Analysis of NK-xor landscapes

C.H.M. van Kemenade

Software Engineering (SEN)

SEN-R9840 December 31, 1998

Report SEN-R9840
ISSN 1386-369X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

Analysis of NK-xor Landscapes

Cees H.M. van Kemenade
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

ABSTRACT
The NK-xor landscapes are a class of optimization problems with adjustable ruggedness of its fitness landscape. The optimal value for each of the loci is dependent upon a k-neighbourhood of this locus. A set of these landscapes is analyzed and the evolution of certain generic algorithms when applied to these fitness-landscapes is approximated by means of a transmision-function model.

1991 Mathematics Subject Classification: 68T05, 68T20
1991 Computing Reviews Classification System: F.2.m, G.1.6, I.2.8
Keywords and Phrases: genetic algorithms, NK-landscapes, selection schemes, transmission functions
Note: Work carried out under theme SEN4 "Evolutionary Computation". Parts of this report were published in the proceedings of the IEEE World Congress on Computational Intelligence/IEEE Conference on Evolutionary Computation.

1. Introduction

To get a better understanding of the dynamical properties of genetic algorithms we are looking for problem instances that are simple enough to be analyzed theoretically. Simultaneously these problems have to be complex enough to observe the kind of behavior of the GA that we are expecting on practical optimization problem. A lot of theory has been developed for different variants of the oneMax problem, also called the counting-ones problem; to mention a few [KF95, RW91, TG93]. A typical property of this problem is that all bits can be optimized independently of one another. More complex problems on which the behavior of the GA is studied, are the deceptive trap-functions [DG94, GDKH93, vK97].

Both problems adhere to the building block hypothesis, which states that a GA looks for a set of short, low order building blocks in parallel and then combines these building blocks to obtain a good solution. Furthermore both problems are separable in the sense that the global optimum can be partitioned in a set of non-overlapping building blocks, such that there is no interaction between the loci belonging to different parts of the partition.

The building block hypothesis does not state that building blocks should not overlap. Even though the class of separable problems is a interesting class of problems in itself, it is interesting to look for other problems instances that do allow overlap of building block. In this paper we are going to focus on problems having interactions between all bits. An example of such problems are the NK-landscapes [Kau93]. A class of NK-landscapes is obtained by choosing a value for n, the number of bits in a bit-string, and a value for k in the range 0 to $n-1$. With each bit in the string we associate a neighbourhood of k other bits. Two types of NK-landscapes exist. The first type uses nearest-neighbour interactions, which means that the k-neighbourhood of a bit is given by the k nearest loci on the bit-string. The second type of NK-landscapes use a random neighbourhood for each of the bits. Finding the highest peak is easier for the first type of NK-landscapes as the neighbourhoods of two bits will only overlap if the distance on the chromosome is less than $\lceil \frac{k}{2} \rceil$. Given a class of NK-landscapes we generate a specific instance of a NK-landscape by generating a matrix $\mathbf{R}$ of size $n \times 2^{k+1}$ of random real values in the range [0,1]. The fitness of a bit-string is given by the average fitness-contribution of all of the bits. The fitness-contribution of a bit i is computed by concatenating

π_2	5	3	4	1	2
π_1	3	4	2	5	1

bits	1	0	1	1	0

Figure 1: Example NK-xor landscape for $k = 2$.

the value of the bit and its k neighbours, resulting in an integer value j in the range 0 to $(2^{k+1} - 1)$. The fitness-contribution of bit i is now given by matrix-element $\mathbf{R}[i, j]$.

Each optimal assignment to a locus and its k-neighbourhoods can be considered as a building block. Such a building block will overlap with approximately $k + 1$ other building blocks. Many of these building-blocks will overlap. The building block hypothesis might be applicable to this problem as it does have $(k + 1)^{th}$-order building blocks, but the large amount of overlap between building blocks might also result in a different way of processing information. The NK-landscapes have been analyzed themselves [HW97] but an analysis of the behavior of a GA on such a complex problem might be infeasible. In this paper we introduce a variant, the NK-xor landscapes, that have many properties in common with Kauffman's NK-landscapes, but are more easy to analyze.

