united nations
educational, scientific
and cultural
organization

international atomic
energy agency

the
abdus salam
international centre for theoretical physics

SMR/1220-34

Workshop on

Nuclear Reaction Data and Nuclear Reactors: Physics, Design and Safety

13 March - 14 April 2000

Miramare - Trieste, Italy

Genetic Algorithms: Theory and Applications in the Safety Domain

M. Marseguerra
Polytechnic of Milan
Italy

GENETIC ALGORITHMS: THEORY AND APPLICATIONS IN THE SAFETY DOMAIN

DRAFT

M. MARSEGUERRA AND E. ZIO
Dept. of Nuclear Engineering, Polytechnic of Milan, Via Ponzio 34/3
20133 Milan – ITALY
E-mail: marzio.marseguerra@polimi.it

This work illustrates the fundamentals underlying optimization by genetic algorithms. All the steps of the procedure are sketched in details for both the traditional breeding algorithm as well as for more sophisticated breeding procedures. The necessity of affine transforming the fitness function, object of the optimization, is discussed in detail, together with the transformation itself. Procedures for the inducement of species and niches are also presented. The theoretical aspects of the work are corroborated by a demonstration of the potential of genetic algorithm optimization procedures on three different case studies. The first case study deals with the design of the pressure stages of a natural gas pipeline system; the second one treats a reliability allocation problem in system configuration design; the last case regards the selection of maintenance and repair strategies for the logistic management of a risky plant.

1. Introduction

As a first definition, it may be said that Genetic Algorithms (GA) are numerical search tools aiming at finding the global maximum (or minimum) of a given real *objective function* of one or more real variables, possibly subject to various linear or non linear constraints. Later, we shall see that the GAs are also suitable for other applications, such as finding the optimal control function of an industrial plant or improving the medical imaging.
The GAs owe their name to the fact that their functioning is inspired by the rules of the natural selection: correspondingly, the adopted language contains many terms borrowed from biology, which need to be suitably redefined to fit the algorithmic context. Thus, when we say that the GA operate on a set of (artificial) *chromosomes*, these must be understood as strings of numbers, generally sequences of binary digits 0 and 1. If the objective function has many arguments, each string is partitioned in as many substrings of assigned lengths, one for each argument and, correspondingly, we say that each chromosome is analogously partitioned in (artificial) *genes*. The genes constitute the so called *genotype* of the chromosome and the substrings, when decodified in real numbers called *control factors*, constitute its *phenotype*. When the objective function is evaluated in correspondence of the values of the control factors of a chromosome, its value is called the *fitness* of that chromosome. Thus each chromosome gives rise to a trial solution to the problem.
The GA search is performed by constructing a sequence of populations of chromosomes, the individuals of each population being the *children* of those of the previous population and the *parents* of those of the successive population. The initial population is generated by randomly sampling the bits of all the strings. At each step, the new population is then obtained by manipulating the strings of the old population in order to arrive at a new population hopefully characterized by an increased mean fitness. This sequence continues until a termination criterion is reached. As for the natural selection, the string manipulation consists in selecting and mating pairs of chromosomes in order to groom chromosomes of the next population. This is done by repeatedly performing on the strings the four fundamental operations of reproduction, crossover, replacement and mutation, all based on random sampling [1]. These operations will be detailed below.

[1] In the literature, the two operations of crossover and replacement are often unified and the resulting operation is called crossover, so that the fundamental operations are then three.

2. Definitions

Individuals and Population: An individual is a chromosome, constituted by $n \geq 1$ genes and a population is a collection of individuals. To code/decode the i-th gene in a control factor, that is in an argument of the objective function, the user:

- defines the range (a_i, b_i) of the corresponding argument in the objective function;
- assignes the resolution of that independent variable by dividing the range (a_i, b_i) in 2^{n_i} intervals. A number n_i of bits is then assigned to the substring representative of the gene and the relation between a real value $x \in (a_i, b_i)$ and its binary counterpart β is

$$x = a_i + \beta \frac{b_i - a_i}{2^{n_i}}$$

The values a_i, b_i, n_i are called the *phenotyping parameters* of the gene.
Figure 1 shows the constituents of a chromosome made up of three genes and the relation between the genotype and the external environment, i.e. the phenotype, constituted by three control factors, x_1, x_2, x_3, one for each gene. The passage from the genotype to the phenotype and viceversa is ruled by the phenotyping parameters of all genes, which perform the coding/decoding actions. Each individual is characterized by a fitness, defined as the value of the objective function calculated in correspondence of the control factors pertaining to that individual. Thus a population is a collection of points in the solution space, i.e. in the space of f.

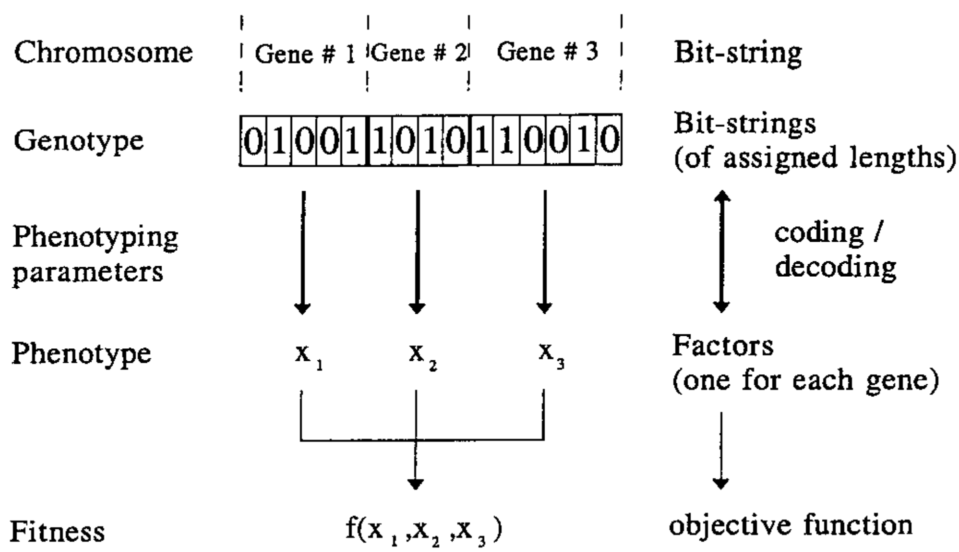

Figure 1. Components of an individual (a chromosome) and its fitness.

An important feature of a population is its genetic diversity: if the population is too small, the scarcity of genetic diversity may result in a population dominated by almost equal chromosomes and then, after decoding the genes and evaluating the objective function, in the quick convergence of the latter towards an optimum which may well be a local one. At the other extreme, in too large populations, the overabundance of genetic diversity can lead to clustering of individuals around

different local optima: then the mating of individuals belonging to different clusters can produce children (newborn strings) lacking the good genetic part of either of the parents. In addition, the manipulation of large populations may be excessively expensive in terms of computer time.
In most computer codes the population size is kept fixed at a value set by the user so as to suit the requirements of the model at hand. The individuals are left unordered, but an index is sorted according to their fitnesses. During the search, the fitnesses of the newborn individuals are computed and the fitness index is continuously updated.

3. Creation of the initial population

As said in the Introduction, the initial population is generated by random sampling the bits of all the strings. This procedure corresponds to uniformly sampling each control factor within its range. The chromosome creation, while quite simple in principle, presents some subtleties worth to mention: indeed it may happen that the admissible hypervolume of the control factors is only a small portion of that resulting from the cartesian product of the ranges of the single variables, so that one must try to reduce the search space by resorting to some additional condition. For example, in one of the application below described (The Natural Gas Pipeline System), the chromosome is made up of 150 bits so that the search hypervolume contains $\sim 10^{45}$ possible points. Obviously, it is highly improbable that an optimal solution can be found by a random search in such a huge space and indeed the search was unsuccessful until we realized that the value sampled for a gene drastically reduced the admissible range for the successive gene of the chromosome. By adopting this condition, which we called *conditioned sampling*, the search hypervolume was drastically reduced and the solution was readily found. The upshot is that the in the course of the population creation, a chromosome should be accepted only if suitable criteria are satisfied. This remark also applies to the *chromosome replacement*, below described.

4. The traditional breeding algorithm

The breeding algorithm is the way in which the $(n+1)$-th population is generated from the n-th previous one.

The first step of the breeding procedure is the generation of a temporary new population. Assume that the user has chosen a population of size N (generally an even number).The population reproduction is performed by resorting to the Standard Roulette Selection rule: to find the new population, the cumulative sum of the fitnesses of the individuals in the old population is computed and normalized to sum to unity. The new population is generated by random sampling, with replacement, individuals from this cumulative sum which then plays the role of a cumulative distribution function (cdf) of a discrete random variable (the position of an individual in the population). By so doing, on the average, the individuals in the new population are present in proportion to their relative fitness in the old population. Since individuals with relatively larger fitness have more chance to be sampled, most probably the mean fitness of the new population is larger.

The second step of the breeding procedure, i.e. the *crossover*, is performed as indicated in Fig.2: after having generated the new (temporary) population as above said, $N/2$ pairs of individuals, the parents, are sampled at random without replacement and irrespectively of their fitness, which has already been taken into account in the first step. In each pair, the corresponding genes are divided into two portions by inserting at random a separator in the same position in both genes (one--site crossover): finally, the first portions of the genes are exchanged. The two chromosomes so produced, the children, are thus a combination of the genetic features of their parents. A variation of this procedure consists in performing the crossover with an assigned probability p_c (generally rather high, say $p_c \geq 0.6$): a random number R is uniformly sampled in (0,1] and the crossover is performed only if $R < p_c$. Viceversa, if $R \geq p_c$, the two children are copies of the parents.

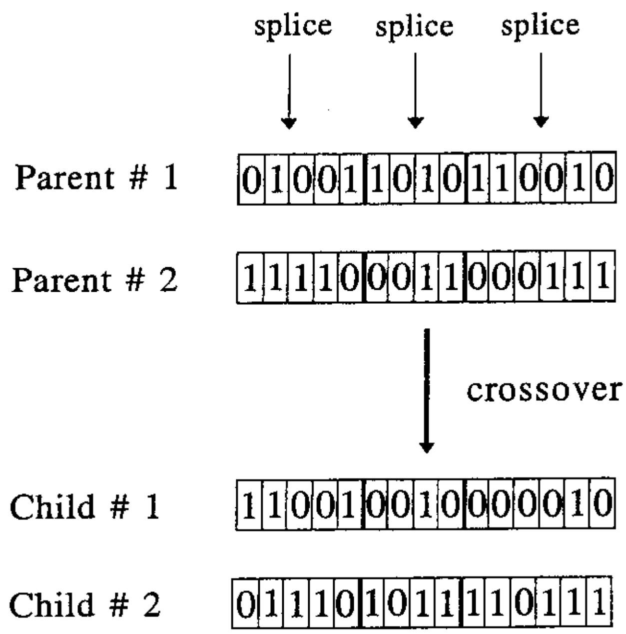

Figure 2. Example of crossover in a population with chromosomes constituted by three genes.

<u>The third step</u> of the breeding procedure, performed after each generation of a pair of children, concerns the replacement in the new population of two among the four involved individuals. The simplest recipe, again inspired by natural selection, just consists in the children replacing the parents: children live, parents die. In this case each individual breeds only once.

<u>The fourth and last step</u> of the breeding procedure eventually gives rise to the final $(n+1)$-th population by applying the mutation procedure to the (up to this time temporary) population obtained in the course of the preceding steps. The procedure concerns the mutation of some bits in the population, i.e. the change of some bits from their actual values to the opposite one ($0 \rightarrow 1$) and viceversa. The mutation is performed on the basis of an assigned mutation probability for a single bit (generally quite small, say 10^{-3}). The product of this probability by the total number of bits in the population gives the mean number μ of mutations. If $\mu < 1$ a single bit is mutated with probability μ. Those bits to be actually mutated are then located by randomly sampling their positions within the entire bit population.

