A general theory of genetic algorithms

A general theory of genetic algorithms

by

E.H.L.Aarts, A.E.Eiben, K.M. van Hee

89/08

December, 1989

COMPUTING SCIENCE NOTES

This is a series of notes of the Computing Science Section of the Department of Mathematics and Computing Science Eindhoven University of Technology. Since many of these notes are preliminary versions or may be published elsewhere, they have a limited distribution only and are not for review. Copies of these notes are available from the author or the editor.

Eindhoven University of Technology
Department of Mathematics and Computing Science
P.O. Box 513
5600 MB EINDHOVEN
The Netherlands

Editors: prof.dr.M.Rem
prof.dr.K.M. van Hee

A GENERAL THEORY OF GENETIC ALGORITHMS

E.H.L. AARTS †, ‡

A.E. EIBEN †

K.M. VAN HEE †

October, 1989

† Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513 5600 MB Eindhoven, The Netherlands

‡ Philips Research Laboratories
P.O. Box 80000 5600 JA Eindhoven, The Netherlands

ABSTRACT

The idea – and the name – of genetic algorithms (GAs) originates from biology. Applying certain biological principles (crossover, survival of the fittest) they form a robust tool kit to handle mathematical optimization problems. Many practical results have proved their usefulness, but still, there is no concise theory of GAs. This paper has a double objective: to work out an abstract concept of GAs and establish general convergence results. To reach the first goal we define a stochastic algorithm AGA that generalizes and unifies genetic algorithms and simulated annealing. For the second goal we model the search process of AGA by a Markov chain and set up conditions that imply convergence with probability 1.

Keywords: discrete optimization, genetic algorithm, simulated annealing, Markov chain.

0. INTRODUCTION

Genetic algorithms were developed in the early and mid seventies [Hol75], and for about a decade they remained rather unobserved. In the last years, however, they are gaining interest, the research community of GAs is growing and carrying out promising investigations [Gol89]. An important reason for this increasing popularity is that GAs are general, with a wide range of applicability and good performance results [Gre85, Gre87, Gre89]. They even succeed on problems where no specialized methods score good [Gol89]. Nevertheless, most of the existing work is practical; little effort was done to investigate GAs theoretically. In this article we want to make a step towards a future theory of GAs.

In nature the development of a certain population depends on two major factors: how individuals are born, and how they die. In simple terms, children are produced by recombining the parents' gen patterns, the new pattern (genotype) determines a new being. Dying of the individuals is due to their fitness, unfit elements cease existing. The mathematical problem of having a search space S and an object function f resembles the biological situation. To traverse the space in order to find a minimum of f requires generating and eliminating elements of S. In GAs the search space consists of tuples, new elements are produced by applying a certain crossover operation to the 'parent' tuples, elimination is depending on the object function value. The major –but often hidden– idea behind the use of GAs is that of inheritance. Roughly speaking one figures that 'strong' individuals get more children than 'weak' ones, and that the 'strength' of the parents is inherited by the children. This is the mechanism that is driving the system towards an optimum. The improvement of the individuals can be considered as adaptation for the population as a whole. This explains the basic approach and terminology of several works on GAs, see eg. [Hol75, DeJ80].

In this paper we give a general description of genetic algorithms. We try to distinguish the most essential properties of GAs, and put them together into one general model. The result is an Abstract Genetic Algorithm (AGA). Interesting, although not unexpected, is that this universal algorithm unifies simulated annealing [Aar89] and traditional genetic algorithms. Strictly speaking, not only the classic GAs, but also any simulated annealing algorithm can be obtained as an instance of AGA.

The paper is organized as follows. In Chapter 1 we present the terminology and describe the Abstract Genetic Algorithm. In Chapter 2 we specify the Markov chain that can be associated with the search procedure of AGA. In Chapter 3 we establish convergence with probability 1 for this Markov chain, considering both the homogeneous and the inhomogeneous case. In Chapter 4 we interpret the general conditions of convergence and obtain conditions for the algorithm AGA. Finally we present some conclusions in Chapter 5.

1. AGA : AN ABSTRACT GENETIC ALGORITHM

For our discussion we restrict the application domain of genetic algorithms to that of combinatorial optimization problems. In general, such a problem is a pair (S,f), where S is a finite set, the search space or solution space, $f \in S \longrightarrow \mathbb{R}$ is the object function. The aim is to find a (global) minimum or optimum, that is an $s \in S$, such that $\forall\ t \in S \quad f(s) \leq f(t)$. Notice that the finiteness of S implies that f has at least one maximum over S.

The algorithm introduced below is a stochastic one, i.e. it is influenced by random variables. A deterministic instance can be easily obtained by keeping the random variables constant. Another remarkable feature of the algorithm is that it belongs to the so called local search methods [Pap82, Joh88]. The meaning of the word 'local' is given by the notion 'a neighbourhood of an $s \in S$'. We use this term in a way that does not coincide with the usual topological notion of neighbourhoods. Namely, here we only assume that every element s of S has exactly one non–trivial neighbourhood. Naturally, the whole space can be specified as the neighborhood of its elements. With this special instantiation we can relax locality and obtain a global (non–local) method.

A population is a subset of S. We model birth and death of individuals by a generation and a reduction function respectively. The generation function, however, is composed from two other functions: a selection function to choose the parents, and a production function to make the offspring. We surpass the biological analogies by not restricting the number of parents to the usual two.

Let $N = |S|$, $a \in \mathbb{N}$ such that $a \leq N$. We assume that the successive populations are of the same cardinality a, and that a is much much smaller than N. Let

$S_a = \{ x \subseteq S : a = |x| \}$ the set of 'well sized' populations, and let

$S_{a+} = \{ x \subseteq S : a \leq |x| \}$ the set of 'oversized' populations.

$P \subseteq \mathcal{P}(S)$ stands for the set of possible parents, i.e. let P contain all those sets that are capable of producing offspring. The elements of P will be called parent–sets.

To incorporate probabilities we introduce the sets A, B and C, and assume that the parameters α, β and γ are chosen from A, B and C by independent random drawings.

REMARK 1 Observe what it means to have a randomly parameterized function $f \in X \longrightarrow Y$. Strictly speaking it requires a set of functions $F \subseteq X \longrightarrow Y$, a probability space $(A,\mathcal{A},\mathbb{P})$ and a random variable $f \in (A,\mathcal{A},\mathbb{P}) \longrightarrow F$. Then $f(\alpha) \in X \longrightarrow Y$ for any $\alpha \in A$, hence f uniquely determines another function $g \in (A,\mathcal{A},\mathbb{P}) \times X \longrightarrow Y$ and vice versa. With a bit sloppy notation one mostly does not distinguish g and f but "extends $f \in X \longrightarrow Y$ by the parameter $\alpha \in A$" and denotes it as $f \in A \times X \longrightarrow Y$.