The outline of the paper is as follows. In section Section 2 introduces the NK-xor landscapes. The application of crossover is analyzed in section 3. We are going to use transmission-function models to analyse the behavior of GA's on this problem. These models are introduced in section 4 while the actual implementation details are discussed in section 5. The results obtained by tracing the models are presented in section 6 followed by the conclusions and some directions for further research in section 7.

2. NK-xor problem

The NK-xor landscapes are a class of problems quit similar to Kauffman's NK-landscapes. The fitness of the NK-xor landscapes is computed as the sum of the fitness of the fitness-contribution of its n bits, and the fitness of a bit is dependent upon a k-neighbourhood of other bits. A specific instance out of this problem class is obtained by selecting k random permutations $\pi_1, \pi_2, ..., \pi_k$, where each permutation consist of a single cycle. Now the neighbourhood of bit i is defined as the set of bits $(\pi_1(i), \pi_2(i), ..., \pi_k(i))$. An example of a problem instance is given in Figure 1. The fitness contribution of bit i is set to 1 if

$$b_i = b_i \ xor \ b_{\pi_1(i)} \ xor \ b_{\pi_2(i)} \ xor \ ... \ xor \ b_{\pi_k(i)}$$

and 0 otherwise. Here b_j denotes bit j and xor denotes the exclusive-or operation. The truth-table of this operation is:

xor	0	1
0	0	1
1	1	0

The fitness of a complete bitstring is computed as the sum of the individual fitness contributions of each of the bits. The fitness of two bit-strings at Hamming-distance one can differ at most $k + 1$, so larger values of k can result in more rugged landscapes. For odd k a bit-string s will have the same fitness as its bit-wise complementary string, so these NK-xor landscapes will always have an even number of global optima. Furthermore there is no preference for either a 0-bit or a 1-bit at any locus.

3. Crossover on the NK-xor landscapes

In the rest of this paper we are going to study NK-xor landscapes with $k = 1$. Each problem-instance for $k = 1$ and even n has two optimal solutions with fitness equal to n. When observing the bits in the order defined by the permutation π_1 an alternating sequence of bits is obtained.

Assume that we have two parent individuals P_1 and P_2 having respectively fitness f_1 and f_2. Now we can compute the expected fitness of the offspring obtained when applying uniform crossover to this pair of parents. For each bit b_i we have to discriminate between three different cases.

The first case is that b_i has a fitness contribution of one in both of its parents. This case has probability $f_1 f_2/n^2$. The fitness contribution of b_i in the offspring is 1 if b_i and b_{π_i} are taken from the same parent or when the bits are taken from different parents, but the value of b_i is the same for both parents. Hence the probability that the fitness contribution of b_i is 1 in the offspring is $\frac{1}{2} + (1 - \frac{1}{2})\frac{1}{2} = \frac{3}{4}$.

The second case is that b_i has fitness contribution 1 in exactly one of its parents. This case has probability $(f_1/n)(1 - f_2/n) + (1 - f_1/n)(f_2/n) = (f_1 + f_2)/n - f_1 f_2/n^2$. Now either $b_i^{(1)} = b_i^{(2)}$ or $b_{\pi(i)}^{(1)} = b_{\pi(i)}^{(2)}$. Let k be this location where both parents match. Hence b_i of the offspring will have fitness contribution 1 when both bits are taken from the parent where b_i has fitness contribution one, or when only b_k is taken from the parent where b_i has fitness contribution 0. In this case the probability that b_i has fitness contribution 1 is $(\frac{1}{2})^2 + (\frac{1}{2})^2 = \frac{1}{2}$.

The third case is that b_i gives no fitness contribution in either of the parents. This case has probability $1 - (f_1 + f_2)/n - f_1 f_2/n^2$. If the string $b_i b_{\pi_i}$ is the same for both parents than the fitness contribution of b_i in the offspring is 0. If the strings are different then these strings have to be complementary and the fitness contribution of b_i will be 1 with probability $\frac{1}{2}$. The third case results in a fitness contribution of 1 for b_i with probability $(\frac{1}{2})0 + (1 - \frac{1}{2})\frac{1}{2} = \frac{1}{4}$.