<u>End of the search</u> The sequence of successive population generations is usually stopped according to one of the following criteria:

i. when the mean fitness of the individuals in the population increases above an assigned convergence value;
ii. when the median fitness of the individuals in the population increases above an assigned convergence value;

iii. when the fitness of the best individual in the population increases above an assigned convergence value. This criterion guarantees that at least one individual is good enough;
iv. when the fitness of the weakest individual in the population drops below an assigned convergence value. This criterion guarantees that the whole population is good enough;
v. when the assigned number of population generations is reached.

5. More sophisticated breeding procedures

Reproduction: Alternative procedures are:

i. Hybrid Roulette Selection: the main disadvantage of the Standard Roulette Selection procedure follows from the fact that the new individuals are actually sampled from a multinomial distribution, so that their fitnesses are fairly dispersed around the mean and the convergence towards the best solution can be delayed or even lost. The Hybrid Roulette Selection rule starts by normalizing the fitnesses to their sum and by sampling one of them as in the Standard Roulette Selection case. This normalized fitness is then multiplied by the population size (number of individuals in the population) and the integer part of the product yields the number of individuals, identical to that having fitness f in the old population, which are deterministically assigned to the new population (of course, this number may be zero). The remainder of the above product is then treated as the probability of adding a further identical individual to the new population. By so doing the permanence of good individuals, i.e. those with relatively higher fitness, is favoured along the population sequence.
ii. Random Selection and Mating: the two parents are randomly selected over the entire population, regardless of the fitnesses of the individuals.With respect to both Roulette Methods, on the average, this procedure is more disruptive of the genetic codes: in other words, the chromosomes of the two parents, suitably decodified, can give rise to points very far from each other in the control factor space and, correspondingly, the fitnesses of the newborn children can be largely far apart in the solution space.
iii. Fit-Fit Selection and Mating: the population is scanned by stepping through the fitness index and pairing each individual with the next fittest individual. On the average, this procedure is highly conservative of the genetic information and a (generally local) maximum of the objectivefunction is soon attained since weak individuals are rapidly eliminated;
iv. Fit-Weak Selection and Mating: as in the preceding case, the population is scanned by stepping through the fitness index, but this time each individual is paired with that individual located in the symmetric position of the fitness index, with respect to the mid of the fitness list. On the average, this procedure is highly disruptive of the genetic codes and, actually, it is seldom adopted.

In all the described procedures, after having selected the two parents and before proceeding to the selection of another couple, the two parents are crossed and the two individuals resulting from the adopted replacement procedure are immediately replaced in the population. Most important, before selecting the successive pair of parents, the fitness index is immediately updated: by so doing the sampling is performed on a dynamically varying population.

Crossover: An obvious generalization of the simple one-site crossing described above, is the multi-site crossing, consisting the interposition of more than one separator in the substrings representative of the homologous genes of the parents, followed by the exchange of pieces of the involved substrings. The simplest case is the two-site crossing: two separators are randomly positioned in the homologous substrings and the bits between the two points are interchanged. However it should be said that the multi-site crossing is rarely adopted and that the simple, one-site, crossover remains the most popular technique.

Replacement: Alternative procedures are:

i. Fittest individuals: out of the four individuals involved in the crossover procedure, the fittest two, parent or child, replace the parents.This procedure should not be used when weak individuals are discarded in the parent selection step, otherwise the weak individuals have a large chance to remain forever in the population.
ii. Weakest individuals: the children replace the two weakest individuals in the entire population, parents included, provided the children fitness is larger. This technique shortens the permanence of weak individuals in the successive populations and it is particularly efficient in large populations.
iii. Random replacement: the children replace two individuals randomly chosen in the entire population, parents included. By this technique, weak individuals have the same chance as the fit ones of being included in the new population: therefore the method is particularly efficient in small populations since it can give rise to a deep search in the space of the control factors.

6. Affine transform of the fitness

In general, the initial population contains a few chromosomes which, by chance, have a moderately good fitnesses embedded in a majority of second-rate individuals. Then, in a few generations, the selection rules cause that almost all these second-rate individuals disappear, being substituted by the moderately good ones, which are actually mediocre individuals: the genetic diversity is drastically reduced and we may have a premature convergence of the population fitness towards a local maximum. In the course of the successive generations, the crossover procedure generates mediocre individuals in place of mediocre individuals and the genetic selection may be seen as a random walk among mediocres [1]. To obviate to this unpleasant premature convergence to mediocrity, a pretreatment of the fitness function is welcome.
In the following we shall consider the affine transform of the fitnesses: instead of applying the selection rules to the fitness $f(x)$ we shall use its affine transform $f'(x)$, viz.,

$$f'(x) = a\,f(x) + b$$

Clearly, if b were set to zero, the transform would be useless, since the probability of selecting any population member would be the same before and after the transform. Otherwise, a positive b value favours the selection of lowly individuals and viceversa a negative b value favours the valuable ones.
Let f_m, f_{avg}, f_M and f'_m, f'_{avg}, f'_M be the minimum, average and maximum fitnesses before and after the transform, respectively. A first condition generally adopted for the determination of the coefficients a,b stems from the fact that the average offspring of an individual with average fitness is of one child. To preserve this relation after the fitness transform, the transformed and the original fitnesses should have equal average fitnesses, viz.,

$$f'_{avg} = f_{avg}$$

The second condition follows from the above consideration that the evolution of the chromosome population may be divided in two phases:

i. at the beginning, when the average population fitness is close to that of the worse individual, we would like to increase the genetic diversity by increasing the probability of selecting lowly individuals. This may be achieved by limiting the contribute of the best individual to the next generation. If C is the average offspring of the best individual, the condition is that C should be not much larger than one: for typical small populations ranging in the interval from 50 to 100 individuals, Goldberg [1]suggests a C value in the interval (1.2,2). This second condition yields the following expression for the transform of the maximum fitness

$$f'_M = C\, f_{avg}$$

The a,b values thereby resulting are

$$a = (C-1)\frac{f_{avg}}{f_M - f_{avg}}$$

$$b = (f_M - C\, f_{avg})\frac{f_{avg}}{f_M - f_{avg}}$$

The selection of the lowly individuals is then favoured provided b is positive, that is for $C < f_M / f_{avg}$. By so doing the few good individuals are scaled down and the majority of the mediocre members are scaled up;

ii. later, with the progression of the generations, the fitness of the best individual improves and in the meantime the average fitness gets close and close to the best one. Then, it may happen that the above expressions for a,b yield a negative value for the minimum transformed fitness, f'_m, which is unacceptable. In such case the second condition is substituted by the following

$$f'_m = 0$$

Then, the expressions for the a,b values are

$$a = \frac{f_{avg}}{f_{avg} - f_m}$$

$$b = -\frac{f_{avg}}{f_{avg} - f_m} f_m$$

In this case b turns out to be negative so that the affine transform favours the selection of the valuable individuals.

Note that in both the above cases i) and ii) the sum of the fitnesses is invariant with respect to the transform, viz.,

$$\sum_i f_i = \sum_i f_i'$$

Therefore, when the parents are selected in proportion to their fitnesses, we may compare the probabilities of selecting the i-th individual with reference to the original or to the transformed fitness just by comparing the numerical values of f_i and f'_i.

7. Inducement of Species and Niches

In the present context, a species is defined as a class of individuals sharing common features, a niche is a set of functions performed by the individuals of a species and the environment is the

collection of the external conditions, including the interactions with the other species. Examples of species, niche and environment are the humans, the elephant hunters and the ivory market, respectively.
In nature, we observe a large number of species which live - possibly competing - simultaneously. They develop or decline according to their degree of adaptability to the environmental changes and survive until the environment is propitious to them. In other words, each species has found a niche and survives until the niche is favourable.
The above considerations have been suggested by the observations of the various species living on the earth. In particular, for our species, the individuals are partitioned in groups according to the place they live in and the kinds of activities they carry out (niches). This division in groups, joined with a wise mixing of groups, has turned out to be of utmost advantage for the survival and the development of the species.
Let us now come back to the GAs and see how the grouping experience of the human species could be beneficial when suitably transferred to them. In particular, the problem here considered is that of seeing how the rules so far described for the selection of the successive GA populations might be altered. We shall describe two important techniques, only: the Isolation by Distance and the Spatial Mating. In both cases, the choice of the parents is performed within a limited number of individuals and therefore with a remarkable saving of computer time.

Isolation by Distance: The population of chromosomes is divided in sub-populations, i.e. in groups of individuals with scarce mutual interactions. Adopting the jargon of demography, we may say that the individuals of each sub-population live in isolated islands and that the various islands evolve separately, with different convergence rates, towards possibly different solutions. Such a rigid picture of totally isolated islands may be lowered by the introduction of *emigrants*. In general, the individuals living in an island mate with individuals of the same group, but there is also a small number of wanderlust individuals, the emigrants, which travel to other islands looking for the ideal mate.
The algorithmic implementation of the Isolation by Distance procedure is quite easy: the sub-populations in the various islands reproduce separately following the rules described in the preceding Sections for the whole population. Moreover, an emigration matrix E is assigned, whose generic element E_{ij} gives the emigration probability of an individual from island i to island j. Let us consider a particular island: at the beginning of each reproduction step, the numbers of immigrants from the other islands is sampled from the probability matrix E and the individuals selected as below specified are added to the natives. In the meantime an equal number of native elements (generally the weakest) are eliminated in order to keep constant the population size. The selection of the individuals emigrating from an island may be done in several different ways. A couple of them are:

i. at random from the current sub-population. This choice results in a great mixing of genes which prevents an excessive similarity among the individuals of an island;
ii. by sending copies of the fittest individual to the other islands. With respect to the previous one, this choice is more guided and therefore the genetic diversity among the different sub-populations is lower.

Spatial Mating: The population is again divided in sub-populations called *demes* [2] but these now overlap almost totally. Around each individual is created a deme constituted by an user assigned number of neighbours so that there is an equal number of individuals and demes. A first parent is selected throughout the entire population following one of the above described rules, but its partner is now selected within the corresponding deme only. There are several definition of the deme topology:

[2] A deme is one of the administrative divisions of the ancient Attica and of modern Greece.

i. the individuals are disposed on a planar square grid and the deme of an individual is constituted by its immediate neighbours (8,24,48,...);
ii. the individuals are disposed on a monodimensional wheel and the deme of an individual is constituted by an equal number of individuals symmetrically disposed around it (2,4,6,...);
iii. the deme is the entire population, again disposed on a monodimensional wheel: the second parent is selected by means of a random walk game with a given interval probability, generally an exponential distribution with a mean value of few steps: in practice the deme has a boundary of a few mean free paths.

8. Optimization of the design of a natural gas pipeline system

The operations of a natural-gas pipeline systems are characterized by inherent nonlinearities and numerous constraints. The optimization, actually minimization, of the total power required by the compressors in the transmission of natural gas in a serial pipeline problem was proposed by Wong and Larson and solved by the dynamic programming method [2]. Later, the same problem was solved by Goldberg [1] in the framework of the GAs. We shall now reconsider the problem by utilizing the GAs, with some variations with respect to the Goldberg's approach.
Let us consider a natural gas pipeline constituted by 10 compressors serially connected by 10 pipes, as schematically shown in Figure 3. The *i*-th compressor receives a given gas flow rate Q_{i-1} at the suction pressure PS_i and delivers a flow rate Q_i at the discharge pressure PD_i.

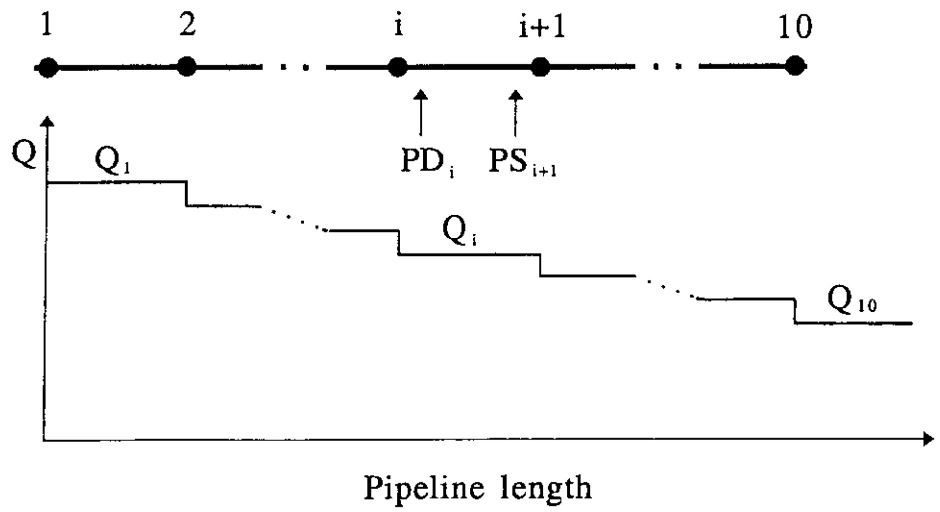

Figure 3. The natural gas pipeline system.