To specify our algorithm we need the following functions:

<u>A neighbourhood function</u> $N \in S \longrightarrow \mathcal{P}(S)$, such that for every $s \in S$:

$N(s) \neq \emptyset$ and $N(s) \neq \{s\}$.

$N(s)$ is the <u>neighbourhood</u> of $s \in S$. A $t \in S$ is a <u>neighbour of</u> $s \in S$, $(s \triangleright t)$ iff $t \in N(s)$. Notice that the relation $\triangleright \subseteq S \times S$ is not necessarily symmetric.

<u>A selection function</u> $f_s \in A \times S_a \longrightarrow \mathcal{P}(P)$, such that for every $\alpha \in A$, $x \in S_a$:
- $y \in f_s(\alpha,x) \implies y \subseteq x$,
- $y \in f_s(\alpha,x) \implies y \neq \emptyset$.

<u>A production function</u> $f_p \in B \times P \longrightarrow \mathcal{P}(S)$, such that for every $\beta \in B$ and $x \in P$:
- $f_p(\beta,x) \setminus x \neq \emptyset$,
- $f_p(\beta,x) \subseteq \bigcup_{s \in x} N(s)$.

<u>A reduction function</u> $f_r \in C \times S_{a+} \longrightarrow S_a$, such that for all $\gamma \in C$, $x \in S_{a+}$:

$f_r(\gamma,x) \subseteq x$.

<u>A stop function</u> $f_{st} \in D \times \mathcal{P}(S) \longrightarrow \{\text{true, false}\}$, where $d \in D$ is an external parameter. (As for d, one can think, for instance, of the number of iterations as external parameter to influence terminating.)

Notice that the locality of the search is due to the production function. The definition of f_p states that all the children are from the neighbourhood, that is only the neighbourhood is explored in search for improvements. In the meanwhile, do not forget that $N(s) = S$ is a correct definition, which frees us from being restricted to local search.

Let (S,f) be a combinatorial optimization problem, N be a neighbourhood function on S, and let f_s, f_p, f_r, f_{st} be a selction–, a production–, a reduction–, and a stop function, respectively.

The Abstract genetic Algorithm (AGA) contains the following basic steps.

0. Set the initial population $x \in S_a$.
1. Select parent–sets: $Q = f_s(\alpha,x)$
2. Produce the children of the selected parent–sets: $y = \bigcup_{q \in Q} f_p(\beta,q)$
3. Check the termination condition:
   If $f_{st}(d, x \cup y)$ **then output** a best element from $x \cup y$ **and stop,**
   else $\longrightarrow$ **step 4,**
4. Reduce the extended population: $x' = f_r(\gamma, x \cup y)$
5. **Let** $x = x'$ **and** $\longrightarrow$ **step 1.**

Notice that the selection function may choose more parent–sets, i.e. $|f_s(\alpha,x)| > 1$ can occur, and that the production operations in step 2 are independent. This is thus the point where parallel execution can be involved.

We claim that this model covers classical genetic algorithms and simulated annealing [Aar89]. To illustrate this let us consider two examples.

EXAMPLE 1 Take a classical deterministic GA: a finite binary space, with crossover of two parents plus mutation of single elements to create children, and a pure survival of the fittest mechanism. The appropriate, although partial, instantiation of the algorithm AGA is then the following.

$S = \{0,1\}^k$ $(k \in \mathbb{N})$, $a > 1$ arbitrary,

$P = \{\{s\} \mid s \in S\} \cup \{\{s,t\} \mid s,t \in S, s \neq t\}$,

$\forall s \in S : N(s) = S$,

$$f_p(\beta,x) = \begin{cases} cross(s,t) & \text{if } x = \{s,t\} \\ mut(s) & \text{if } x = \{s\} \end{cases} \qquad \text{(crossover and mutation are as usual),}$$

$f_r(\gamma,y) = \{s_1 \in y, \ldots, s_a \in y \mid \forall\, 1 \le i \le a\ \forall s \in y \setminus \{s_1, \ldots, s_a\} : f(s_i) \le f(s)\}$.

For the sake of convenience we leave out the further details. ▮

EXAMPLE 2 With the following instantiation we obtain a classic simulated annealing algorithm.

$S = \{0,1\}^k \ (k \in \mathbb{N}), \ \ a = 1,$

$P = \{\{s\} \mid s \in S\},$

$f_s(\alpha,\{s\}) = \{\{s\}\},$

$f_p(\beta,\{s\}) = \{t\}$ such that $t \in N(s), \ t \neq s,$

$$f_r(\gamma,\{s,t\}) = \begin{cases} \{t\} & \text{if } \exp\left[\dfrac{f(s) - f(t)}{c}\right] > \gamma \\ \{s\} & \text{otherwise} \end{cases},$$

where $0 < \gamma < 1$ is a random number, c is the control parameter. ∎

To give a better view on the abstraction here, we summarize the differences and the similarities between AGA and the classic GAs.

Similarities:

1) A finite search space is traversed in search for a minimal object function value.

2) The search is iterative, in each cycle of the iteration we have a set of candidates, a population.

3) New candidates in the search space are generated by constructing them from the old ones. Parents are chosen, offspring are produced, the population is extended.

4) There is an elimination mechanism to abort unfit elements, and thus increasing the fitness of the population.

Differences:

1) The search space in AGA is simply a set, the representation of the individuals is not restricted to binary coding.

2) Creation of children and mutation are unified by strongly generalizing the notion of 'parent', namely by dropping the tradition of having two of them (one for mutation).

3) The usual crossover mechanism for making offsprings is generalized to a production function.

4) Also the elimination mechanism is left very free in AGA by requiring the minimum from the reduction function.

The algorithm AGA is creating populations successively. This results in a sequence of populations which we shall call evolution. For a precise and easy definition of this notion we define two new functions.