The expected fitness of the offspring is the given by the following formula:

$$\begin{aligned} f_o = n\Big(& (f_1 f_2/n^2)\tfrac{3}{4} + \\ & ((f_1 + f_2)/n - f_1 f_2/n^2)\tfrac{1}{2} + \\ & (1 - (f_1 + f_2)/n - f_1 f_2/n^2)\tfrac{1}{4}\Big) \end{aligned}$$

This corresponds to $f_o = (f_1 + f_2 + n)/4$. The expected fitness of a random string is $n/2$. If at least one of the parents has $f > n/2$ then we see

$$\begin{aligned} f_o &\leq \max\{f_1, f_2\}/2 + n/4 \\ &< \max\{f_1, f_2\} \end{aligned}$$

So the uniform operator is quite disruptive in this case and the expected fitness of the offspring is lower than the expected fitness of the fittest parent.

4. Transmission function models

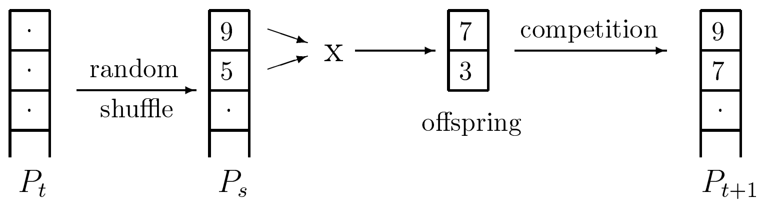

Figure 2: Schematic representation of Elitist recombination

We are going to consider the canonical genetic algorithm [Hol75], the generational genetic algorithm using tournament selection [Gol89], and elitist recombination [TG94]. In order to describe the different models we are going to use transmission functions [Alt94]. The general form of transmission-selection recursion was used at least as early as 1970 by Slatkin [Sla70].

The action of a genetic operator can be represented by a transmission function. Such a transmission function describes the probability distribution of offspring from every possible mating. For a binary genetic operator the transmission function looks like $T(i \leftarrow j,k)$ where j and k are the labels of the two parents and i is the label of the offspring. To be more specific, let $\mathcal{S}$ be the search space; then $T : \mathcal{S}^3 \rightarrow [0,1]$. As T represents a distribution we should have $\sum_i T(i, \leftarrow j,k) = 1$ for all $j,k \in \mathcal{S}$. For a symmetric operator we have the additional condition $T(i \leftarrow j,k) = T(i \leftarrow k,j)$.

Regarding transmission functions Altenberg [Alt94] states the following:

> It is the relationship between the transmission function and the fitness function that determines GA performance. The transmission function "screens off" [Sal71, Bra90] the effect of the choice of representation and operators, in that either affect the dynamics of the GA only through their effect on the transmission function.

The canonical genetic algorithm is a generation GA using fitness proportional selection. This dynamical system can be described by [Alt94]:

$$x_i' = \sum_{j,k} T_o(i \leftarrow j,k) \frac{f_j f_k}{\bar{f}^2} x_j x_k,$$

where x_i is the frequency of type i, and x_i' is its frequency during the next generation, f_i is the fitness of type i, $\bar{f}$ is the average fitness, and T_o is the transmission function describing the actual interaction between genetic operators and representation.

A generational GA using tournament selection can be described by [vK97]:

$$x_i' = \sum_{j,k} T_o(i \leftarrow j,k) P_{tour}^{(n)}(j) P_{tour}^{(n)}(k),$$

where $P_{tour}^{(n)}(j)$ describes the probability that the individual with label j is selected during a t-tournament. A t-tournament selection is performed by choosing t individuals uniform at random from the population and selecting the one having the highest fitness. In case of a tie we assume that the individual which was chosen first wins. Given the distribution of the current population we can compute this probability following the sequence of choice events that give the selected individual:

$$P_{tour}^{(n)}(j) = \sum_{t=1}^{n} P_<(j)^{t-1}(j) x_j P_\leq(j)^{n-t-1},$$

where $P_<(j)$ denotes the probability that an individual selected uniform at random has a fitness lower than individual j.