Mathematical model. Under steady state condition it is assumed that:

i. a known fraction r_{i-1} of Q_{i-1} is utilized as a fuel for operating the i-th compressor, so that

$$Q_i = (1 - r_{i-1})\, Q_{i-1} \qquad (i = 1,2,...,10)$$

where the r_i's are known constants;

ii. the equation relating the gas pressures at the beginning and at the end of the i-th pipe connecting the compressors i and $i+1$ is

$$PD_i^2 - PS_{i+1}^2 = K_i\, Q_i^2 \qquad (i = 1,2,...,10) \tag{1}$$

where the K_i's are known constants;

iii. the power required by the i-th compressor is

$$HP_i = Q_i \left[A_i \left(\frac{PD_i}{PS_i} \right)^{R_i} - B_i \right] , \quad (i = 1,2,...,10) \tag{2}$$

where K_i, A_i, B_i, R_i are given constants;

iv. The pressures are subject to the following constraints:

$$(P_{i-1})_{\min} \le PS_i \le (P_{i-1})_{\max} \tag{3}$$

$$(P_i)_{\min} \le PD_i \le (P_i)_{\max} \tag{4}$$

$$1 \le \frac{PD_i}{PS_i} \le (S_i)_{\max} \tag{5}$$

where $(S_i)_{max}$, $(P_i)_{min}$, $(P_i)_{max}$ are given constants

The problem may be stated as follows: given

- the suction pressure PS_1 at the entrance;
- the 10 flow rates Q_i,
- the 80 constants r_i, K_i, A_i, R_i, B_i, $(P_i)_{min}$, $(P_i)_{max}$, $(S_i)_{max}$ ($r_i = 5 \cdot 10^{-3}$. The other constants are given in the Table),

determine the 10 pressures PS_i, i=1,…,10 which satisfy the constraints iv) and minimize the total power required by all the compressors, viz.,

$$HP = \sum_{i=1}^{10} HP_i$$

i	K_i [psi^2 ·mmcdf $^{-2}$]	R_i	A_i [hp ·mmcdf $^{-1}$]	B_i [hp ·mmcdf $^{-1}$]	$P_{i,min}$ [psi]	$P_{i,max}$ [psi]	$S_{i,max}$
1	0.800	0.217	215.8	213.9	500	1000	1.6
2	0.922	0.217	215.8	213.9	500	1000	1.6
3	1.870	0.217	215.8	213.9	500	1000	1.5
4	0.894	0.217	215.8	213.9	500	1000	1.3
5	0.917	0.217	215.8	213.9	500	900	1.6
6	0.989	0.217	323.7	320.8	500	1000	1.6
7	0.964	0.217	215.8	213.9	500	900	1.75
8	1.030	0.217	215.8	213.9	500	1000	1.5
9	1.950	0.217	215.8	213.9	500	1000	1.6
10	1.040	0.217	215.8	213.9	500	1000	1.6

1 mmcdf (millions of cubic feet per day) = 0.3277 m^3/s;
1 psi (pounds per square inch) = 0.069 bar.

A first attempt to solve this problem consisted in trying to start with an initial population created by randomly sampling the bits of all the strings. Moreover, we adpted the Goldberg's control factors, i.e. the $U_i = PD^2_I + PS^2_i$. At the end of the GA processing, some pressures turned out to be complex, some others out of their allowed ranges. In a second attempt, we tried to accept the randomly created chromosomes provided they satisfy the constraints: also this attempt was quite unsuccessful since the time required for the creation of the initial population was exceedingly long. This was interpreted as an indication that, for some reason, the admissible ranges of some control factors are much smaller than the nominal ones, so thatthe probability that a randomly created gene could fall within it and therefore be accepted, is negligible. Finally, in a third attempt, assuming that a value sampled for a control factor would influence the ranges of the remaining ones, we adopted the *conditional sampling*: during the creation of the initial population, the first control factor of each chromosome is sampled at random within the specified range; this value is then inserted in the expressions for the problem constraints and the admissible range for the second control factor is computed. When the second control factor is sampled within this reduced range, it automatically satisfies its constraint. By applying this procedure to all the control factors of a chromosome, the creation of new chromosomes was quite easy and fast. In addition the control factors were changed from the Goldberg's $U_i = PD^2_I + PS^2_i$ to $U_i = PS_{i+1}$. More specifically, consider the i-th pipe-section between compressors i and $i+1$. The value PS_i is known either from the calculation of the preceding pipe-section or assigned as the initial condition for $i = 1$.
The conditioned range for PS_{i+1} is evaluated as follows:

i. From constraint (3) it follows that

$$(P_i)^2_{\min} \le PS^2_{i+1} \le (P_i)^2_{\max}$$

ii. Moreover, from equation (1) it follows that

$$PS^2_{i+1} = PD^2_i - K_i Q^2_i$$

so that PS^2_{i+1} must also belong to the range:

$$(PD_i)^2_{\min} - K_i Q^2_i \le PS^2_{i+1} \le (PD_i)^2_{\max} - K_i Q^2_i$$

where the range of PD_i^2 is established by constraints (4) and (5):

$$\max\left\{(P_i)^2_{\min}, PS_i^2\right\} \leq PD_i^2 \leq \min\left\{(P_i)^2_{\max}, (S_1)^2_{\max}\ PS_i^2\right\}$$

Finally, the range for PS_{i+1} is

$$\max\left\{(P_i)^2_{\min}, (PD_i)^2_{\min} - K_i Q_i^2\right\} \leq PS_{i+1}^2 \leq \min\left\{(P_i)^2_{\max}, (PD_i)^2_{\max} - K_i Q_i^2\right\}$$

Once the range for PS_{i+1} is established, a value PS_{i+1} is uniformly sampled. Then it is possible to proceed to the next pipe-section.
In this third attempt, the input values for the GA code were:

i. Population constituted by 50 chromosomes with 10 genes per chromosome and 15 bits per gene; the ranges of all the 10 control factors where from 500 to 1000 *psi*;
ii. parents selected by means of the fit-weak method;
iii. crossover performed with probability one, using a single separator;
iv. replacement performed as follows:
  - by selecting the fittest individuals with probability 500/555
  - by selecting the weakest individuals with probability 50/555
  - by the method children live, parents die with probability 4/555
  - by a random selection with probability 1/555
v. mutation performed with probability 0.003
vi. affine transform of the fitness with a coefficient C=1.6.

Figure 4 shows eight examples of chromosome initialisation by the conditional sampling method: the vertical lines and the stars represent the admissible ranges and the sampled PS_i values, respectively. Note the variation of the ranges of a specified control factor in the eight examples.

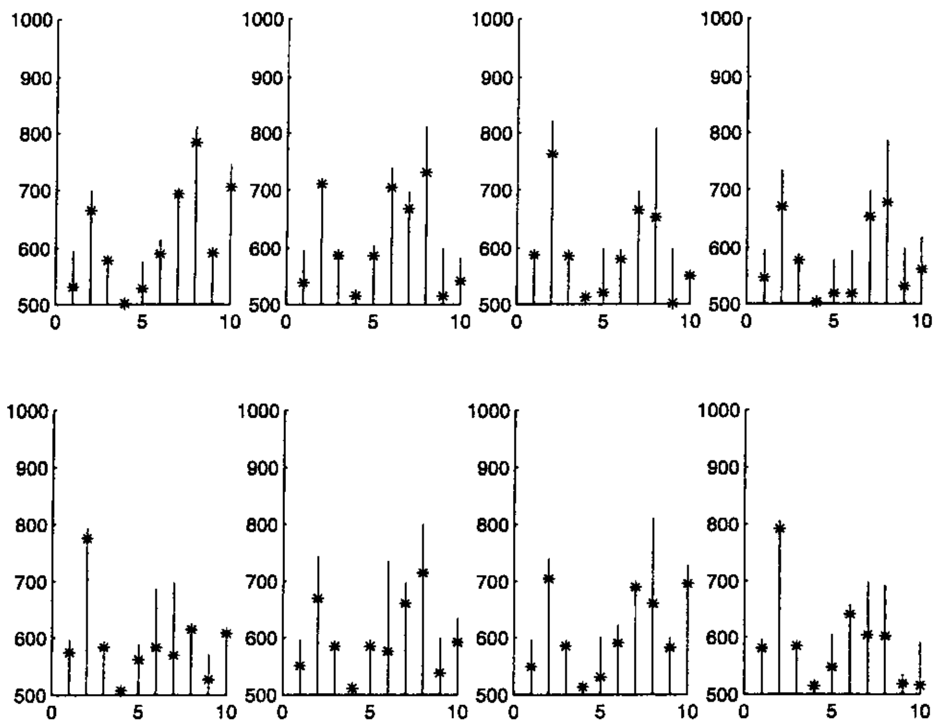

Figure 4. Eight examples of chromosome initialisation by the conditional sampling method.

The following figure 5 gives the mean ranges of the control factors during initialisation by the conditional sampling method. Note the smallness of the mean range of the fourth gene: this smallness explains why the attempt to create a population by random sampling the control factors over their nominal range was unsuccessful.

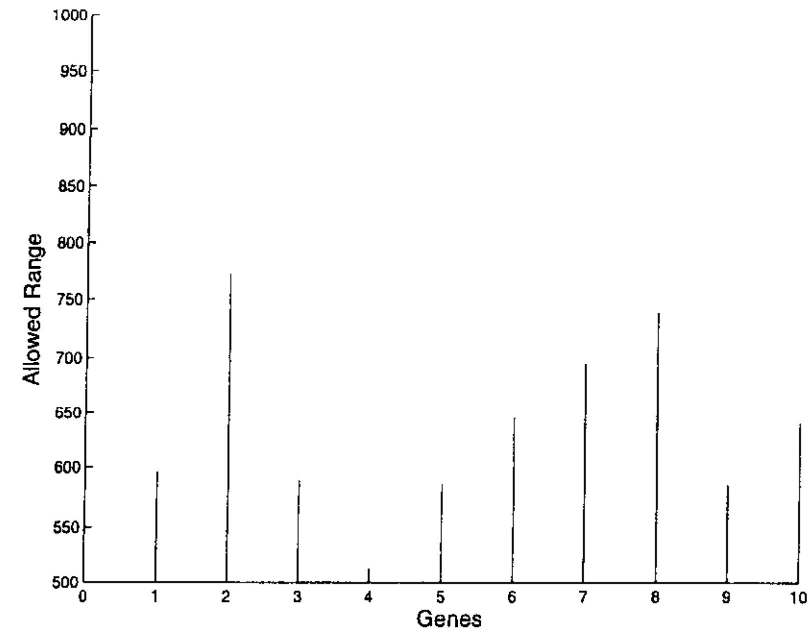

Figure 5. Mean ranges of the control factors during initialisation by the conditional sampling method.

Figure 6 represents the behaviour of the minimum and mean fitnesses along the generations: it appears that in ~70 generations they attain their minimum values.