The <u>generation function</u> $f_g \in A \times B \times S_a \rightarrow \mathcal{P}(S)$ is to create all the children 'in one go':

$$f_g(\alpha,\beta,x) = \bigcup_{y \in f_s(\alpha,x)} f_p(\beta,y).$$

The <u>transition function</u> $f_t \in A \times B \times C \times S_a \rightarrow S_a$ is to create the next population:

$$f_t(\alpha,\beta,\gamma,x) = f_r(\gamma, x \cup f_g(\alpha,\beta,x)).$$

Now let us take $\alpha_n \in A$, $\beta_n \in B$, $\gamma_n \in C$ $(n \in \mathbb{N})$ by independent random drawings and define the following sequence of populations:

$x_0 \in S_a$ be the initial population,

$x_{n+1} = f_t(\alpha_n,\beta_n,\gamma_n,x_n)$ for $n \geq 0$.

The set (sometimes referred to as a sequence) $\{x_n : n \in \mathbb{N}\}$ is the <u>evolution</u>.

Obviously one wants that the algorithm *converges*, that is it is approaching an optimum through the iterative (life) cycli. With genetic terminology the following could be expected: for any initial population an optimal population (i.e. a one containing an optimum) will occur in the evolution.

An interesting aspect is the need for ***divergence***. Besides convergence, we want to avoid that the course of the algorithm gets stuck at some local minimum. This requires some 'diversification', which is carried out by the random parameters of the algorithm.

2. THE MARKOV CHAIN BELONGING TO THE SEARCH

Let $(\Omega,\mathcal{A},\mathbb{P})$ be a probability space, and let us take a sequence of independent random variables $Z_n \in \Omega \longrightarrow A \times B \times C$ $(n \in \mathbb{N})$. Then $f_t \circ Z_n \in \Omega \times S_a \longrightarrow S_a$ is the transition function in the n–th iteration of AGA. Denoting the projections of $Z_n(\omega)$ as $\alpha_n = Z_n(\omega).1$, $\beta_n = Z_n(\omega).2$ and $\gamma_n = Z_n(\omega).3$ we get back the former notation.

The 'inside' of the transition mechanism is irrelevant for the followig investigations. Therefore we introduce the sequence of random variables $Y_n \in \Omega \longrightarrow (S_a \longrightarrow S_a)$, $n \in \mathbb{N}$. Our idea is that for each execution of AGA an $\omega \in \Omega$ is chosen by a random mechanism. Then for every $n \in \mathbb{N}$ $Y_n(\omega) \in S_a \longrightarrow S_a$ stands for the (already deterministic) n–th transition function. In its most general form the evolution is a sequence $X_n(\omega)$ $(n \in \mathbb{N})$:

$X_0(\omega) = x$, $x \in S_a$ is arbitrarily fixed, that is $\mathbb{P}[X_0(\omega) = x] = 1$,

$X_{n+1}(\omega) = Y_n(\omega)(X_n(\omega))$ for $n \geq 0$.

To provide an easier reading of the formulae we often leave out the symbol ω from the notation, i.e. we abbreviate $Y_n(\omega)$ by Y_n and $X_n(\omega)$ by X_n. In such cases Y_n stands for $Y_n \in S_a \longrightarrow S_a$, and $\mathbb{P}[X_n \in B]$ means $\mathbb{P}[\ \{\omega \in \Omega \mid X_n(\omega) \in B\}\ ]$, where $B \in \mathcal{P}(S_a)$.

Notice that we obtain different evolutions for different initial populations. Therefore we use a notation that indicates the dependence on the initial population:

$\{X_n : n \in \mathbb{N}\}_x$ denotes the evolution with $\mathbb{P}[X_0 = x] = 1$ and

$\mathbb{P}_x[\,..\,X_n\,..\,]$ stands for $\mathbb{P}[\,..\,X_n\,..\mid X_0 = x]$.

The assumption about the independence of the Z_n's naturally 'inherits' for the Y_n's, i.e. the following is assumed for every $n \in \mathbb{N}$ and $B_i \subsetneq S_a \longrightarrow S_a$ $(0 \leq i \leq n)$:

$$\mathbb{P}[Y_n \in B_n \wedge Y_{n-1} \in B_{n-1} \wedge \ldots \wedge Y_0 \in B_0] = \prod_{i=0}^{n} \mathbb{P}[Y_i \in B_i].$$

The next statement expresses a rewriting rule that will be applied in the following.

LEMMA 1 $\mathbb{P}_x[X_n = y \mid X_{n-1} = z] = \mathbb{P}_x[Y_{n-1}(z) = y] \quad \forall\, n \geq 1, \forall\, x,y,z \in S_a$.

Proof

It is trivial, we only remark that the independence of the Y_n's is necessary. ∎

The fact that the way of producing the offspring does not change from generation to generation can be formulated by assuming that the Y_n's have the same distribution.

Due to the definition $X_n(\omega)$ is an element of S_a for each $\omega \in \Omega$. Therefore we can consider X_n not only as an abbreviation of $X_n(\omega)$ but also as a random variable $X_n : \Omega \longrightarrow S_a$. On this basis the question whether the evolution $\{X_n : n \in \mathbb{N}\}_x$ is a Markov chain is a reasonable one.

LEMMA 2 $\{X_n : n \in \mathbb{N}\}_x$ is a Markov chain, and if the Y_n's have the same distribution then the chain is homogeneous.

Proof

Let $n > 0$, $x_i \in S_a$ $(i = 1, \ldots, n+1)$. Then by the independence and Lemma 1 we get

$\mathbb{P}[X_{n+1} = x_{n+1} \mid X_n = x_n \wedge \ldots \wedge X_0 = x] =$

$\mathbb{P}[Y_n(x_n) = x_{n+1} \mid Y_{n-1}(x_{n-1}) = x_n \wedge \ldots \wedge Y_0(x) = x_1] =$

$\mathbb{P}[Y_n(x_n) = x_{n+1}] =$

$\mathbb{P}[X_{n+1} = x_{n+1} \mid X_n = x_n]$, which proves the Markov property.

If the Y_n's have the same distribution then

$\mathbb{P}[X_m = y \mid X_{m-1} = z] = \mathbb{P}[X_n = y \mid X_{n-1} = z]$

is self–evident for any $y,z \in S_a$ and $m,n \in \mathbb{N}$. ∎

Notice that homogeneity does not hold for simulated annealing in general. The changing value of the control parameter leads to a changing distribution of f_r and hence the distribution of the transition function is not steady either. We return to this question in Chapter 4.

3. CONVERGENCE RESULTS

In this section we establish convergence in a broad sense, taking the Markov chain as a basis. First we want to express formally that the algorithm tends to an optimum. Observe that the search space is simply a set without any norm or distance measure. Therefore we can not expect convergence criteria saying that X_n $(n \longrightarrow \infty)$ is 'getting close' to an optimum. What remains is to require that X_n contains an optimum, or rather, that the chance of containing an optimum is growing to 1.