Elitist recombination [TG94] selects parents uniformly at random. Figure 2 shows how the next population P_{t+1} is produced from the current population P_t. On the left we see the current population P_t, where each box represents a single individual. The values in the boxes denote the fitness of the corresponding individuals. An intermediate population P_s is generated by doing a random shuffle on P_t. Population P_s is partitioned in a set of adjacent pairs and for each pair apply the recombination operator in order to obtain two offspring. Next a competition is held amongst the two offspring and their two parents, and the two winners are transferred to the next population P_{t+1}. In the presented example one parent and one offspring are transferred to P_{t+1}. Elitism is used as parents can survive their own offspring. The dynamical system representing elitist recombination is [vK97]:

$$x_i' = \sum_{j,k} T_{er}(i \leftarrow j,k) x_j x_k.$$

The selection of survivors is not visible anymore in this representation. This selection mechanism consists of a local competition between parents and their direct offspring and therefore is located in the transmission function T_{er}.

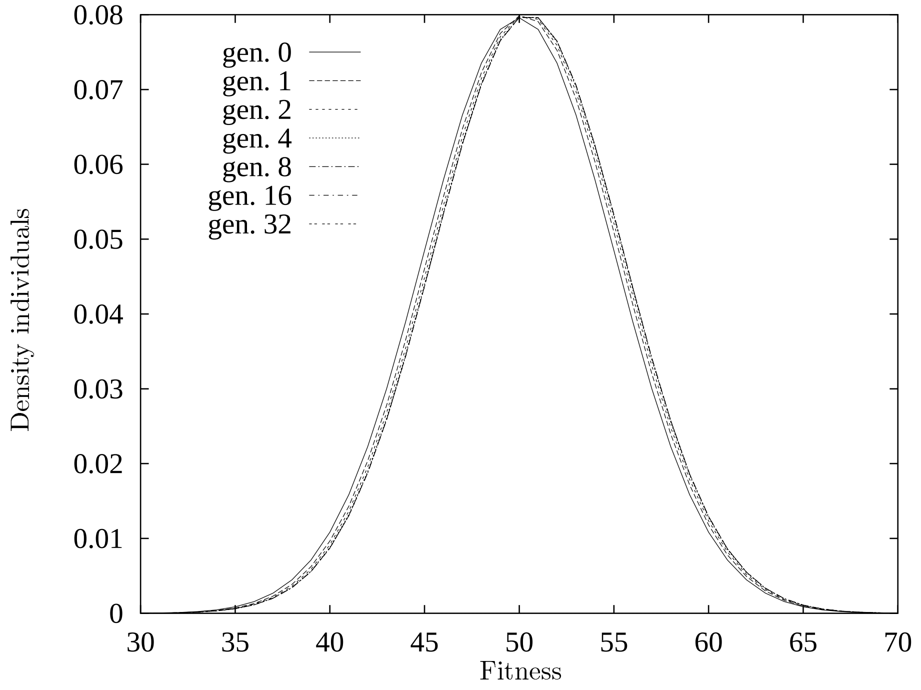

Figure 3: Distribution of the population for a set of generations when using a generational GA with proportional selection.

In the rest of this section we denote the binomial distribution by

$$B(n,k,p) = \left(\begin{array}{c} n \\ k \end{array} \right) p^k (1-p)^{n-k}$$

where n is the total number of experiments, k is the number of successful outcomes and p is the probability of a successful outcome.

For the elitist recombination and its generalization that creates more than two offspring for each parent pair, a modified transmission function $T_{er}(i \leftarrow j,k)$ is used as the selection of survivors is done by means of a local competition between parents and offspring. This in contrast to the usual selection mechanisms that operate on the complete population. Such a local selection scheme can be modelled by modifying the original transmission function $T_o(i \leftarrow j,k)$ that describes the interaction between the operators and the representation used. The new transmission function $T_{er}(i \leftarrow j,k)$ is obtained by modifying each column of the corresponding transition matrix.