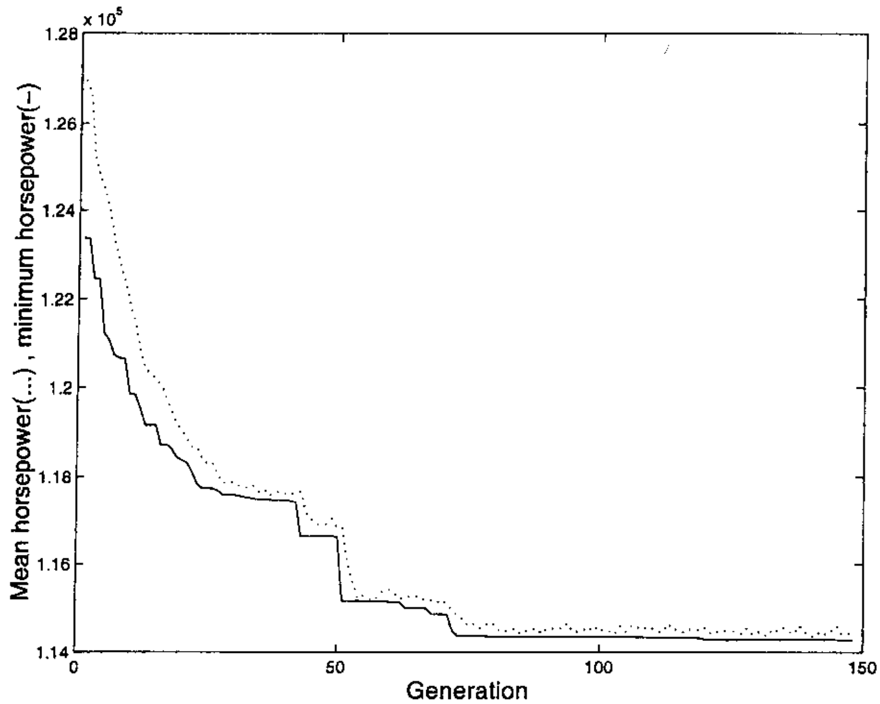

Figure 6. Minimum and mean fitnesses *vs.* generation number.

The results we obtained for the pipeline system are shown in figure 7, together with the Goldberg's ones: it appears that our results, obtained in ~30 s, are slightly better.

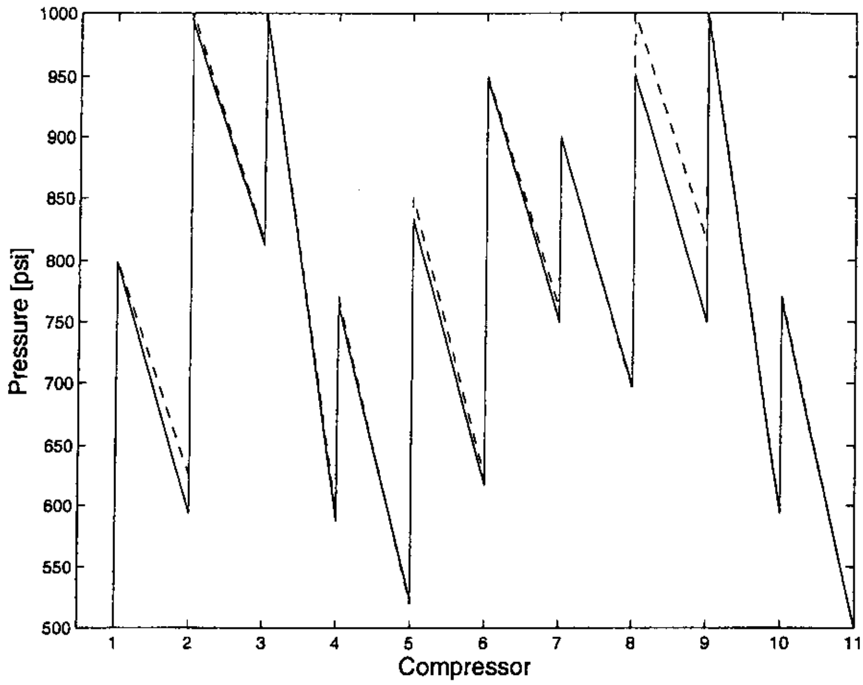

Figure 7. Comparison of this work results (solid line) with Goldberg's (dashed).

9. Optimization of the design of a risky plant

The operation and management of a plant requires proper accounting for the constraints coming from safety and reliability requirements as well as from budget and resource considerations. At the design stage, then, analyses are to be performed in order to guide the design choices in consideration of the many practical aspects which come into play and which typically generate a conflict between safety requirements and economic needs: this renders the design effort an optimization one, aiming at finding the best compromise solution.

In particular, the optimization problem here considered regards a choice among alternative system configurations made up of components which possibly differ for their failure and repair characteristics. The safety vs. economics conflict rises naturally as follows:

Choice of components: choosing the most reliable ones certainly allows the design to be on the safe side and guarantees a high system availability but it may be largely non-economic due to excessive component purchase costs; on the other hand, less reliable components provide for lower purchase costs but loose availability and may increase the risk of costly accidents.

Choice of redundancy configuration: choosing highly redundant configurations, with active or standby components, increases the system reliability and availability but also the system purchase costs (and perhaps even the repair costs, if the units composing the redundancy are of low reliability); obviously, for assigned component failure and repair characteristics, low redundancies are economic from the point of view of purchase costs but weaken the system reliability and availability, thus increasing the risk of significant accidents and the system stoppage time.

These very simple, but realistic, aspects of plant design immediately call for compromise choices which optimize plant operation in view of its safety and budget constraints.

Let us assume that technical considerations have suggested that the system at hand be made up of a series of N_n nodes, each one performing a given function. The task of the plant designer is now that

of selecting the configuration of each node which may be done in several ways, e.g. by choosing different series/parallel configurations with components of different failure/repair characteristics and therefore of different costs. In order to guide the selection, the designer defines an objective function which accounts for all the relevant aspects of plant operation. In our model we consider as objective function the net profit drawn from the plant during the mission time T_M. This profit is based on the following items: profit from plant operation; purchase and installation costs; repair costs; penalties during downtime, due to missed delivery of agreed service; damage costs, due to damages and consequences to the external environment encountered in case of an accident. Note that the last item allows one to automatically account for the safety aspects of system operation.
Any monetary value S encountered at time t is referred to the initial time t_0 by accounting for an interest rate i, as follows:

$$S_0 = \frac{S(t)}{(1+i)^{t-t_0}} \tag{6}$$

Our net profit objective function G (gain) can then be written as follows :

$$G = P - (C_A + C_R + C_D + C_{ACC}) \tag{7}$$

where :

$P = P_t \cdot \int_0^{T_M} \frac{A(t)dt}{(1+i)^t}$ is the plant profit in which P_t is the amount of money per unit time paid by the customer for the plant service (supposed constant in time) and A(t) is the plant availability at time t.

$C_A = \sum_{j=1}^{N_C} C_j$ is the acquisition and installation cost of all N_C components of the system. This capital cost is faced at the beginning, at $t_0=0$, through a bank loan at interest rate i. We imagine that the loan be re-paid at annual constant installments of nominal value $C_A \cdot \left(i + \frac{i}{(1+i)^{T_M} - 1} \right)$. If the plant is shut down before T_M (e.g. because of an accident) the residual installments are paid immediately so as to avoid paying the interests in the remaining time up to T_M.

$C_R = \sum_{j=1}^{N_C} C_{Rj} \cdot \int_0^{T_M} \frac{I_{Rj}(t)dt}{(1+i)^t}$ is the repair cost of all N_C components of the system with C_{Rj} being the cost per unit time of repair of component j (supposed constant in time and such that the total repair costs during the component life up to T_M equal a fixed percentage of its acquisition cost C_j) and $I_{Rj}(t)$ being a characteristic function equal to 1 if the component j is under repair at time t, 0 otherwise.

$C_D = C_U \cdot \int_0^{T_M} \frac{[1 - A(t)]dt}{(1+i)^t}$ is the amount of money to be paid to the customer because of missed delivery of agreed service during downtime, with C_U (constant in time) being the monetary penalty per unit of downtime.

$C_{ACC} = \sum_{k=1}^{N_{ACC}} I_{ACC,k} \cdot \frac{C_{ACC,k}}{(1+i)^{t_{ACC,k}}}$ is the amount of money to be paid for damages and consequences to the external environment in case of an accident; N_{ACC} is the number of different types of accidents that can occur to the plant, $C_{ACC,k}$ is the accident premium to be paid when an accident of type k occurs, $I_{ACC,k}$ is an indicator variable equal to 1 if an accident of type k has happened, 0 otherwise, and $t_{ACC,k}$ is the time of occurrence of the k-accident. We assume that after an accident the plant cannot be repaired and must be shut down.

We imagine that following a first qualitative enquiry, the designer ends up considering a pool of a reasonably small number of possible configurations for each node, a priori all offering similar system performances. Now the problem becomes that of choosing a system assembly by selecting one configuration for each node, aiming at maximizing the objective function. It is important to note that the various terms of the objective function (7) do not lend themselves to an analytical formulation so that it seems that the only feasible approach for their evaluation is the Monte Carlo method. In this approach, for each system configuration we simulate a specified number of histories and eventually estimate the mean values and standard deviations of the following quantities :

T_D = system downtime; T_R (i) = total time under repair of component i, i=1,...,N_C; P_{ACC} (j) = probability of accident j; A(t) = instantaneous system availability at time t.

By so doing, each history corresponds to a pseudo-experiment in which the system configuration at hand is followed in its evolution throughout the mission time. During the system life, the components undergo stochastic failures and repair transitions and, in correspondence, we record, in appropriately devised counters, the observed realizations of the following random variables: the number of times the system fails, the intervals of time during which the system remains in the down state; the intervals of time during which the individual components remain under repair. By performing several Monte Carlo histories of the behaviour of the specified system configuration, we obtain as many independent realizations of these random variables whose averages estimate the quantities of interest, T_D, T_R (i), i=1,...,N_C, P_{ACC} (j) = j = 1, 2, ..., N_{ACC}, A(t), $t \in [0, T_M]$. These quantities allow us to compute the various terms constituting the profit in eq. (7) pertaining to that system configuration. Obviously, the values of these quantities depend on the system configuration and components' reliability characteristics and must, therefore, be estimated anew for each system design examined.

Finally, one could think of saving computer resources by evaluating annual averages of the quantities of interest, instead of referring to the plant lifetime. This would be, indeed, correct in case of stationarity. In practice, realistic aspects such as aging, preventive maintenance, deteriorating repairs, stand-by operation modes, load connections, etc., render the process nonstationary.

A fundamental point in the above approach is that in order to arrive at choosing the best configuration we should, in principle, run one full Monte Carlo simulation for each of the potential system configurations. Here we are soon faced with a combinatorial explosion: even in the case of few nodes, each one with few possible configurations, we easily arrive at thousands of system configurations. It is, then, obvious that running a full Monte Carlo simulation for each configuration is an unconceivable task. This huge search space forces one to resort to a method capable of finding a near-optimal solution by spanning only a limited portion of the space. The method we have utilized is that of the genetic algorithms.

In this case, the parents selection phase is performed by resorting to the standard roulette rule, i.e. by selecting the parents in proportion to their values of fitness; the crossover is obtained by inserting at random a separator, or splice, in the homologous genes of the selected parents; the replacement is then performed by keeping the fittest two, and eliminating the remaining (so as to maintain the population number), among the two parents and two children; finally, the bit mutation is performed with probability 10^{-3}. With the assigned rules, which mimic natural selection, the successive generations tend to contain chromosomes with larger fitness values until a near optimal solution is attained.

As just said , in practical cases the design of a system involves a choice among a large number of potential configurations. This renders unfeasible running a full Monte Carlo simulation for each configuration. If the problem is tackled with the genetic algorithm, a Monte Carlo code should be run for each individual of the chromosome population throughout all the generations. Again, this is impractical. A possible solution to this problem follows from the consideration that in the genetic algorithm approach the best chromosomes appear a large number of times in the successive generations whereas the bad ones are readily eliminated [3]. Then, for each proposed chromosome, one can run a Monte Carlo code with a limited number of trials, e.g. 500, obtaining poorly significant statistical results. An archive of the simulated configurations, and corresponding Monte Carlo estimates, is maintained and updated by discarding the least fit configuration when the archive dimensions are exceeded. Then, whenever a chromosome is re-proposed, the results thereby obtained are accumulated with those stored in the archive as obtained in previous Monte Carlo runs pertaining to the same chromosome. The large number of times a good chromosome is proposed allows accumulating over and over the results of the few-histories runs, thus achieving at the end statistically significant results. We call this approach 'drop-by-drop' for its similarity to this way of filling a glass of water. At the same time, this way of proceeding avoids wasting time on 'bad' configurations which have small fitness values and are therefore simulated only a small number of times.