Let $S^* := \{s \in S \mid s \text{ is an optimum of } f\}$.

DEFINITION 1 An $s \in S$ is <u>accessible</u> by $\{X_n : n \in \mathbb{N}\}_x$ if $\mathbb{P}_x[\exists\, n \in \mathbb{N} : s \in X_n] > 0$.

DEFINITION 2 $\{X_n : n \in \mathbb{N}\}_x$ <u>surely reaches an optimum</u> if $\mathbb{P}_x[\exists\, n \in \mathbb{N} : X_n \cap S^* \neq \emptyset] = 1$.

LEMMA 3 "$\mathbb{P}_x$" and "$\exists$" commute, that is for every $x \in S_a$

$\exists\, n \in \mathbb{N}\ \mathbb{P}_x[s \in X_n] > 0 \iff \mathbb{P}_x[\exists\, n \in \mathbb{N} : s \in X_n] > 0$ for any $s \in S$.

Proof

Let us take an arbitrary $s \in S$ and introduce $A_n = \{\omega \in \Omega \mid s \in X_n(\omega)\}$ as an abbreviation.

$\Longrightarrow$

$\exists\, n \in \mathbb{N} : \mathbb{P}_x[A_n] > 0$ implies $\mathbb{P}_x[A_k] > 0$ for a certain $k \in \mathbb{N}$. Notice that $A_k \subseteq \{\omega \in \Omega \mid \exists\, n \in \mathbb{N} : s \in X_n(\omega)\}$ holds for any $k \in N$, hence we have

$0 < \mathbb{P}_x[A_k] \leq \mathbb{P}_x[\exists\, n \in \mathbb{N} : s \in X_n]$.

$\Longleftarrow$

Let $B_0 = A_0$, $B_{n+1} = A_{n+1} \setminus (A_n \cup \ldots \cup A_0)$ for $n > 0$. These B_i's are disjoint and $\{\omega \in \Omega \mid \exists\, n \in \mathbb{N} : s \in X_n(\omega)\} = \bigcup_{i \in \mathbb{N}} B_i$ holds obviously.

Then we have

$0 < \mathbb{P}_x[\exists\, n \in \mathbb{N} : s \in X_n] = \mathbb{P}_x[B_0] + \ldots + \mathbb{P}_x[B_i] + \ldots,$ which implies

$0 < \mathbb{P}_x[B_k]$ for a certain $k \in \mathbb{N}$. But then also

$0 < \mathbb{P}_x[s \in X_k]$ thus

$0 < \exists\, n \in \mathbb{N} : \mathbb{P}_x[s \in X_n]$. ∎

DEFINITION 3 The chain $\{X_n : n \in \mathbb{N}\}_x$ is <u>monotone</u> if
$\forall\, n \in \mathbb{N} : \min\{\, f(s) \mid s \in X_{n+1}\} \le \min\{\, f(s) \mid s \in X_n\}$.

REMARK 2 Observe that

$\forall\, s_{opt} \in S^* \; \forall\, n \in \mathbb{N} : s_{opt} \in X_n \Rightarrow s_{opt} \in X_{n+1}$ is not necessarily true, but
$\forall\, n \in \mathbb{N} : X_n \cap S^* \ne \emptyset \Rightarrow X_{n+1} \cap S^* \ne \emptyset$ holds for monotone chains.

LEMMA 4 If $\{X_n : n \in \mathbb{N}\}_x$ is monotone then "$\mathbb{P}_x$" and "$\lim_{n\to\infty}$" commute, consequently the following assertions are equivalent:

a) $\{X_n : n \in \mathbb{N}\}_x$ surely reaches an optimum,

b) $\mathbb{P}_x[\lim_{n\to\infty} X_n \cap S^* \ne \emptyset] = 1$,

c) $\lim_{n\to\infty} \mathbb{P}_x[X_n \cap S^* \ne \emptyset] = 1$.

Proof

Notice that if $A_n = \{\omega \in \Omega \mid X_n(\omega) \cap S^* \ne \emptyset\}$ and $\{X_n : n \in \mathbb{N}\}$ is monotone then the sets $A_1, \ldots, A_n, \ldots$ form a monotone sequence due to the above remark. The existence and the equality of $\lim_{n\to\infty} \mathbb{P}_x[A_n]$ and $\mathbb{P}_x[\lim_{n\to\infty} A_n]$ for monotone sequences is a known result of elementary measure theory. This implies the equivalence of (b) and (c).
The equivalence of (a) and (b) is straightforward if we consider that $\lim_{n\to\infty} A_n = \bigcup_{n\in\mathbb{N}} A_n$. ∎

The next theorem is the most general convergence result. The main idea underlying the proof is to have upper bounds on the probability of taking the wrong way, i.e. transitions that do not reach any optimum.

DEFINITION 4 For $x \in S_a$ the set of all populations that can occur in $\{X_n : n \in \mathbb{N}\}_x$ is $\{\vec{x}\} = \{y \in S_a \mid \exists\, n \in \mathbb{N} : \mathbb{P}[X_n = y \mid X_0 = x] > 0\}$. Furthermore, if $U \subseteq S_a$ then $\vec{U} = \bigcup_{x\in U} \{\vec{x}\}$.

THEOREM 1 Let $U \subsetneq S_a$ and let the following hold

a) $\{X_n : n \in \mathbb{N}\}_x$ is monotone for every $x \in U$,

b) $n_k \in \mathbb{N}$ and $\varepsilon_k \in (0,1]$ $(k \in \mathbb{N})$ are such that $n_k \to \infty$ $(k \to \infty)$ and $\prod_{k=0}^{\infty} \varepsilon_k = 0$, and

$\forall y \in \vec{U} : \mathbb{P}[X_{n_{k+1}} \cap S^* = \emptyset \mid X_{n_k} = y] \le \varepsilon_k$ holds for every $k \in \mathbb{N}$.

Then $\{X_n : n \in \mathbb{N}\}_x$ surely reaches an optimum for every $x \in U$.

Proof

Choose an arbitrary $x \in U$ such that $x \cap S^* = \emptyset$. Due to the monotony and Lemma 4 it is sufficient if we justify $\lim_{n\to\infty} \mathbb{P}_x[X_n \cap S^* \neq \emptyset] = 1$.

Let us define $p_0 = 1$ and $p_k = \mathbb{P}[X_{n_k} \cap S^* = \emptyset \mid X_0 = x]$ $(k > 0)$.