Let $P_{accept}(P_<, P_=, n, a)$ denote the probability that an offspring is selected given that the proportion of offspring having smaller fitness $P_<$,the proportion of offspring having equal fitness $P_=$, the number of additional offspring generated n, and the number of offspring that can be accepted apart from the current offspring a. This function can be computed by first conditioning on the number of superior offspring detected followed by a conditioning on the number of offspring having equal fitness

$$\begin{array}{l} P_{accept}(P_<, P_=, n, a) = \\ \quad \sum_{l=0}^{n-1} B(n,l,1-P_< - P_=) \sum_{m=0}^{n-l} \\ \qquad B\left(n-l, m, \frac{P_=}{P_< + P_=}\right) \; \min\{1, \frac{a-l-1}{m+1}\} \end{array}$$

This expression forms the basis of our further computations.

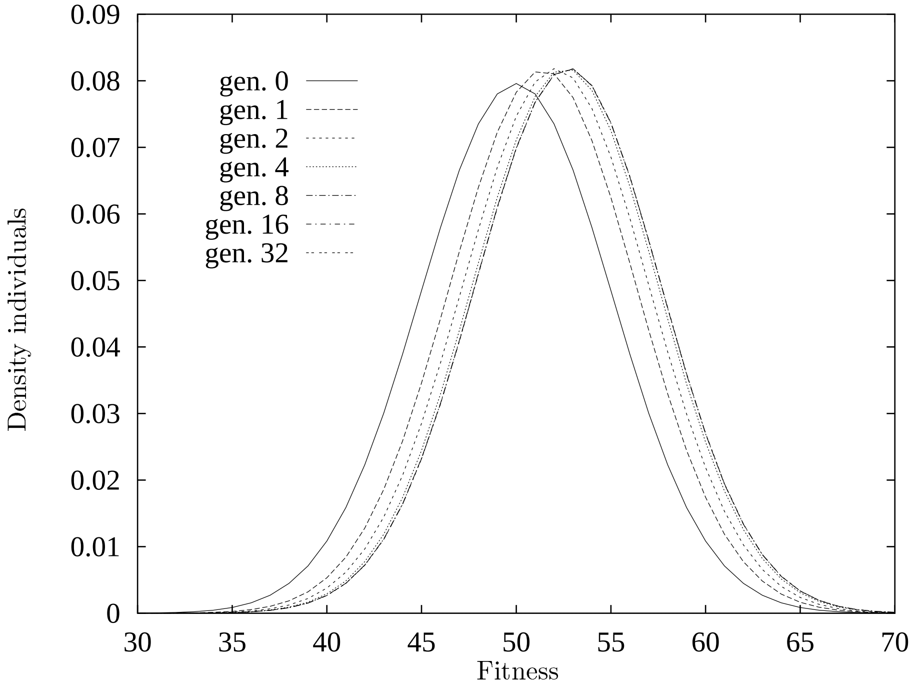

Figure 4: Distribution of the population for a set of generations when using a generational GA with tournament selection.

A column of the matrix describing $T_o(i \leftarrow j,k)$ represents the probability density function (also called the mass function) over the space of all possible offspring i for a given pair of parents j and k. Let f_i denote the fitness of individual i and label the parents such that $f_J \geq f_K$, then the offspring can be divided among three sets:

$$\begin{aligned} S_I &= \{i \in \mathcal{S} : f_i \geq f_J\} \\ S_{II} &= \{i \in \mathcal{S} : f_K \leq f_i < f_J\} \\ S_{III} &= \{i \in \mathcal{S} : f_i < f_K\} \end{aligned}$$