In order to exploit the genetic algorithm for the search of the optimal design configuration we need to code the available configurations into distinct bit-strings chromosomes. Recalling that our system is made up of N_n nodes, we identify the possible configurations of each node by an integer number so that a system configuration is identified by a sequence of integers, each one indicating a possible node configuration. A chromosome identifying a system configuration must then contain N_n integers, one for each node. For the coding, we choose to take a chromosome made up of a single gene containing all the indexes of the node configurations in a string of $k = \sum_{n=1}^{N_n} k_n$ bits, where k_n is the minimum number of bits needed to count the alternative configurations pertaining to the n-th node of the system. For example, if for a node we can choose among 6 alternative configurations we will need a 3-bit string which allows the coding of up to 8 configurations. In this case, the two chromosomes which code nonexistent configurations are discarded automatically by the genetic algorithm before entering the evolving population.

The choice of this coding strategy, as compared to a coding with one gene dedicated to each node, is such that the crossover generates children-chromosomes with nodes all equal to the parents except for the one in which the splice occurs. This avoids excessive dispersion of the genetic patrimony thus favoring convergence.

The present application, regards the optimization of the design of a plant with characteristics similar to those of a shale oil plant taken from literature [4]. The system consists of N_n = 5 nodes corresponding to the 5 process units indicated in Figure 8. For each node we have assumed that a design choice is required among several alternatives (Table 1): 7 for nodes A and E, 16 for node B and 14 for nodes C and D. The total number of possible system configurations is 153,664. Note that for nodes C and D although the 4-bits coding would allow 16 possible configurations, only 14 of these are physically significant, as generated by the combination of the various kinds of components in the three different lay-outs considered. The components are assumed to have exponential failure and repair behaviors in time.

The system configuration is coded into a chromosome with one 3+4+4+4+3 =18-bit gene. The huge dimension of the search space has made it necessary to consider a rather large chromosome population of 500 individuals.

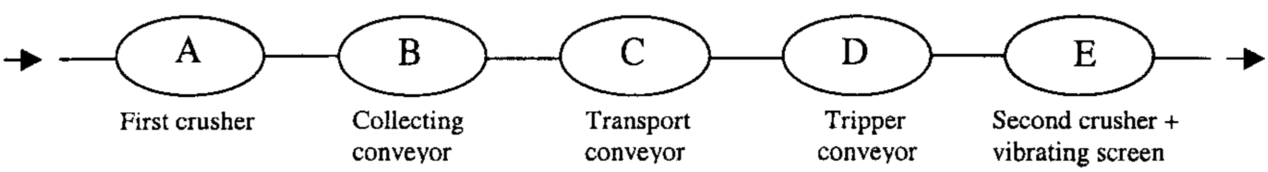

Figure 8. Sketch of the shale oil plant.

Table 1. Potential node configurations.

Node	Number of alternative configurations	Type of components	Operational logic
A	7	a	3-out-of-3 G 3-out-of-4 G 3-out-of-5 G 3-out-of-6 G 3-out-of-7 G 3-out-of-8 G 3-out-of-9 G
B	16	b1, b2, b3	2-out-of-2 G 2-out-of-3 G
C	14	c1, c2	1-out-of-1 G 1-out-of-1 G + 1 standby 1-out-of-1 G + 2 standby
D	14	d1, d2	1-out-of-1 G 1-out-of-1 G + 1 standby 1-out-of-1 G + 2 standby
E	7	e	3-out-of-3 G 3-out-of-4 G 3-out-of-5 G 3-out-of-6 G 3-out-of-7 G 3-out-of-8 G 3-out-of-9 G

All components are assumed to be subject to degrading repairs according to a modified Brown-Proschan model which postulates that a system is repaired to an "as good as before" condition (*minimal repair*) only with a certain probability p and is, otherwise, returned in a "deteriorated" condition (*deteriorating repair*) [5]. Thus, these two conditions obey a Bernoulli distribution. Inclusion of this model within the Monte Carlo simulation scheme is straightforward. When a repair action is completed, we sample a uniform random number r in [0,1): if $r<p$, then the repair is minimal and the failure properties of the component are returned to the conditions existing prior to failure; otherwise, repair is deteriorating and the component emerges with a failure rate increased by a given percentage π_λ of its value before the failure. The analyst-defined parameter π_λ then specifies the amount of deterioration induced by the failure-repair process. Repairs of deteriorated units are also assumed to become more difficult by reducing the repair rate by a given percentage π of its value before the failure. In our application we take $p = 0.3$, $\pi_\lambda = 1.5$ and $\pi_\mu = 1.3$ so as to enforce the effects of degradation. Moreover, all standby's are cold (no failure when in standby mode) and the failures of nodes A and E are assumed to be accidents with damaging consequences. Under these hypotheses, the total plant net profit over the mission time can no longer be evaluated analytically and Monte Carlo simulation becomes the only feasible approach. Tables 2 and 3 contain the component and system failure and cost data.

Table 2. Component data.

Component i	Failure rate λ_i [y^{-1}]	Repair rate μ_i [y^{-1}]	Purchase cost C_i [10^6 \$]	Repair cost C_{Ri} [10^6 \$$\cdot y^{-1}$]
a	$1.5\cdot10^{-3}$	$4.0\cdot10^{-2}$	3.0	0.55
b1	$2.0\cdot10^{-4}$	$8.0\cdot10^{-3}$	5.0	10.0
b2	$2.0\cdot10^{-3}$	$8.0\cdot10^{-2}$	3.0	6.2
b3	$2.0\cdot10^{-2}$	$8.0\cdot10^{-1}$	1.0	2.1
c1	$1.0\cdot10^{-4}$	$8.0\cdot10^{-3}$	10.0	41.0
c2	$1.0\cdot10^{-3}$	$8.0\cdot10^{-2}$	5.0	20.0
d1	$1.0\cdot10^{-4}$	$8.0\cdot10^{-3}$	7.0	28.0
d2	$1.0\cdot10^{-3}$	$8.0\cdot10^{-2}$	3.0	12.0
e	$1.7\cdot10^{-3}$	$4.0\cdot10^{-2}$	5.0	0.85

Table 3. System data.

Profit per unit time P_t	[10^6 \$$\cdot y^{-1}$]	20.0
Downtime penalty per unit time C_U	[10^6 \$$\cdot y^{-1}$]	200.0
Accident 1 (node A) reimbursement cost $C_{ACC,1}$	[10^6 \$]	70.0
Accident 2 (node E) reimbursement cost $C_{ACC,2}$	[10^6 \$]	50.0
Interest rate i		3%
Mission time T_M	[y]	50

The data are adapted from the original data to fit the model. Due to the huge number of potential solutions in the search space, we have found it convenient to run the procedure in two successive steps. The first step has consisted in a usual genetic algorithm search spanning the whole space with a limited number of generations (25) sufficient to achieve convergence to a near optimal solution. An analysis of the chromosome archive at convergence has allowed us to conclude that all the potential solutions with highest fitness values where characterized by the same 3-out-of-5 G redundancy configurations at the risky nodes A and E whereas the difference in net profit is given by variable combinations of the configurations at the remaining nodes. We have then proceeded to a fine tuning of the system optimal design configuration by running another search procedure focussed only on the $16\cdot14\cdot14=3136$ configurations of systems having nodes B, C and D variable and A and E fixed at their optimal configuration (i.e., the 3-out-of-5 G). At convergence, this step has led to the optimal configuration of Figure 9. We can now try to give an a posteriori interpretation of this result. Nodes A and E are the risky ones so that low-redundancy configurations are not acceptable due to the high cost of accident consequences. On the other hand, high-

redundancy configurations imply very large components purchase costs so that the optimal choice falls on an averagely-redundant configuration, the 3-out-of-5 G, which is capable of guaranteeing a low probability of risky failures which, in this case, turned out to be equal to $1.15 \cdot 10^{-3}$ and $1.16 \cdot 10^{-3}$ for nodes A and E, respectively (note that the failure of node A leads to worse consequences, as in Table 3). The types of components which can be used to make up node B, C and D are characterized by a wide range of purchase costs. Economics would then drive the choice towards the least expensive, but this has to be compatible with the need of minimizing system downtime which bears a strong penalization. These observations explain the choice, for node B, of the 2-out-of-3 G configuration made of a combination of two components of the most expensive and most reliable kind (b_1) and one of the least expensive and least reliable (b_3). For nodes C and D a single component among the cheapest ones is found to suffice to guarantee a high level of reliability, given that the associated mean time to failure is 20 times the mission time T_M. The chosen system configuration has an unreliability of $9.69 \cdot 10^{-3}$ and an average unavailability of $1.27 \cdot 10^{-3}$.
A glance at the chromosome archive allows us to see that the selected configuration is better than the second best of an amount of net profit ten times larger than its standard deviation, thus re-assuring on the robustness of the result. For further checking, we have run, for the three best configurations, a standard Monte Carlo simulation with 10^6 trials: the results are reported in Table 4 and show that the optimal solution is indeed better than the other two, beyond statistical dispersion.

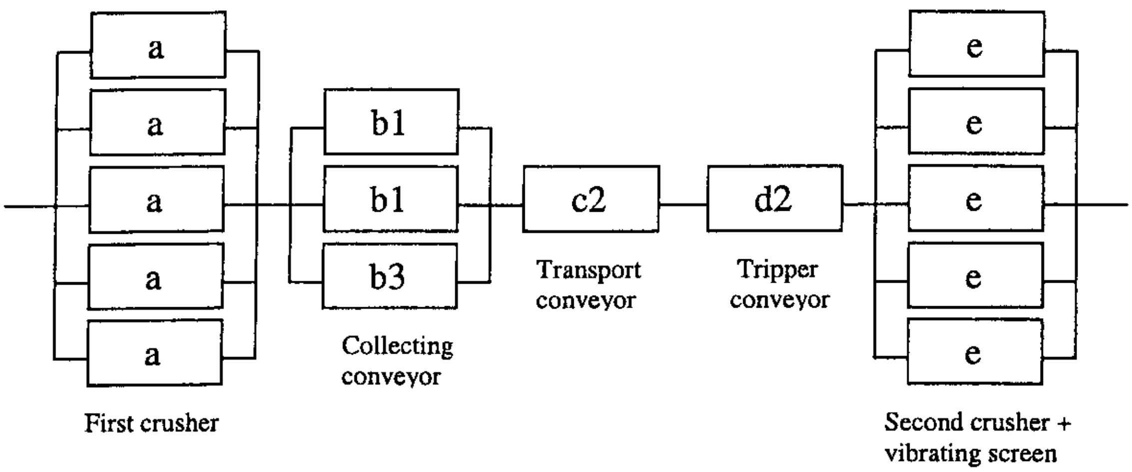

Figure 9. Sketch of the optimal configuration for the shale oil plant.

Table 4. Monte Carlo results with 10^6 trials for the three best system configurations.

Configuration index in decreasing order of optimality	**Total net profit at T_M [10^6 \$]**
1	471.57 ± 0.08
2	470.20 ± 0.05
3	469.39 ± 0.07

10. Optimization of the maintenance and repair strategies of a risky plant

Maintenance is nowadays recognized as a fundamental aspect to be accounted for in risk/economic analyses. An efficient maintenance strategy can ensure both safe operation and economic gain; on the contrary unsatisfactory performances and substantial waste of resources may result from an inefficient management of maintenance activities.
In this work we assume that components are assumed to age in time according to a linear model [6] for the failure rate $\lambda(t)$,

$$\lambda(t) = \lambda_0 + a \cdot t \tag{8}$$

where λ_0 is a constant term and a is the aging rate. Then, the failure times behave according to the following cumulative distribution function (cdf) and probability density function (pdf):

$$F(t) = 1 - e^{-\int_0^t \lambda(u)du} = 1 - e^{-\int_0^t (\lambda_0 + a\cdot u)du} = 1 - e^{-(\lambda_0 \cdot t + \frac{1}{2} a \cdot t^2)} \tag{9}$$

$$f(t) = \frac{dF(t)}{dt} = (\lambda_0 + a \cdot t) \cdot e^{-(\lambda_0 \cdot t + \frac{1}{2} a \cdot t^2)} \tag{10}$$

The corresponding mean time to failure (MTTF) is :

$$\overline{t_f} = \int_0^\infty t \cdot f(t)dt = \int_0^\infty (\lambda_0 t + at^2) e^{-(\lambda_0 t + \frac{1}{2}at^2)} dt = \frac{1}{2}\sqrt{\frac{2\pi}{a}} e^{\frac{\lambda_0^2}{2a}} \left[1 - \Phi\left(\frac{\lambda_0}{\sqrt{2a}}\right)\right] \tag{11}$$

where $\Phi(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-u^2} du$ is the error function.