Then

$$p_{k+1} = \sum_{y \cap S^* = \emptyset} \mathbb{P}[X_{n_{k+1}} \cap S^* = \emptyset \mid X_{n_k} = y] \cdot \mathbb{P}[X_{n_k} = y \mid X_0 = x] \le$$

$$\le \sum_{y \cap S^* = \emptyset} \varepsilon_k \cdot \mathbb{P}[X_{n_k} = y \mid X_0 = x] = \varepsilon_k \cdot \sum_{y \cap S^* = \emptyset} \mathbb{P}[X_{n_k} = y \mid X_0 = x] = \varepsilon_k \cdot p_k.$$

This implies that

$$p_{k+1} \le \prod_{i=0}^{k} \varepsilon_k \cdot p_0.$$

Hence

$$\lim_{k\to\infty} \mathbb{P}[X_{n_k} \cap S^* = \emptyset \mid X_0 = x] = \lim_{n\to\infty} p_k \le \prod_{k=0}^{\infty} \varepsilon_k \cdot p_0 = 0.$$

Notice that the monotony of $\{X_n : n \in \mathbb{N}\}_x$ implies the monotony of the sequence $\mathbb{P}_x[X_n \cap S^* = \emptyset]$ $(n \in \mathbb{N})$, and then from $n_k \to \infty$ we have that

$\lim_{n\to\infty} \mathbb{P}_x[X_n \cap S^* = \emptyset] \le \lim_{k\to\infty} \mathbb{P}_x[X_{n_k} \cap S^* = \emptyset] = 0$, consequently

$\lim_{n\to\infty} \mathbb{P}_x[X_n \cap S^* \neq \emptyset] = 1$ holds. ▮

COROLLARY 1 Let the following conditions be satisfied:

a) $\{X_n : n \in \mathbb{N}\}_x$ is monotone for every $x \in S_a$,

b) $n_k \in \mathbb{N}$ and $\varepsilon_k \in (0,1]$ $(k \in \mathbb{N})$ are such that $n_k \to \infty$ $(k \to \infty)$ and $\prod_{k=0}^{\infty} \varepsilon_k = 0$, and

$\forall x \in S_a : \mathbb{P}[X_{n_{k+1}} \cap S^* = \emptyset \mid X_{n_k} = x] \le \varepsilon_k$ holds for every $k \in \mathbb{N}$.

Then for every $x \in S_a$ $\{X_n : n \in \mathbb{N}\}_x$ surely reaches an optimum. ∎

The following is our general convergence theorem for genetic algorithms.

THEOREM 2 Let $x \in S_a$ and the following conditions be satisfied:

a) $\{X_n : n \in \mathbb{N}\}_x$ is monotone, and

b) $\{X_n : n \in \mathbb{N}\}_x$ is homogeneous, and

c) for every $y \in \{\vec{x}\}$ there exists at least one accessible optimum.

Then $\{X_n : n \in \mathbb{N}\}_x$ surely reaches an optimum.

Proof

We take $U = \{x\}$ and construct a sequence $n_0, n_1, \ldots$, and a sequence $\varepsilon_0, \varepsilon_1, \ldots$ so that they satisfy condition (b) of Theorem 1.

Let $m_y = \min \{n \in \mathbb{N} \mid \mathbb{P}[X_n \cap S^* \neq \emptyset \mid X_0 = y] > 0\}$, the minimum number of steps required to find an optimum with positive chance when taking y as initial population.

According to (c), for every $y \in \{\vec{x}\}$

$\mathbb{P}_y[\exists\, n \in \mathbb{N} : s_{opt} \in X_n] > 0$ holds for a certain $s_{opt} \in S^*$. Then by Lemma 3 we have

$\exists\, n \in \mathbb{N} : \mathbb{P}_y[s_{opt} \in X_n] > 0$ which implies that for every $y \in \{\vec{x}\}$ m_y is finite.

Then $m = \max \{m_y \mid y \in \{\vec{x}\}\ \}$ is finite because S_a is finite, thus $\{\vec{x}\}$ is finite. Hence

$\forall\, y \in \{\vec{x}\} : \mathbb{P}[X_m \cap S^* \neq \emptyset \mid X_0 = y] > 0$ holds by the monotony (Remark 2), and thus

$$\forall\, y \in \{\vec{x}\} : \mathbb{P}[X_m \cap S^* = \emptyset \mid X_0 = y] < 1. \tag{i}$$

Introducing the abbreviation $p_y = \mathbb{P}[X_m \cap S^* = \emptyset \mid X_0 = y]$ we can define

$p = \max \{p_y \mid y \in \{\vec{x}\}\ \}$, where $p < 1$ since $\{\vec{x}\}$ is finite and $\forall\, y \in \{\vec{x}\} : p_y < 1$ by (i).

Now we have that

$\forall\, y \in \{\vec{x}\} : \mathbb{P}[X_m \cap S^* = \emptyset \mid X_0 = y] \le p$, and $p < 1$.

Let us define $n_k = m \cdot k$ and $\varepsilon_k = p$ $(k \in \mathbb{N})$ and observe that $n_k \to \infty$ $(k \to \infty)$ holds, and so does $\prod_{k=0}^{\infty} \varepsilon_k = 0$ since $p < 1$.

What remains is to show that

$$\forall\, y \in \{\vec{x}\} : \mathbb{P}[X_{n_{k+1}} \cap S^* = \emptyset \mid X_{n_k} = y] \leq \varepsilon_k \quad \text{for every } k \geq 0. \qquad \text{(ii)}$$

By the homogeneity we have that for every $y \in \{\vec{x}\}$

$$\mathbb{P}[X_{m\cdot(k+1)} \cap S^* = \emptyset \mid X_{m\cdot k} = y] = \mathbb{P}[X_m \cap S^* = \emptyset \mid X_0 = y] \leq p \quad \text{holds.}$$

This proves (ii), and hereby also the proof of the theorem is complete. ∎

4. APPLICATION OF THE CONVERGENCE RESULTS

In this section we apply the convergence results obtained for Markov chains. For simulated annealing Theorem 2 can be applied only if the control parameter is fixed, thus the corresponding Markov chain is homogeneous. For inhomogeneous cases we can make use of Theorem 1. Namely, $\mathbb{P}[X_n \cap S^* = \emptyset \mid X_0 = x]$ can be computed as a function of the control parameter c_k, then the conditions about n_k and ε_k impose conditions on c_k. Since many convergence theorems are known for SA [Aar89], but –to our best knowledge– none is for GA, we shall only investigate the second case in the present paper. Nevertheless, we remark that in order to have monotony in an SA algorithm one should slightly modify it. Without changing its characteristic acceptance mechanism, an SA can be extended to a = 2, such that the second element s' is 'the best seen so far'.