Given a column from $T_o(i \leftarrow j,k)$ we can compute the corresponding column from $T_{er}(i \leftarrow j,k)$ as follows. Let us assume that c_i is an element from the original matrix T_o and d_i the corresponding element in the new matrix T_{er}, both corresponding to state i. The probability than an offspring of type i is obtained by recombination is given by c_i. The probability that the offspring is also accepted when applying elitist recombination with n offspring for each pair of parents depends on the set S_x, where $x = I, II$ or III, that offspring i belongs to. If $i \in S_I$ then an offspring is accepted if at most one offspring having higher fitness is produced,

$$d'_i = c_i n P_{accept}(P_<(i), P_=(i), n-1, 1)$$

where $P_<(i)$ denotes the probability that a produced offspring is inferior to offspring i and $P_=(i)$ denotes the probability that a produced offspring has equal fitness. If $i \in S_{II}$ then the probability that an offspring is accepted depends on the number of offspring in region S_I and can be computed according to the formula

$$\begin{aligned} d'_i = c_i n \textstyle\sum_{l=0}^{1} & B(n-1, l, 1-P_<(J)) \\ & P_{accept}\left(\frac{P_<(i)}{P_<(J)}, \frac{P_=(i)}{P_<(J)}, n-l-1, 0\right). \end{aligned}$$

If $i \in S_{III}$ then $d'_i = 0$ as the individual i will always be rejected.

Furthermore the possibility to retain parents must be accounted for. Parent K will only be retained if all offspring belongs to to set S_{III}. This corresponds to the substitution

$$d'_K \leftarrow d'_K + B(n, n, P_<(K))$$

Parent J is retained when all offspring is in $S_{II} + S_{III}$ or when one offspring is in set S_I and all other offspring are in set S_{III}. This corresponds to the substitution

$$\begin{aligned} d'_J \leftarrow d'_J + B(n, n, P_<(J)) + \\ B(n, n-1, P_<(J))\, B\left(n-1, n-1, \tfrac{P_<(K)}{P_<(J)}\right) \end{aligned}$$

The vector $\vec{d'}$ corresponds to the density of the two offspring that result from a elitist tournament. The actual values of d_i are computed by normalizing this vector ($d_i = \frac{1}{2}d'_i$).

5. Tracing the models

In order to trace the models we must implement the transmission function $T_o(i \leftarrow j, k)$ that has been defined in section 4. Normally the transmission function is defined as an operator over the complete search space $T : \mathcal{S}^3 \rightarrow [0, 1]$. Tracing the evolution of such a dynamical system will require a lot of computation. When tracing the evolution for the problem we are studying here we can get a much more efficient method by mapping the original search space $\mathcal{S}$ to a more compact space $\mathcal{V}$ where each element of $\mathcal{V}$ represents an equivalent class containing a set of elements from $\mathcal{S}$ and then to define the transmission function on the space of equivalence classes $\mathcal{V}$. Of course we must adjust the distribution of the initial population in order to account for the differences in cardinality of the sets that are represented by the elements of $\mathcal{V}$. For the problem we are studying we can define a set of equivalence classes $\mathcal{V}$. The space $\mathcal{V}$ contains n elements, where n is the number of bits in a bit-string. Element v_i of $\mathcal{V}$ represents all bit-string where exactly i bits deliver a non-zero fitness contribution. All bit-strings in the same equivalence class have the same fitness. When using uniform crossover it will be possible to use a transmission function on $\mathcal{V}^3$ instead of a function on $\mathcal{S}^3$.

When using a random initial population, which corresponds to a uniform distribution over $\mathcal{S}$ the distribution in $\mathcal{V}$ can be modelled by means of the binomial distribution. The expected fraction of the initial population that falls in equivalence class v_i is given by $B(n, i, \frac{1}{2})$, where n is the size of an individual.

The transmission function can be represented by a $(n \times n^2)$-matrix of transmission probabilities. We can compute the element $T_o(i \leftarrow j, k)$ using the probabilities we have calculated in section 3. Note that an offspring of type j has exactly j bits with a fitness contribution of 1, so its fitness is j. Given the parents of type j and k we can define three sets a, b, and c, where a denotes the set of bits where both parents have a fitness contribution, b denotes the set of bits where exactly one parent has a fitness contribution, and set c covers the remaining bits. Let n_x denote the number of bits in set x. Taking the formulas given in section 3 and substituting f_1 by j and f_2 by k, we get $n_a = jk/n$, $n_b = j + k - 2jk/n$, and $n_c = n - j + k + jk/n$. Furthermore we know that the probability that an bit in the offspring has fitness contribution 1 is $p_a = \frac{3}{4}$, $p_b = \frac{1}{2}$, and $p_c = \frac{1}{4}$. Given these numbers we can compute

$$\begin{aligned} T(i \leftarrow j, k) = \textstyle\sum_{k_a=0}^{i} \sum_{k_b=0}^{i-k_a} \\ B(n_a, k_a, p_a) B(n_b, k_b, p_b) B(n_c, i - k_a - k_b, p_c) \end{aligned}$$