The effects of aging are mitigated through preventive maintenance actions, performed with period τ, which rejuvenate the component. The period τ is chosen sufficiently small that the failure rate $\lambda(t)$ increases only slightly. In order to simplify the calculations, we approximate the matter by substituting the cdf (9) with the simpler exponential distribution, whose parameter λ^* is determined by imposing that the two cdfs give the same failure probability within $[0, \tau]$. It follows that the failure rate of the exponential distribution must have the effective value (constant throughout the maintenance period):

$$\lambda^*(\tau) = \lambda_0 + \frac{1}{2} a\tau \tag{12}$$

which is the average of the $\lambda(t)$ function over the period τ. Note that this effective failure rate of a component, λ^*, is strictly linked to the maintenance period τ and this will have a fundamental impact on the optimization of maintenance with respect to this parameter, as we shall see below.
Obviously, the exponential approximation is good for small values of the aging rate, i.e. for slowly varying failure rates, whereas for large values of the aging rate the discrepancy between the two cdfs becomes significant as they coincide only at $t=\tau$. In this latter case, in the limit of zero repair times, the probability of a failure within the period $[0, \tau]$ is the same by construction but the failure occurs at very different times: the exponential distribution somewhat favors early failures whereas the distribution with linear aging shifts the failures to later times, closer to the end of the period τ.
As for what concerns the repair process, we shall adopt the usual assumption of constant repair rate, μ. Although we realize that the repair process is all but markovian, this assumption, which can be easily removed in a Monte Carlo approach, allows us to compare, when possible, the analytical results with those obtained by the Monte Carlo simulation. Moreover, since the asymptotic system availability of a repairable component depends on the average repair time, the approximation of constant repair rate is significant only during the transient evolution [7].

In practice, different components may have different failure modes which require different types of repair interventions, e.g. electronic, mechanical, hydraulic etc. We assume that such repairs are performed by specialized repair teams, each of which can operate only on one component at the time. Moreover, since it is not likely to have many components of a given kind simultaneously under repair, it is economically convenient to keep only a limited number of specialized teams on paycheck. This, however, entails the introduction of a priority rank for the components belonging to the same class of repairs so as to manage the repair intervention if we were to have a situation in which there are more components in need of a repair than available teams specialized on that kind of repair. Such priority rank is intended to represent the importance of the component for the good operation of the system and should therefore be somewhat related to the classical importance measures [8,9] already developed for risk analysis. Repair interventions are then performed first on components with high priority ranks. In particular, if at one point in time all teams for a given type of repair were out working on failed components and another component with higher priority were to fail and to require the same kind of repair, then the team working on a lower ranked component would interrupt the work and switch to the newly failed, higher ranked component to take care of its failure. The number of repair teams for each kind of repair intervention will be subject to optimization by the genetic algorithms and Monte Carlo procedure.

Finally, we consider the possibility of deteriorating repairs in the sense that as a result of a repair action the component does not necessarily return to an "as good as new" condition but may come out more fragile and prone to future failures. To account for imperfect, deteriorating repairs, we adopt a modified Brown-Proschan model of stochastic repairs which postulates that a system is repaired to an "as good as before" condition (*minimal repair*) only with a certain probability p and is, otherwise, returned in a "deteriorated" condition (*deteriorating repair*) [5]. Thus, these two conditions obey a Bernoulli distribution. In case of deteriorating repair, the component emerges with a failure rate increased and a repair rate reduced by given percentages, π_λ and π_μ respectively, with respect to the values before the failure. Hence the deterioration has a combined effect of increasing the failure rate of the component and of somewhat 'complicating' its repair which thus requires, on average, more time, as modeled by the reduced repair rate. The analyst-defined parameters π_λ and π_μ specify the amount of deterioration induced by the failure-repair process.

The maintenance model here proposed is a modification of a previously adopted model [10] and it is based on the following assumptions: i) the maintenance of a component is performed with variable period τ and the periodicity varies with the component's deterioration due to imperfect repair (Figure 10); ii) the period τ between successive maintenances is made up by the time intervals during which the component is operating in its active state. In other words, when the system is inactive, e.g. in standby mode, the elapsed time does not contribute to τ; iii) maintenance actions are such to restore the conditions existing at the beginning of the previous maintenance period; iv) maintenance actions are instantaneous and a sufficient number of maintenance teams is available at all times for all kinds of maintenance required.

In realistic situations, the maintenance activities become more and more frequent as the component ages and its reliability characteristics deteriorate. In our model, deterioration of a component occurs due to the imperfect repairs, in accordance to the Brown-Proschan model previously introduced, and we allow for an adaptive schedule of maintenance intervention according to which the ratio between the maintenance period τ and the mean time $1/\lambda$ between successive failures is kept constant. Thus, after a minimal repair $\lambda_{new} = \lambda_{old}$ and, thus, $\tau_{new} = \tau_{old}$; on the contrary, after a deteriorating repair, $\lambda_{new} = (1+\pi_\lambda)\lambda_{old}$ and, then, $\tau_{new} = \tau_{old}\ \lambda_{old} / \lambda_{new} = \tau_{old} / (1+\pi_\lambda)$. Figure 11 shows an example with π_λ=1.

Finally, it seems reasonable to assume that for each component the period between maintenances is a fraction $1/\alpha_\tau$ of its MTTF, i.e.:

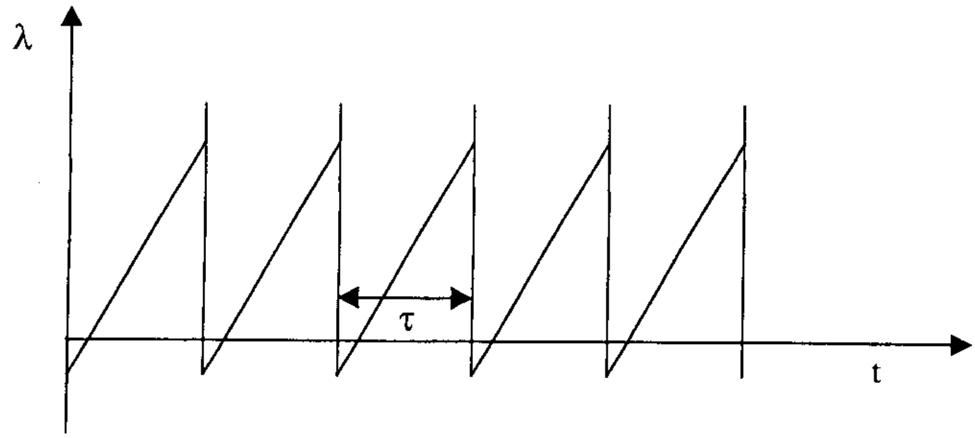

Figure 10. Linearly aging failure rate and mitigating effect of maintenance.

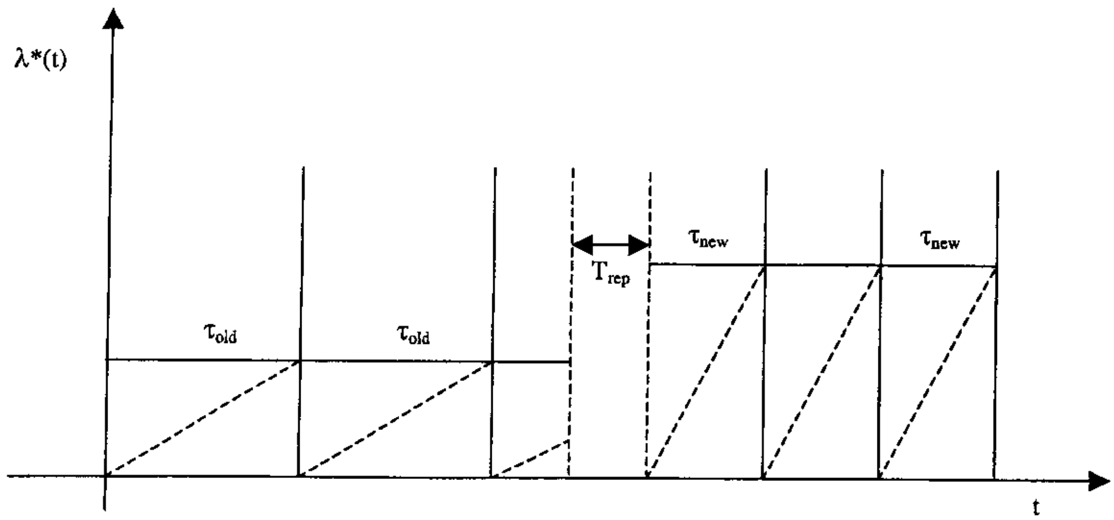

Figure 11. Adaptative maintenance period for a linearly aging component. After component failure and repair (lasting a time T_{rep}) the component ages according to the Brown-Proschan model and the maintenance period is shortened from τ_{old} to τ_{new}.

$$\tau = \frac{1}{\alpha_\tau} \cdot \bar{t_f} = \frac{1}{\alpha_\tau} \frac{1}{\lambda^*(\tau)} = \frac{1}{\alpha_\tau} \cdot \frac{1}{\lambda_0 + a \cdot \frac{\tau}{2}} \tag{13}$$

We then obtain the explicit expression for the maintenance period:

$$\tau = \frac{\lambda_0}{a} \left[\sqrt{1 + \frac{1}{\alpha_\tau \frac{\lambda_0^2}{2a}}} - 1 \right] \tag{14}$$

By so doing, the maintenance period τ of each component is linked to the values of (λ_0,a) characterizing its failure behavior and the optimization of the maintenance periods is limited to the best value of the parameter α_τ, valid for all components.

For the evaluation of the various maintenance and repair strategies we consider the net gain G drawn from the plant operation during the mission time T_M. This gain, which will serve as our objective function in the optimization process, is the same as that of the previous case, with the additional items of maintenance costs and repair teams salaries. The net gain objective function G can then be written as follows :

$$G = P - (C_A + C_M + C_R + C_{team} + C_D + C_{ACC}) \tag{15}$$

where the additional costs are:

$C_M = \sum_{j=1}^{N_C} n_{Mj} C_{Mj}$ is the total cost of maintenance activities on all N_C components of the system, with C_{Mj} being the cost per single maintenance intervention on component j and n_{Mj} being the number of maintenance interventions on component j throughout the mission time T_M.

$C_{team} = \sum_{k=1}^{N_R} n_k C_{team,k}$ is the total amount of money paid for salaries to the repair teams, N_R is the number of different types of repair teams; $C_{team,k}$, k=1, 2, ..., N_R being the cost of a team specialized in repairs of kind k and n_k being the number of such teams.

As before, we resort to the Monte Carlo method for its evaluation. For a fixed maintenance and repair strategy, i.e. for fixed values of the maintenance parameter α_τ and of the number n_k of repair teams of different types, k=1, 2, ..., N_R, we simulate a specified number of system histories and eventually estimate the mean values and standard deviations of the following quantities :

T_D = system downtime

n_{Mj} = number of maintenance intervention on component j, j=1,...,N_C

T_R (j) = total time upon repair of component j, j=1,...,N_C

P_{ACC} (k) = probability of an accident of type k

A(t) = instantaneous system availability at time t

These quantities allow us to compute the various terms constituting the system operation gain of eq. (15). As before, running a full Monte Carlo simulation for each potential combination of α_τ and n_k values is an unconceivable task. We then resort to the same combination of genetic algorithms and Monte Carlo simulation in the 'drop-by-drop' model previously illustrated.