DEFINITION 5 The reduction function $f_r : C \times S_{a+} \longrightarrow S_a$ is <u>conservative</u> if it always keeps the best f value, that is $M_x \cap f_r(\gamma,x) \neq \emptyset$ for every $x \in S_{a+}$ and $\gamma \in C$, where $M_x = \{ s \in x \mid \forall t \in x : f(s) \leq f(t)\}$ contains the minima of x.

LEMMA 5 Let $\{X_n : n \in \mathbb{N}\}_x$ be the evolution created by AGA. If the reduction function is conservative then $\{X_n : n \in \mathbb{N}\}_x$ is monotone.

Proof

Notice that for any arbitrary $y \subseteq S$ and $\gamma \in C$

$\min\{ f(s) \mid s \in x \} \geq \min\{ f(s) \mid s \in x \cup y \} \geq \min\{ f(s) \mid s \in f_r(\gamma,x \cup y) \}$ due to the conservativity of f_r. Hence by

$X_{n+1} = f_r(\gamma_n, X_n \cup f_g(\alpha_n,\beta_n,X_n))$ we have that

$\min\{f(s) \mid s \in X_{n+1}\} \leq \min\{f(s) \mid s \in X_n\}$. ∎

Next we put up certain restrictions on the functions of AGA such that together they imply the conditions of Theorem 2.

i) Neighbourhood function

$\forall s \in S \; \forall t \in S : s \mapsto t$,

where $\mapsto$ stands for the transitive closure of relation 'neighbour–of ' $\triangleright \subseteq S \times S$.

ii) **Selection function**

$\{ \{s\} \mid s \in S \} \subseteq P$ and

$\forall x \in S_a \; \forall t \in x : \mathbb{P}[\{t\} \in f_s(\alpha,x)] > 0.$

iii) **Production function**

$\forall s \in S \; \forall t \in N(s) : \mathbb{P}[t \in f_p(\beta,\{s\})] > 0.$

iv) **Reduction function**

$\forall x \in S_{a+} \; \forall s \in x : \mathbb{P}[s \in f_r(\gamma,x)] > 0.$

Recall that $Z_n \in \Omega \to A \times B \times C$ $(n \in \mathbb{N})$ are the independent random variables that provide us the parameters α_n, β_n and γ_n for f_s, f_p and f_r in the n–th cycle of AGA. If we assume that these Z_n's have the same distribution for every $n \in \mathbb{N}$ then the Markov chain belonging to the search process of AGA is homogeneous by Lemma 2.

For an easier application of the conditions (ii) – (iv) we make another (technical) restriction on the sets A, B and C. In the sequel we assume that the following holds:

v) A, B and C are countable sets, with positive probability for all their members, i.e. $\forall \alpha \in A : \mathbb{P}[\{\omega \mid Z(\omega).1 = \alpha\}] > 0$, etc.

Observe that if (v) holds then the above conditions imply:

ii') $\forall x \in S_a \; \forall s \in x \; \exists \alpha \in A \; : \; \{s\} \in f_s(\alpha,x) \;\wedge\; \mathbb{P}[\{\omega \mid Z(\omega).1 = \alpha\}] > 0.$

iii') $\forall s \in S \; \forall t \in N(s) \; \exists \beta \in B \; : \; t \in f_p(\beta,\{s\}) \;\wedge\; \mathbb{P}[\{\omega \mid Z(\omega).2 = \beta\}] > 0.$

iv') $\forall x \in S_{a+} \; \forall s \in x \; \exists \gamma \in C \; : \; s \in f_r(\gamma,x) \;\wedge\; \mathbb{P}[\{\omega \mid Z(\omega).3 = \gamma\}] > 0.$

THEOREM 3 Let us assume that the drawings Z_n's have the same distribution. Let the conditions (i), (ii), (iii), (iv), (v) hold; furthermore let the reduction function be conservative. Then for any initial population AGA finds an optimum with probability 1.

Proof

The proof goes via Theorem 2, we show that its conditions (a), (b) and (c) hold for any $x \in S_a$. Let $x \in S_a$ be arbitrary and $\{X_n : n \in \mathbb{N}\}_x$ be the evolution created by AGA.

a) The reduction function is conservative and therefore $\{X_n : n \in \mathbb{N}\}_x$ is monotone by Lemma 5.

b) $\{X_n : n \in \mathbb{N}\}_x$ is a homogeneous Markov chain, due to the condition on the Z_n's.

c) We show even more than necessary, namely we prove

$\forall y \in S_a \; \forall s_{opt} \in S^* : \mathbb{P}_y[\exists\, n \in \mathbb{N} : s_{opt} \in X_n] > 0.$

Let $s_{opt} \in S^*$ and $s_0 \in y$ arbitrary. By (i) we have that $s_0 \mapsto s_{opt}$ holds. From the definition of $\mapsto$ it follows that there exists an $n \in \mathbb{N}$ and a sequence $s_1, \ldots, s_n$ from S, such that $s_{opt} = s_n$ and $s_0 \triangleright s_1 \wedge s_1 \triangleright s_2 \wedge \ldots \wedge s_{n-1} \triangleright s_n$.

Then we have

$$\mathbb{P}_y[s_{opt} \in X_n] \geq \mathbb{P}_y[s_1 \in X_1 \wedge \ldots \wedge s_{n-1} \in X_{n-1} \wedge s_n \in X_n] =$$
$$\mathbb{P}_y[s_1 \in f_t(Z_1,y) \wedge s_2 \in f_t(Z_2,f_t(Z_1,y)) \wedge \ldots \wedge s_n \in f_t(Z_n,\ldots f_t(Z_1,y)\ldots)] =$$
$$\sum_{z_1,\ldots,z_n} \mathbb{P}_y[s_1 \in f_t(z_1,y) \wedge \ldots \wedge s_n \in f_t(z_n,\ldots f_t(z_1,y)\ldots) \wedge Z_1 = z_1 \wedge \ldots \wedge Z_n = z_n] =$$
$$\sum_{(z_1,\ldots,z_n)\in H} \mathbb{P}_y[Z_1 = z_1 \wedge \ldots \wedge Z_n = z_n] = \sum_{(z_1,\ldots,z_n)\in H} \prod_{i=1}^{n} \mathbb{P}_y[Z_i = z_i] \qquad (*)$$

where

$H = \{(z_1,\ldots,z_n) \in (A \times B \times C)^n \mid s_1 \in f_t(z_1,y) \wedge \ldots \wedge s_n \in f_t(z_n,\ldots f_t(z_1,y)\ldots)\}$.