One minor complication arises due to the fact that jk/n usually does not have an integer value. Therefore we replace the above computation by a weighted average of four similar computations where the term jk/n in the formula for n_a and n_c are replaced by $\lfloor jk/n \rfloor$ and $\lceil jk/n \rceil$ and n_b is adjusted accordingly. For example using $\lfloor jk/n \rfloor$ in the formula of n_a and $\lceil jk/n \rceil$ in the formula of n_c results in $n_b = j + k - \lfloor jk/n \rfloor - \lceil jk/n \rceil$, and this branch has weight $p(1-p)$, where p is $\lceil jk/n \rceil - jk/n$.

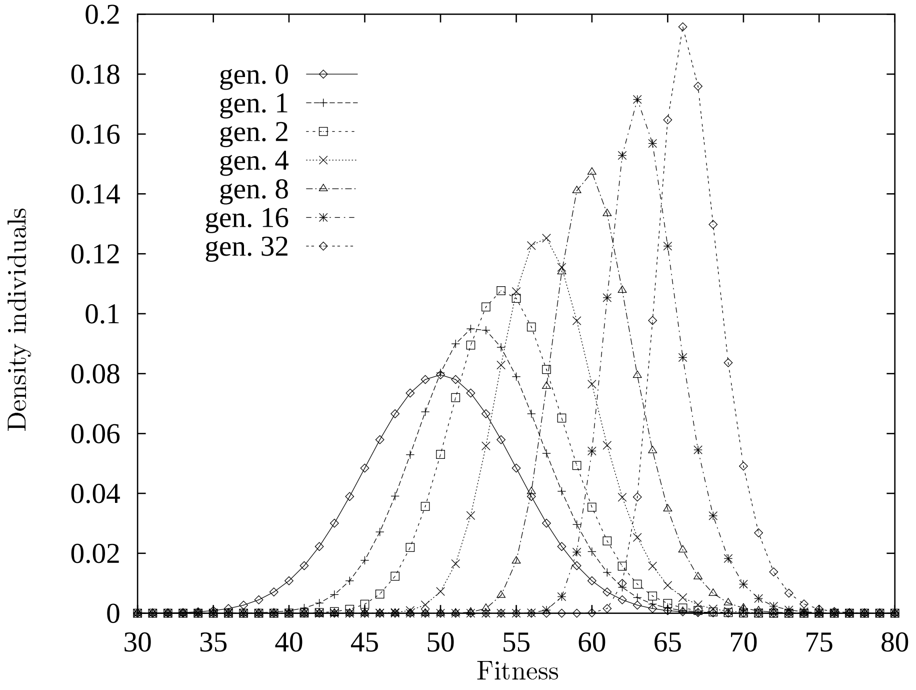

Figure 5: Distribution of the population for a set of generations when using elitist recombination.

6. Results

We have modelled the expected distribution of the population for a generational GA and the elitist recombination on the NK-xor landscape for $k = 1$ and $n = 100$. The models approximate the behavior of GA's with an infinite population size. The results are shown in Figures 3, 4 and 5. The density of the population as a function of the fitness is shown for generation 0, 1, 2, 4, 8, 16, and 32. Generation 0 corresponds to the random initial population.

Figure 3 shows the results for a generational GA with fitness proportional selection and crossover probability 1.0. An individual having fitness 60 will only get 1.2 copies after selection, which is not very much. This GA converges to a fixed point quite rapidly. For this distribution the low selective pressure of the proportional selection is balanced by the disruptive effect of the crossover operator that was discussed in section 3.