In order to exploit the genetic algorithm for the search of the optimal maintenance and repair strategy we need to code the possible solutions into distinct bit-strings chromosomes. To specify a possible maintenance and repair strategy we need to assign values to the 1+N_R parameters α_τ and n_k, k=1, 2, ..., N_R which characterize the maintenance and repair teams logistics, respectively. In our work these values are coded into a chromosome made up of 1+N_R genes. The first gene is devoted to the coding of the values of the maintenance parameter α_τ. The number of bits for this gene can be chosen by the analyst and determines the discretization of the range of possible values to be spanned in the search (note that since α_τ is a real-valued parameter, we could have used a real-value encoding as well). The subsequent N_R genes are used to code the possible numbers of repair teams n_k for repairs of types k=1, 2, ..., N_R. Without getting into too much details, we simply point out that the coding in this case is devised so as to be physically reasonable in that the possible values for the number of repair teams are integers which range from 0 (no teams from that kind of repair action) to the number of components potentially needing that kind of repair.

The system we consider (Figure 12) represents an adaptation of a chemical process plant previously presented in literature [11]. The model contains stand-by units, load connections, induced failures and k-out-of-n modules. Components A and B form a warm stand-by unit. While component A is operational, component B will be in stand-by, at which state it has a reduced failure rate. If component A fails, component B immediately activates (assuming, for simplicity, a perfect switch).

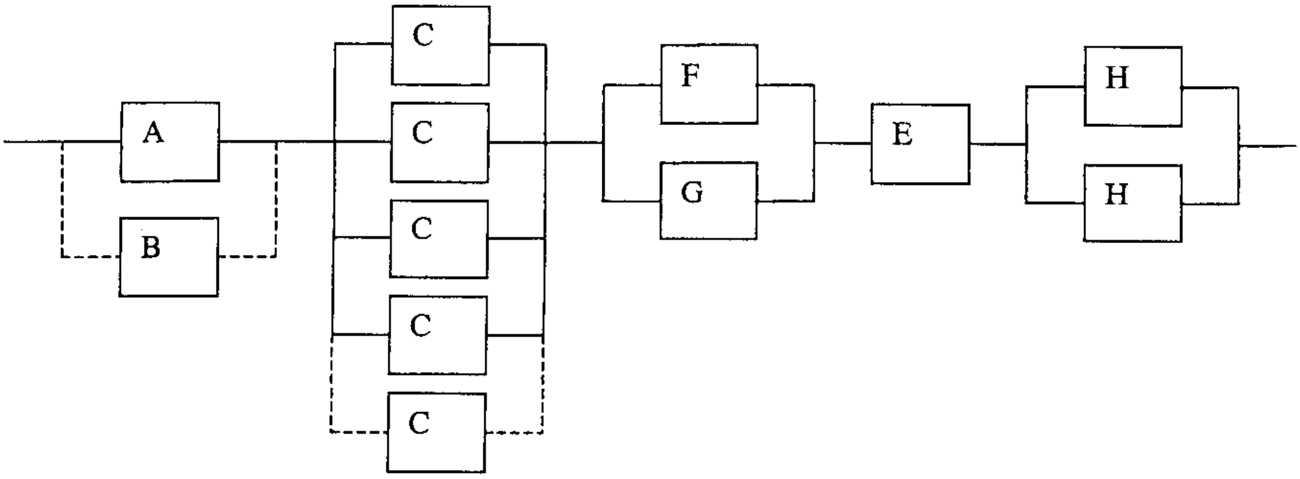

Figure 12. Schematics of the chemical process plant. Solid line = connections of active components. Dotted line = connections of standby components.
A : power supply; B : backup power supply; C : chemical processing unit; F : conveyer; G : conveyer engine; E : filtering unit; H : filtering conveyer.

In our Monte Carlo simulation model the active-to-standby and standby-to-active transitions are dealt with by establishing a proper correlation between the system states before and after the transition, following the multiplicative correlation model proposed in Ref. [12]. This model allows one to modify the transition rates by multiplying the original values times a pre-defined multiplication factor which depends on the current and arrival configurations. The components C are arranged into a 4-out-of 5G cold standby configuration. Failure of this configuration is considered an accident as it leads to a system failure with damages to the outside. There is an operational connection between components A,B and components C which is such that a failure of either one of the components A and B increases the likelihood of failures in the C-components. To treat this connection a correlation between the component states is established [12] which is such to multiply the failure rates of the C-components by 1.5 when a failure of A or B occurs. Components F and G work in a 2-out-of-2G configuration whereas the two H-components are in parallel (1-out-of-2G) but they are related through a load connection according to which failure of one of them leads to an increased load on the surviving component which is then forced to work in more stressful conditions so that its failure rate increases. This is again accounted for by means of a multiplicative connection with factor 1.5. Finally, there are three kinds of different repair actions: type 1 for components A, B and G; type 2 for components F and H; type 3 for components C and E. Repair priorities are assigned to the various components according to their functional importance. Tables 5 and 6 report the failure and repair data and the cost data, respectively, for the components. Table 7 contains the data pertaining to the system operating life.

The optimization regards finding the values of the maintenance parameter α_τ and of the numbers of repair teams, n_1, n_2 and n_3, which maximize the net gain as defined in eq.(15). For α_τ we have chosen the range [1, 10] and 5 bits, so that the range is discretized in 32 possible values which become candidate solutions for the optimal maintenance strategy. As for what concerns the number of repair teams of the three kinds considered, this varies from a minimum of 0 to a maximum of 3, 3 and 6, respectively, with codings of 2, 2 and 3 bits, respectively. The number of candidate solutions for the optimal maintenance and repair strategy is then 3584. A genetic algorithm with a population of 50 chromosomes was run for the optimization (Table 8). Each chromosome is made up of four genes of 5, 2, 2 and 3 bits, respectively. For the Monte Carlo evaluation of each candidate solution proposed by the genetic algorithm 1000 trials are run the first time a chromosome appears and then 100 for all successive appearances.

Table 5. Component failure and repair data.

Comp.	$\lambda_0(y^{-1})$	$a(y^{-1})$	$\mu(y^{-1})$	$\lambda_0^{sb}(y^{-1})$	p	π_λ	π_μ	k-repair	priority
A	$3.1\ 10^{-2}$	$6.2\ 10^{-3}$	1.50	10^{-2}	10^{-1}	1.1	1.1	1	1
B	$4.8\ 10^{-2}$	$9.6\ 10^{-3}$	1.60	10^{-2}	10^{-1}	1.1	1.1	1	2
C	$1.2\ 10^{-1}$	$1.2\ 10^{-3}$	1.20	-	$2\ 10^{-1}$	1.2	1.1	3	1
F	$2.5\ 10^{-2}$	$2.5\ 10^{-4}$	2.50	-	$2\ 10^{-1}$	1.2	1.1	2	1
G	$2.8\ 10^{-2}$	$5.6\ 10^{-3}$	1.28	-	10^{-1}	1.1	1.1	1	1
E	$7.3\ 10^{-2}$	$7.3\ 10^{-4}$	2.30	-	$2\ 10^{-1}$	1.2	1.1	3	2
H	$4.5\ 10^{-2}$	$4.5\ 10^{-4}$	3.28	-	$2\ 10^{-1}$	1.2	1.1	2	2

P= probability of a deteriorating repair according to the Brown – Proschan model
k-repair = kind of repair

Table 6. Component cost data.

Comp.	C_j (\$)	C_{Mj} (\$)	C_R (\$)
A	$3\ 10^3$	$5\ 10^3$	10^4
B	$3\ 10^3$	$5\ 10^3$	10^4
C	$1\ 10^3$	$2\ 10^3$	$4\ 10^3$
F	$1\ 10^3$	$1\ 10^3$	$2\ 10^3$
G	$3\ 10^3$	$5\ 10^3$	10^4
E	$2\ 10^3$	$3\ 10^3$	$6\ 10^3$
H	$2\ 10^3$	$4\ 10^3$	$8\ 10^3$

j=A,B,...,H

The application of the optimization approach presented above results in a best solution consisting of the following values: $\alpha_\tau = 1.70$, $n_1 = 0$, $n_2 = 1$, $n_3 = 3$. This means that maintenance actions are performed at large time intervals (recall that the range for α_τ is [1,10] and that, according to eq.(14), the components maintenance periods decrease with increasing α_τ). For what concerns the repair logistics, the high costs of repairs of components A, B and G and their good reliability characteristics make it convenient not to perform repairs on them so that there is no need to keep repair teams for repairs of type 1 (which are the most expensive specialists) on the paycheck (n_1 = 0). For what concerns repairs of type 2, the criticality of unit F, which must function for the system to function, its relatively low repair cost and the low salaries paid to specialized repairmen for this kind of repair, suggest that at least one team be hired at all times in the system ($n_2 = 1$). Finally, given the criticality of component E and the low reliability of the 5 C-components in the 4-out-of-5G arrangement, $n_3 = 3$ specialized teams for repairs of type 3 are found to be optimal. With these choices, the system achieves an average availability over the mission time of only 0.64.

Table 7. System data.

Profit per unit time P_t	**($\$ y^{-1}$)**	$4.2\ 10^4$
Downtime penalty per unit time C_U	**($\$ y^{-1}$)**	$5\ 10^4$
Accident reimbursement cost C_{ACC}	**($)**	$2.5\ 10^4$
Repair teams cost per unit time	**($\$ y^{-1}$):**	
$C_{team,1}$		50
$C_{team,2}$		10
$C_{team,3}$		20
Interest rate i		0%
Mission time T_M	**(y)**	10

Table 8. Genetic Algorithm rules and parameters.

α_τ n. bits	n_1 n. bits	n_2 n. bits	n_3 n. bits	Population Size	Parents Selection	Crossover	Replacement	Mutation Probability
5	2	2	3	50	Standard Roulette	Single Random Splice	Keep fittest 2 out of the 2 parents & 2 children	10^{-3}

Table 9 reports the solutions proposed by the optimization procedure when we consider different values for the costs of repair teams service. The repair teams costs are varied by multiplying their base case values of Table 7 by a factor θ_{team} which varies between 10^{-2} and 10^2. The optimization still regards both α_τ and the numbers of repair teams. Intuitively, for low salaries paid to the repair teams ($\theta_{team} = 10^{-2}$), it is convenient to have several of them on site. Interestingly enough, even in this case the high costs per unit time of the single repair actions on components A, B and G, which need such repair, and the good reliability characteristics of these components still make it convenient not to perform repairs on them, so that $n_1 = 0$. Instead, the more affordable repairs of the other components are such that n_2 and n_3 are at their maximum values of 3 and 6, respectively. Moreover, in this situation the repairs are so efficient and convenient that some relaxation is possible on the components maintenance, as shown by the reduced value of α_τ. For larger costs of repair teams ($\theta_{team} \geq 10^{-1}$),the optimization leads to a solution with smaller numbers of teams. With very few repair teams (when θ_{team} = 10 or 100), the profit decreases whereas the downtime costs increase significantly and the maintenance periods are slightly extended so as to save some resources from maintenance activities.

Table 9. Sensitivity of the optimal solution to the cost of repair teams.

	$\theta_{team}=10^{-2}$	$\theta_{team}=10^{-1}$	θ_{team}=1(base)	$\theta_{team}=10^{1}$	$\theta_{team}=10^{2}$
α_τ	1.14	1.70	1.70	1.42	1.42
n_1	0	0	0	0	0
n_2	3	3	1	0	0
n_3	6	3	3	1	0
G(10^3 \$)	267.46±1.04	267.42±1.30	270.23±2.07	262.77±0.09	252.92±0.10
P(10^3 \$)	390.33	391.42	392.74	389.15	386.28
C_R(10^3 \$)	8.71	8.72	8.05	6.36	0
C_M(10^3 \$)	3.86	6.08	6.21	3.68	1.674
C_{team}(10^3 \$)	1.40 10^{-2}	9.60 10^{-2}	1.16	2.16	0
C_D(10^3 \$)	72.86	71.53	69.47	75.94	84.32
C_{ACC}(10^3 \$)	16.13	15.85	14.63	17.25	23.37
A_m	0.62	0.62	0.64	0.61	0.44

θ_{team} = factor multiplying the base case repair teams costs for repairs of types 1, 2 and 3 of Table 7.

