				
	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
minimum	7	3	0	0	0	7	7	7	7	7
median	20	18	15	12	6	21	20	19	19	18

Table 6. Evolvability: frequency of improvements as a function of parental fitness

Fitness	**Linkage tree**					**Random tree**				
Bins	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
1	0.21	–	–	–	–	0.06	–	–	–	–
2	0.17	–	–	–	–	0.08	–	–	–	–
3	0.16	0.26	–	–	–	0.06	0.09	–	–	–
4	0.10	0.12	–	–	–	0.04	0.04	0.05	–	–
5	0.07	0.11	0.10	–	–	0.03	0.03	0.03	0.02	0.00
6	0.04	0.08	0.06	0.10	–	0.02	0.02	0.02	0.02	0.02
7	0.04	0.04	0.05	0.10	0.13	0.03	0.01	0.01	0.01	0.01
8	0.02	0.02	0.03	0.05	0.11	0.00	0.00	0.01	0.00	0.01
9	0.00	0.02	0.02	0.03	0.06	–	–	–	–	–
10	–	0.01	0.02	0.02	0.02	–	–	–	–	–

As for the deceptive trap function, we look at the impact of linkage learning by comparing the results with a randomly constructed linkage tree. A linkage tree has $2\ell - 2$ nodes that are used as crossover masks.

Table 5 shows the Hamming distance for the linkage tree and the random tree models. As shown in previous work the linkage tree has no trouble in finding the optimal solution, while the random tree is getting nowhere.

Table 6 shows the evolvability as a function of the parental fitness. We have divided the fitness range between the fitness value *low* and the global optimal value in 10 bins of equal width. The fitness value *low* is the fitness of the least fit solution of a population of 100 single-pass bit-flipped solutions. Whenever an improving solution is generated the bin corresponding to the parent's fitness is updated. The evolvability values for the linkage tree model shows that the successive linkage trees are capable of generating new and more fit solutions with increasing parental fitness. On the contrary, the random trees are not able

Table 7. Linkage model evolvability for the NK-landscape

Linkage tree										
Mask size	Improvements					% improvements				
	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
1	31205	13358	6169	3650	2765	1.5%	0.6%	0.3%	0.2%	0.1%
2	29523	12345	5638	4019	3943	3.9%	1.6%	0.8%	0.5%	0.5%
3	17308	5446	4273	3694	1354	4.8%	1.7%	1.2%	1%	0.5%
4	12409	5278	6732	6479	6953	7.8%	3.3%	4.2%	4%	2.9%
5	14053	12519	8730	6580	2044	7.1%	6.3%	5.5%	5.5%	2.6%
6–10	36673	34266	18561	12871	16274	16%	11%	7%	5%	5.7%
11–20	16686	25813	33354	24045	48664	14%	34%	16%	22%	20%
21-30	6530	14800	–	46270	–	16%	37%	–	29%	–
31–40	14412	15726	15621	–	12502	36%	40%	39%	–	31%
41–50	–	–	–	–	–	–	–	–	–	–
Tot.	178799	139551	99078	107608	94499					
Random tree										
Mask size	Improvements					% improvements				
	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
1	41040	23516	14635	9587	7081	2.1%	1.2%	0.7%	0.5%	0.4%
2	16361	7792	6361	3330	2228	2.9%	1.4%	1.2%	0.5%	0.4%
3	7914	4648	4043	1333	1180	3.3%	1.7%	1.3%	0.4%	0.6%
4	4247	1246	3409	1355	1040	2.7%	1.6%	1.4%	0.9%	0.4%
5	2632	1859	–	1019	328	2.2%	1.6%	–	0.9%	0.3%
6–10	5246	4276	2738	2167	309	1.3%	0.9%	0.8%	0.6%	0.1%
11–20	1233	1121	821	586	241	0.4%	0.3%	0.3%	%	0.2%
21–30	98	279	448	287	567	0.2%	0.3%	0.4%	0.4%	0.3%
31–40	510	–	–	–	503	1.3%	–	–	–	0.6%
41–50	12323	6350	10226	–	10733	31%	16%	13%	–	6.7%
Tot.	91604	51087	42681	19664	24210					

Table 8. Evolvability-based Fitness Distance Correlation for the NK-landscape

Linkage tree					**Random tree**				
Tree 1	Tree 2	Tree 3	Tree 4	Tree 5	Tree 1	Tree 2	Tree 3	Tree 4	Tree 5
-0.26	-0.44	-0.47	-0.58	-0.41	-0.27	-0.13	-0.21	0.07	-0.32

to generate improving solutions above a certain fitness level. Successive trees are not able to generate new solutions that fall in the higher valued bins. The search clearly stagnates in the lower valued fitness bins.