If $H \neq \emptyset$ then (*) is positive by (v) which proves $\mathbb{P}_y[s_{opt} \in X_n] > 0$. To show $H \neq \emptyset$ is thus sufficient to prove the theorem. Hence we need to construct a sequence $z_1, \ldots, z_n$ such that for any i $(0 < i \leq n)$ $s_i \in f_t(z_i, \ldots f_t(z_1,y)\ldots)$ holds.

Let $w \in S_a$, $s \in w$, $t \in N(s)$ arbitrary. Then

- there exists a $z^1 \in A$ such that $\{s\} \in f_s(z^1,w)$ by ii',
- there exists a $z^2 \in B$ such that $t \in f_p(z^2,\{s\})$ by iii', thus

  $t \in f_g(z^1,z^2,w)$ holds too,
- there exists a $z^3 \in C$ such that $v \in f_r(z^3, w \cup f_g(z^1,z^2,w))$ by iv', and hence

$t \in f_t(z,w)$ holds for $z = (z^1,z^2,z^3) \in (A \times B \times C)$.

Iterating this construction method for $w = y$, $s = s_0$ and $t = s_1$, then for $w = f_t(z_1,y)$, $s = s_1$, $t = s_2$ etc. we obtain the desired sequence $(z_1,\ldots,z_n) \in (A \times B \times C)^n$ such that $s_1 \in f_t(z_1,y) \wedge \ldots \wedge s_n \in f_t(z_n, \ldots f_t(z_1,y)\ldots)$ holds.

This verifies that $H \neq \emptyset$ and completes the proof of the theorem. ∎

The requirements on N can be relaxed at the cost of a further restriction of the reduction function.

weak i) $\exists\, s \in S\ \forall\, t \in S : s \mapsto t$, and let σ be such an element, i.e $\forall\, t \in S : \sigma \mapsto t$.

DEFINITION 6 Let $\sigma \in S$ be as in (weak i). Then the reduction function is <u>σ–preserving</u> if $\forall\, x \in S_{a+}\ \forall\, \gamma \in C : \sigma \in x \implies \sigma \in f_r(\gamma,x)$.

LEMMA 6 Let $x \in S_a$. If $\sigma \in x$ and the reduction function is σ–preserving, then $\forall\, y \in \{\vec{x}\} : \sigma \in y$.
The proof is trivial. ∎

THEOREM 4 Let us assume that the drawings Z_n's have the same distribution. Let the conditions (weak i), (ii), (iii), (iv), (v) hold; furthermore let the reduction function be conservative and σ–preserving. Then for any initial population which contains σ the algorithm AGA finds an optimum with probability 1.

Proof

Again, the proof is based on Theorem 2. Let $x \in S_a$ such that $\sigma \in x$, and $\{X_n : n \in \mathbb{N}\}_x$ be the evolution created by AGA.

The conditions (a) and (b) of Theorem 2 hold by the same reasoning as in Theorem 3.

c) We show that $\forall\, y \in \{\vec{x}\}\ \forall\, s_{opt} \in S^* : \mathbb{P}_y[\exists\, n \in \mathbb{N} : s_{opt} \in X_n] > 0$.

Let $y \in \{\vec{x}\}$ and $s_{opt} \in S^*$ be arbitrary. Due to $\sigma \in x$ and Lemma 6 we have that $\sigma \in y$. Hence for $s_0 \in y$ we can take σ, and then weak i implies that $s_0 \mapsto s_{opt}$.

The rest of the proof is identical to that of Theorem 3. ∎

5. CONCLUSIONS

In this paper we discussed a general theory of genetic algorithms. At the beginning we formulated two objectives:

1) to set up an abstract model of genetic algorithms, such that any special GA at hand is an instance of the model,
2) to achieve convergence results at a general level, such that these results are applicable at any special instance.

We claim that the first objective was reached by the Abstract Genetic Algorithm (AGA), that is AGA represents the set of all GAs. More precisely, we think that there is no algorithm which would be generally recognized as a genetic one, but is not an instance of AGA.

Furthermore, AGA generalizes simulated annealing, in the sense that the latter is an instance of AGA, where populations of size 1 (or 2 for the extended case) are used in combination with a special acceptance / reduction mechanism.

As to the second objective, a number of convergence theorems have been proved. Theorem 1 is general, it implies convergence with probability 1, if there is a bound on the probability of fruitless branches in the search. As a special case we have Theorem 2 for homogeneous Markov chains. Theorem 3 and Theorem 4 are further specializations of the above. They tell what kind of selection, production and reduction functions can guarantee that a GA finds an optimum with probability 1. These results can also be applied to simulated annealing. Since the changing value of the control parameter leads to inhomogeneity in the Markov chain, it is Theorem 1 that we can use. Further analysis is needed to disclose dependence on the control parameter to such an extent that we can deduce constraints for c_k which imply convergence.

LITERATURE

Aar89 Aarts, E.H.L., and Korst, J., Simulated Annealing and Boltzmann Machines, J. Wiley and Sons, 1989.

DeJ80 De Jong, K.A., Adaptive System Design: A Genetic Approach, IEEE Trans. on Sys.,Man & Cybern. SMC–10,9 566–574, Sept. 1980.

Hol75 Holland, J.H., Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor, 1975.

Gol89 Goldberg, D.E., Genetic Algorithms in Search, Optimization and Machine Learning, Addison–Wesley, Reading MA, 1989.

Gre85 Grefenstette, J.J., ed., Proceedings of the International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, Hillsdale, N.J., 1985.

Gre87 Grefenstette, J.J., ed., Proceedings of the Second International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, Hillsdale, N.J., 1987.

Gre89 Grefenstette, J.J., ed., Proceedings of the Third International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, Hillsdale, N.J., 1989.

Joh88 Johnson, D.S., Papadimitriou, C.H. and Yannakakis, M., How Easy Is Local Search?, Journal of Computer and System Sciences 37, 79–100 (1988).

Pap82 Papadimitriou, C.H. and Steiglitz, K., Combinatorial Optimization: Algorithms and Complexity, Prentice –Hall, Englewood Cliffs, N.J., 1982.