Figure 4 shows the results for a generational GA with tournament selection using tournament-size 2. Again the distribution converges to a fixed point with a relatively low average fitness. Increasing the tournament size will shift the population towards higher fitness values, but large tournament sizes, although applicable for the NK-xor problem, are likely to give rise to premature convergence on many other optimization problems.

Figure 5 shows the results for elitist recombination. This GA performs much better than the two generational GA's. The elitist recombination is not sensitive to the low performance of the crossover operator as a parent will only replaced by offspring having a higher fitness, which results in population-wide elitism. The irregular shape of the distribution for last generations is due to the fact that we can only discriminate between n equivalence classes where n is the length of a single individual.

7. Conclusions

The NK-xor landscapes are an interesting class of problems that have many properties in common with NK-landscapes, but are easier to analyse theoretically. Another interesting property of the NK-xor landscapes is that for odd k there is no specific preference for either the 0 or the 1-bit when observing the complete landscape. So if we apply a GA with a finite population and we observe that the 0 and the 1 value have different probabilities then we know that this convergence is due to the context introduced by the population or genetic drift.

The uniform crossover is not very efficient on the NK-xor landscapes and we need a high selective pressure or elitism to get convergence towards the global optimum.

The transmission functions are powerful tools to model genetic algorithms and compare them. Tracing such models corresponds to running the corresponding genetic algorithm with an infinite population size. Results on such runs give an upper bound on the reliability of the corresponding real genetic algorithm.

Further research should involve the investigation of the evolution of GA's for more rugged NK-xor landscapes having $k > 1$.

References

[Alt94] L. Altenberg. The evolution of evolvability in genetic programming. In Kenneth E. Kinnear, Jr., editor, *Advances in Genetic programming*, chapter 3, pages 47–74. M.I.T. Press, 1994.

[Bäc97] Th. Bäck, editor. *Proceedings of the 7th International Conference on Genetic Algorithms.* Morgan Kaufmann, 1997.

[Bra90] R.N. Brandon. *Adaption and Environment.* Princeton University Press, 1990.

[DG94] K. Deb and D.E. Goldberg. Sufficient conditions for deceptive and easy binary functions. *Annals of Mathematics and Artificial Intelligence*, 10:385–408, 1994.

[For93] S. Forrest, editor. *Proceedings of the 5th International Conference on Genetic Algorithms.* Morgan Kaufmann, 1993.

[GDKH93] D.E. Goldberg, K. Deb, H. Kargupta, and G. Harik. Rapid, accurate optimization of difficult problems using the fast messy genetic algorithms. In Forrest [For93], pages 56–64.

[Gol89] D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[Hol75] J.H. Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* The University of Michigan Press/Longman Canada, 1975.

[HW97] R.B. Heckendorn and D. Whitley. A walsh analysis of NK-landscapes. In Bäck [Bäc97], pages 41–48.

[Kau93] S.A. Kauffman. *The origins of order.* Oxford University press, New York/Oxford, 1993.

[KF95] J.N. Kok and P. Floréen. Tracing the behavior of genetic algorithms using expected values of bit and walsh products. In S. Forrest, editor, *Proceedings of the 6th International Conference on Genetic Algorithms*, pages 201–208. Morgan Kaufmann, 1995.

[RW91] Y. Rabinovich and A. Wigderson. An analysis of a simple generic algorithm. In R.K. Belew and L.B. Booker, editors, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 215–221. Morgan Kaufmann, 1991.

[Sal71] W.C. Salmon. *Statistical Explanation and Statistical Relevance.* University of Pittsburgh Press, 1971.

[Sla70] M. Slatkin. Selection and polygenic characters. In *Proceedings of the National Academy of Science U.S.A. 66*, pages 87–93, 1970.

[TG93] D. Thierens and D.E. Goldberg. Mixing in genetic algorithms. In Forrest [For93], pages 38–45.

[TG94] D. Thierens and D.E. Goldberg. Elitist recombination: an integrated selection recombination GA. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, pages 508–512. IEEE Press, 1994.

[vK97] C.H.M. van Kemenade. Cross-competition between building blocks, propagating information to subsequent generations. In Bäck [Bäc97], pages 1–8.