When repair teams are a significant presence in the plant analysis ($\theta_{team} = 10^{-2}, 10^{-1}$), the cost due to repairs is, overall, larger as it is also that due to maintenance because of the deteriorating effects of the repairs which, according to the Brown-Proschan model here adopted, entail more frequent maintenance interventions. For fewer repair teams, and consequent fewer repair actions, repair and maintenance costs overall decrease but, obviously, the conflict arises because the system availability and reliability worsens so that, as pointed out, downtime and accident costs increase significantly. These behaviours are pictorially reported in Figure 13 for what concerns the repair, downtime and accident costs.

Table 10 and Figures 14-15 show the effect of varying the cost of maintenance actions. This is obtained by multiplying the base case values of Table 6 by a factor θ_M which varies between 0 and 5. The increase in the cost of maintenance leads to less frequent maintenance interventions, as shown by the decreasing value of the factor α_τ in Figure 14 and by the behavior of the individual components maintenance periods in Figure 15. Also, the data chosen are such that in spite of the fewer maintenances, the increase in the cost of maintenance actions leads to higher overall system maintenance expenses C_M (Figure 14). Reducing maintenance also worsens the reliability/availability performance of the system so that its downtime cost C_D also increases (Figure 14): this has a major effect on the overall system gain G which decreases when maintenance efforts are reduced, notwithstanding some cuts in expenditures for repair teams and, consequently, in repair actions (Table 10).

Finally, it is worthwhile noticing that the above results are robust in the sense that they do not essentially change by varying the chromosome population inizialization. As far as the CPU time is concerned, in some of the cases the genetic algorithm was capable of arriving at convergence rather quickly, after only few hundreds of generations, while for other cases the procedure was stopped in correspondence of the maximum allowed number of generations, set to 1000 in our case. Correspondingly, the computation time needed for the optimization ranged from a minimum of 18 minutes to a maximum of 160 minutes, on an Alpha Station 250 4/266, and different degrees of accuracy were attained by the 'drop-by-drop' method as shown by the statistics on G in Tables 9 and 10.

Table 10. Sensitivity of the optimal solution to the maintenance costs.

	θ_M=0	**θ_M=0.2**	**θ_M=1(base)**	**θ_M=5**
α_τ	9.30	2.83	1.70	1.14
n_1	0	0	0	0
n_2	1	3	1	0
n_3	3	4	3	2
G(10^3 \$)	281.59±2.46	278.31±1.78	270.23±2.07	256.45±0.10
P(10^3 \$)	394.07	396.40	392.74	388.30
C_R(10^3 \$)	7.93	8.15	8.05	7.38
C_M(10^3 \$)	0	3.82	6.21	10.76
C_{team}(10^3 \$)	7.40 10^{-1}	1.16	1.16	4.30 10^{-1}
C_D(10^3 \$)	65.50	67.13	69.47	75.87
C_{ACC}(10^3 \$)	13.82	14.43	14.63	16.41
A_m	0.64	0.65	0.64	0.61

θ_M = factor multiplying the base case component maintenance costs of Table 6.

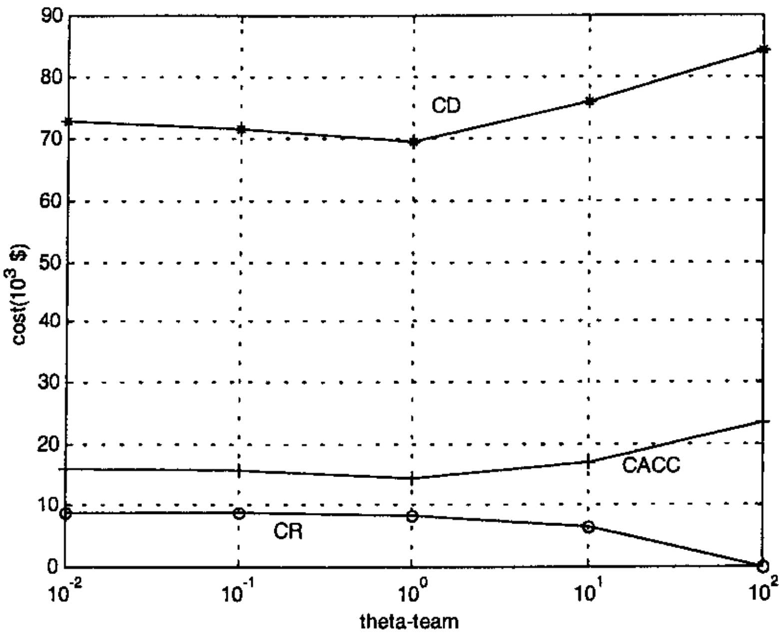

Figure 13. System Repair (circles), Downtime (stars) and Accident Costs (solid line) as a function of Repair teams cost parameter θ_{team} (from Table 9).

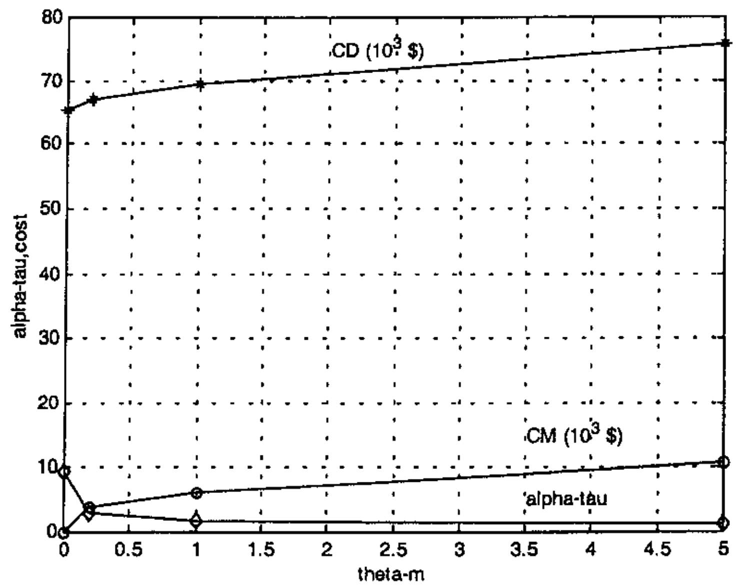

Figure 14. System Maintenance cost (circles), Downtime cost (stars) and maintenance parameter α_τ (diamonds) as a function of maintenance cost parameter θ_M (from Table 10).

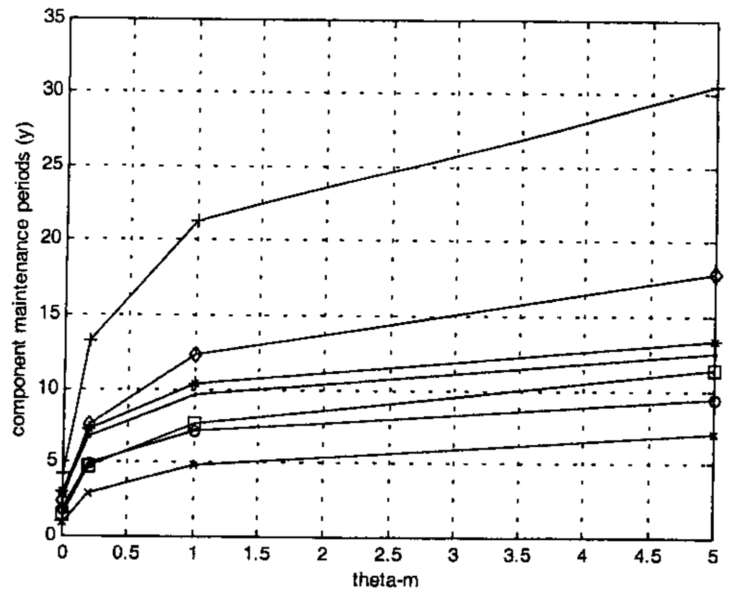

Figure 15. Components maintenance periods as a function of maintenance cost parameter θ_M:
A = dots; B = circles; C = x's; F = crosses; G = stars; E = squares; H = diamonds

11. Discussion

The goal of this paper was to provide an overview of the potentials of genetic algorithms for optimization problems in risk and safety analysis. The basics behind the functioning of genetic algorithms have been presented. The application of such algorithms have been illustrated in three different optimization problems with rather large search spaces. The first application considers the optimization of the pressure stages of a natural gas pipeline system. The simplicity of the objective function for this case allowed for its analytical evaluation. The second and third applications were concerned with the choice of the optimal system configuration and maintenance and repair strategies, respectively, which provide the maximum profit of running a risky plant. For the former, the choice regarded the types of components and their assembly; for the latter the choice concerned the components' maintenance periods and the numbers of repair teams to have available for each kind of specialized repair requested in the plant. The profit function was quite realistic so that its complexity entailed a Monte Carlo solution. However, given that several parameters are involved, the search for an optimal solution in a test-error fashion would require an excessively large number of different Monte Carlo evaluations, which would render the problem computationally impracticable: one would then necessarily be forced to limit the analysis at only few points in the search space and, most likely, be satisfied by a solution far from optimal.

To overcome this problem we have embedded the Monte Carlo evaluation of plant operation within the Genetic Algorithms-maximization procedure. The genetic algorithms procedure efficiently guides the search for an optimum of a profit function which accounts for the plant safety and economic performance and which is evaluated, for each possible configuration choice, by the above Monte Carlo simulation. In the proposed 'drop-by-drop' approach, each potential solution proposed by the genetic algorithm is explored only by few hundreds Monte Carlo histories. Then, due to the fact that during the genetic evolution the fit chromosomes appear repeatedly many times, statistically significant results for the solutions of interest (i.e., the best ones) are obtained. This approach coupled with the 'evolutionary guidance' in the search procedure by the genetic algorithms allows one to efficiently perform the analysis of a realistic system in reasonable computing times.

References

1. Goldberg D.E., Genetic Algorithms in Search, Optimization, and Machine Learning. *Addison-Wesley Publ. Co.*, (1989).
2. Wong P.J. and Larson R.E., Optimization of Natural-Gas Pipeline Systems Via Dynamic Programming. *IEEE Trans. Aut. Control*, Vol. AC-13, No.5, October 1968.
3. Joyce, P.A., Withers, T.A. and Hickling, P.J., Application of Genetic Algorithms to Optimum Offshore Plant Design. *Proceedings of ESREL 98*, Trondheim (Norway), June 16-19, (1998) pp. 665 – 671.
4. Henley, E.J. and Kumamoto, H., Probabilistic Risk Assessment. *IEEE Press*, (1992) pp. 554 – 561.
5. Brown, M. and Proschan, F., Imperfect Repair. *Journal of Applied Probability*, **20** (1983) pp. 851-859.
6. NUREG/CR-4769, Risk Evaluations of Aging Phenomena: The Linear Aging Reliability Model and Its Extension. *US Nuclear Regulatory Commission*, April 1987.
7. Vaurio, J.K., On time-dependent availability and maintenance optimization of standby units under various maintenance policies. *Reliab. Engng. and Sys. Safety*, **56** (1997) pp. 79-89.
8. Vesely, W.E., Davis, T.C., Denning, R.S. and Saltos, N., Measures of risk importance and their applications. *NUREG/CR-3385, US Nuclear Regulatory Commission*, (1983).
9. Cheok, M.C., Parry G.W. and Sherry, R.R., Use of importance measures in risk-informed regulatory applications. *Reliab. Engng. and Sys. Safety*, **60** (1998) pp. 213-226.
10. Borgonovo, E., Marseguerra, M. and Zio, E., A Monte Carlo Methodological Approach to Plant Availability Modelling with Maintenance, Ageing and Obsolescence. Accepted for publication in Reliab. Engng. And Sys. Safety, (1999).
11. Dubi, A., Analytic approach & Monte Carlo methods for realistic systems. *Tutorial notes* (April 1997).
12. Marseguerra, M. and Zio, E., Nonlinear Monte Carlo reliability analysis with biasing towards top event. *Reliab. Engng. and Sys. Safety*, **40** (1993) pp. 31-42.