In this series appeared :

No.	Author(s)	Title
85/01	R.H. Mak	The formal specification and derivation of CMOS-circuits
85/02	W.M.C.J. van Overveld	On arithmetic operations with M-out-of-N-codes
85/03	W.J.M. Lemmens	Use of a computer for evaluation of flow films
85/04	T. Verhoeff H.M.L.J.Schols	Delay insensitive directed trace structures satisfy the foam rubber wrapper postulate
86/01	R. Koymans	Specifying message passing and real-time systems
86/02	G.A. Bussing K.M. van Hee M. Voorhoeve	ELISA, A language for formal specifications of information systems
86/03	Rob Hoogerwoord	Some reflections on the implementation of trace structures
86/04	G.J. Houben J. Paredaens K.M. van Hee	The partition of an information system in several parallel systems
86/05	Jan L.G. Dietz Kees M. van Hee	A framework for the conceptual modeling of discrete dynamic systems
86/06	Tom Verhoeff	Nondeterminism and divergence created by concealment in CSP
86/07	R. Gerth L. Shira	On proving communication closedness of distributed layers
86/08	R. Koymans R.K. Shyamasundar W.P. de Roever R. Gerth S. Arun Kumar	Compositional semantics for real-time distributed computing (Inf.&Control 1987)
86/09	C. Huizing R. Gerth W.P. de Roever	Full abstraction of a real-time denotational semantics for an OCCAM-like language
86/10	J. Hooman	A compositional proof theory for real-time distributed message passing
86/11	W.P. de Roever	Questions to Robin Milner - A responder's commentary (IFIP86)
86/12	A. Boucher R. Gerth	A timed failures model for extended communicating processes

	W.P. de Roever	
87/16	H.M.M. ten Eikelder J.C.F. Wilmont	Normal forms for a class of formulas
87/17	K.M. van Hee G.-J.Houben J.L.G. Dietz	Modelling of discrete dynamic systems framework and examples
87/18	C.W.A.M. van Overveld	An integer algorithm for rendering curved surfaces
87/19	A.J.Seebregts	Optimalisering van file allocatie in gedistribueerde database systemen
87/20	G.J. Houben J. Paredaens	The R^2 -Algebra: An extension of an algebra for nested relations
87/21	R. Gerth M. Codish Y. Lichtenstein E. Shapiro	Fully abstract denotational semantics for concurrent PROLOG
88/01	T. Verhoeff	A Parallel Program That Generates the Möbius Sequence
88/02	K.M. van Hee G.J. Houben L.J. Somers M. Voorhoeve	Executable Specification for Information Systems
88/03	T. Verhoeff	Settling a Question about Pythagorean Triples
88/04	G.J. Houben J.Paredaens D.Tahon	The Nested Relational Algebra: A Tool to Handle Structured Information
88/05	K.M. van Hee G.J. Houben L.J. Somers M. Voorhoeve	Executable Specifications for Information Systems
88/06	H.M.J.L. Schols	Notes on Delay-Insensitive Communication
88/07	C. Huizing R. Gerth W.P. de Roever	Modelling Statecharts behaviour in a fully abstract way
88/08	K.M. van Hee G.J. Houben L.J. Somers M. Voorhoeve	A Formal model for System Specification
88/09	A.T.M. Aerts K.M. van Hee	A Tutorial for Data Modelling
88/10	J.C. Ebergen	A Formal Approach to Designing Delay Insensitive Circuits
88/11	G.J. Houben J.Paredaens	A graphical interface formalism: specifying nested relational databases

86/13	R. Gerth W.P. de Roever	Proving monitors revisited: a first step towards verifying object oriented systems (Fund. Informatica IX-4)
86/14	R. Koymans	Specifying passing systems requires extending temporal logic
87/01	R. Gerth	On the existence of sound and complete axiomatizations of the monitor concept
87/02	Simon J. Klaver Chris F.M. Verberne	Federatieve Databases
87/03	G.J. Houben J.Paredaens	A formal approach to distributed information systems
87/04	T.Verhoeff	Delay-insensitive codes - An overview
87/05	R.Kuiper	Enforcing non-determinism via linear time temporal logic specification.
87/06	R.Koymans	Temporele logica specificatie van message passing en real-time systemen (in Dutch)
87/07	R.Koymans	Specifying message passing and real-time systems with real-time temporal logic
87/08	H.M.J.L. Schols	The maximum number of states after projection
87/09	J. Kalisvaart L.R.A. Kessener W.J.M. Lemmens M.L.P. van Lierop F.J. Peters H.M.M. van de Wetering	Language extensions to study structures for raster graphics
87/10	T.Verhoeff	Three families of maximally nondeterministic automata
87/11	P.Lemmens	Eldorado ins and outs Specifications of a data base management toolkit according to the functional model
87/12	K.M. van Hee and A.Lapinski	OR and AI approaches to decision support systems
87/13	J.C.S.P. van der Woude	Playing with patterns searching for strings
87/14	J. Hooman	A compositional proof system for an occam-like real-time language
87/15	C. Huizing R. Gerth	A compositional semantics for statecharts

88/12	A.E. Eiben	Abstract theory of planning
88/13	A. Bijlsma	A unified approach to sequences, bags, and trees
88/14	H.M.M. ten Eikelder R.H. Mak	Language theory of a lambda-calculus with recursive types
88/15	R. Bos C. Hemerik	An introduction to the category theoretic solution of recursive domain equations
88/16	C.Hemerik J.P.Katoen	Bottom-up tree acceptors
88/17	K.M. van Hee G.J. Houben L.J. Somers M. Voorhoeve	Executable specifications for discrete event systems
88/18	K.M. van Hee P.M.P. Rambags	Discrete event systems: concepts and basic results
88/19	D.K. Hammer K.M. van Hee	Fasering en documentatie in software engineering.
88/20	K.M. van Hee L. Somers M.Voorhoeve	EXSPECT, the functional part
89/1	E.Zs.Lepoeter-Molnar	Reconstruction of a 3-D surface from its normal vectors
89/2	R.H. Mak P.Struik	A systolic design for dynamic programming
89/3	H.M.M. Ten Eikelder C. Hemerik	Some category theoretical properties related to a model for a polymorphic lambda-calculus
89/4	J.Zwiers W.P. de Roever	Compositionality and modularity in process specification and design: A trace-state based approach
89/5	Wei Chen T.Verhoeff J.T.Udding	Networks of Communicating Processes and their (De-)Composition
89/6	T.Verhoeff	Characterizations of Delay-Insensitive Communication Protocols
89/7	P.Struik	A systematic design of a paralell program for Dirichlet convolution
89/8	E.H.L.Aarts A.E.Eiben K.M. van Hee	A general theory of genetic algorithms
89/9	K.M. van Hee P.M.P. Rambags	Discrete event systems: Dynamic versus static topology