In Table 7 we see the evolvability contributions of different crossover masks. The linkage tree model has a much higher evolvability than the random tree in both absolute as relative measures. It is interesting to see that all mask sizes contribute to the search, which shows that the linkage tree's capability of representing interacting problem variables at multiple levels is beneficial to the search.

Finally, Table 8 shows the evolvability-based fitness distance correlation. For the five successive linkage trees the EFDC remains significantly negative, meaning that fitness gains are correlated with reductions in Hamming distance to the optimal solution. For the random linkage trees the EFDC are also negative but have a lower value, except for the first generation trees. It appears that in the first generation the solutions can be easily improved and the Hamming distance towards the optimal solution is reduced. If we look again at Table 7 we see that most of these improvements for the random tree are obtained with masks of length

1 and 2, so the offspring are only 1 or 2 bits different from to their parents but do have a higher fitness and are mostly closer in Hamming distance to the optimal bitstring. It is also noteworthy that the EFDC for the fifth tree of the random models has a rather high correlation ($= -0.32$). Looking again at Table 7 reveals that almost half of the improvements are obtained by masks of size larger than 40: the high correlation can thus be explained by the copying effect of good donor solutions by large masks. Although these improvements increase the EFDC value they do not significantly contribute to finding new good solutions.

5 Conclusion

We have analyzed the evolvability of the linkage tree genetic algorithm. For this, we have defined the linkage model evolvability and the evolvability-based fitness distance correlation. We have seen how these measures give an insight in the performance of the LTGA. We have also made a comparison with a randomly constructed tree and discussed the differences observed in the evolvability measures. On a deceptive trap function, the measures clearly show that learning the linkage tree makes this an easy problem for the LTGA. On the nearest-neighbor NK-landscape the evolvability analysis shows that the LTGA does capture enough of the structure of the problem to solve it reliably and efficiently even though the linkage tree cannot represent the overlapping epistatic information in the NK-problem. We believe that measures like the linkage model evolvability and the evolvability-based fitness distance correlation are useful tools to describe and understand the characteristics of linkage model building genetic algorithms.

References

1. Altenberg, L.: Fitness distance correlation analysis: An instructive counterexample. In: Proc. 7th Intern. Conf. Genetic Algorithms, pp. 57–64. Morgan Kaufmann (1997)
2. Duque, T.S., Goldberg, D.E.: A new method for linkage learning in the ECGA. In: Proc. 11th Int. Conf. Genetic and Evol. Computation, pp. 1819–1820. ACM (2009)
3. Hauschild, M., Pelikan, M.: Advanced neighborhoods and problem difficulty measures. In: Krasnogor, et al. (eds.) [5], pp. 625–632. ACM (2011)
4. Jones, T., Forrest, S.: Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In: Proceedings of the Sixth International Conference on Genetic Algorithms, pp. 184–192. Morgan Kaufmann (1995)
5. Krasnogor, N., Lanzi, P.L. (eds.): Proc. 13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011. ACM (2011)
6. Pelikan, M., Hauschild, M., Thierens, D.: Pairwise and problem-specific distance metrics in the linkage tree genetic algorithm. In: Krasnogor et al. [5], pp. 1005–1012
7. Pelikan, M., Sastry, K., Goldberg, D.E., Butz, M.V., Hauschild, M.: Performance of evolutionary algorithms on NK-landscapes with nearest neighbor interactions and tunable overlap. In: Raidl, G. (ed.) GECCO, pp. 851–858. ACM (2009)
8. Thierens, D.: The Linkage Tree Genetic Algorithm. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN XI. LNCS, vol. 6238, pp. 264–273. Springer, Heidelberg (2010)
9. Thierens, D., Bosman, P.A.N.: Optimal mixing evolutionary algorithms. In: Krasnogor and Lanzi [5], pp. 617–624