Introduction to Genetic Algorithms

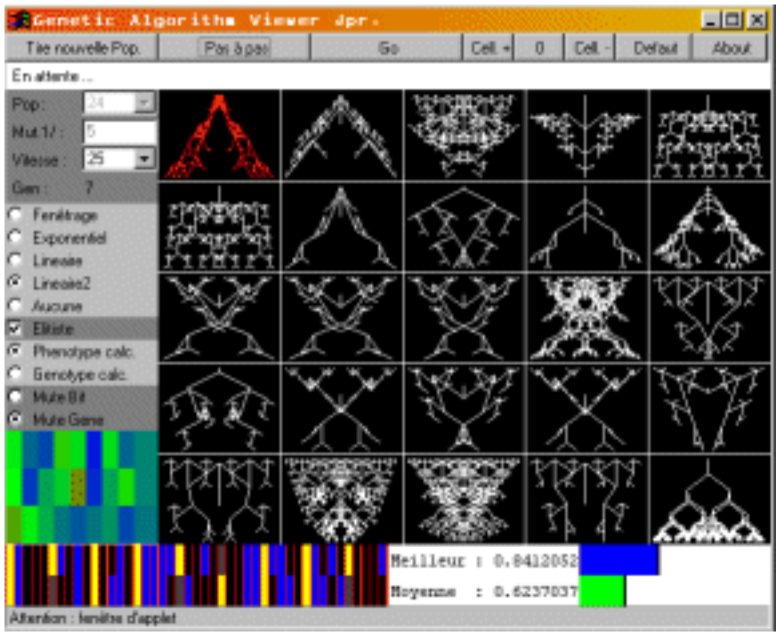
Genetic Algorithm Viewer 1.0

Download Applet, Java sources and Javadoc

Some Statistics about GAV

Download that text in PDF

Get Acrobat Reader



En attente...

Pop : 24
Mut 1/ : 5
Vitesse : 25
Gen : 7

Meilleur : 0.8412052
Moyenne : 0.6237037

Attention : fenêtre d'applet

Genetic Algorithm Viewer shows the functioning of a genetic algorithm. It permits the user to test the major parameters of a genetic algorithm.

rated jars top25%

Introduction to Genetic Algorithms.

Physics, Biology, Economy or Sociology often have to deal with the classical problem of optimization. Economy particularly has become specialist of that field[1]. Generally speaking, a large part of mathematical development during the XVIIIth century dealt with that topic (remember those always repeated problems where you had to obtain the derivative of a function to find its extremes).

Purely analytical methods widely proved their efficiency. They nevertheless suffer from a insurmountable weakness : Reality rarely obeys to those wonderful differentiable functions your professors used to show you[2].

Other methods, combining mathematical analysis and random search have appeared. Imagine you scatter small robots in a Mountainous landscape. Those robots can follow the steepest path they found. When a robot reaches a peak, it claims that it has found the optimum. This method is very efficient, but there's no proof that the optimum has been found, each robot can be blocked in a local optimum. This type of method only works with reduced search spaces.

What could be the link between optimization methods and artificial life ?

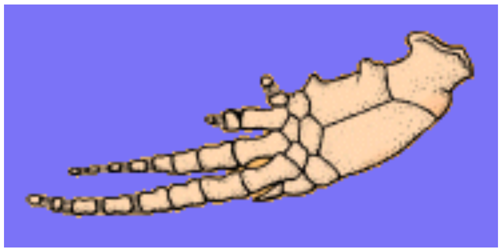
A- Evolution and optimization.

We are now 45 millions years ago examining a Basilosaurus :

Basilosaurus

The Basilosaurus was quite a prototype of a whale. It was about 15 meters long for 5 tons. It still had a quasi-independent head and posterior paws. He moved using undulatory movements and hunted small preys[3]. Its anterior members were reduced to small flippers with an elbow articulation.

Movements in such a viscous element (water) are very hard and require big efforts. People concerned must have enough energy to move and control its trajectory. The anterior members of basilosaurus were not really adapted to swimming[4]. To adapt them, a double phenomenon must occur : the shortening of the "arm" with the locking of the elbow articulation and the extension of the fingers which will constitute the base structure of the flipper.



Tursiops flipper

The image shows that two fingers of the common dolphin are hypertrophied to the detriment of the rest of the member.

The basilosaurus was a hunter, he had to be fast and precise. Through time, subjects appeared with longer fingers and short arms. They could move faster and more precisely than before, and therefore, live longer and have many descendants.

Meanwhile, other improvements occurred concerning the general aerodynamic like the integration of the head to the body, improvement of the profile, strengthening of the caudal fin ... finally producing a subject perfectly adapted to the constraints of an aqueous environment.

This process of adaptation, this morphological optimization is so perfect that nowadays, the similarity between a shark, a dolphin or a submarine is striking. But the first is a cartilaginous fish (Chondrichtyen) originating in the Devonian (-400 million years), long before the apparition of the first mammal whose Cetacean descend from[5].

Darwinian mechanism hence generate an optimization process[6], Hydrodynamic optimization for fishes and others marine animals, aerodynamic for pterodactyls, birds or bats. This observation is the basis of genetic algorithms.

B- Evolution and Genetic Algorithms

John Holland, from the University of Michigan began his work on genetic algorithms at the beginning of the 60s. A first achievement was the publication of *Adaptation in Natural and Artificial System*[7] in 1975.

Holland had a double aim : to improve the understanding of natural adaptation process, and to design artificial systems having properties similar to natural systems[8].

The basic idea is as follow : the genetic pool of a given population potentially contains the solution, or a better solution, to a given adaptive problem. This solution is not "active" because the genetic combination on which it relies is split between several subjects. Only the association of different genomes can lead to the solution. Simplistically speaking, we could by example consider that the shortening of the paw and the extension of the fingers of our basilosaurus are controlled by 2 "genes". No subject has such a genome, but during reproduction and crossover, new genetic combination occur and, finally, a subject can inherit a "good gene" from both parents : his paw is now a flipper.

Holland method is especially effective because he not only considered the role of mutation (mutations improve very seldom the algorithms), but he also utilized genetic recombination, (crossover)[9] : these recombination, the crossover of partial solutions greatly improve the capability of the algorithm to approach, and eventually find, the optimum.

C- Functioning of a Genetic Algorithm

As an example, we're going to enter a world of simplified genetic. The "chromosomes" encode a group of linked features. "Genes" encode the activation or deactivation of a feature.

Let us examine the global genetic pool of four basilosaurus belonging to this world. We will consider the "chromosomes" which encode the length of anterior members. The length of the "paw" and the length of the "fingers" are encoded by four genes : the first two encode the "paw" and the other two encode the fingers.

In our representation of the genome, the circle on blue background depict the activation of a feature, the cross on green background depict its deactivation. The ideal genome (short paws and long fingers) is : ⊠⊠⊙⊙.

The genetic pool of our population is the following one :

Subject	Genome
A	⊙⊠⊠⊠
B	⊠⊙⊠⊠
C	⊠⊙⊠⊙
D	⊠⊠⊙⊠

We can notice that A and B are the closest to their ancestors ; they've got quite long paws and short fingers. On the contrary, D is close to the optimum, he just needs a small lengthening of his fingers.

This is such a peculiar world that the ability to move is the main criteria of survival and reproduction. No female would easily accept to marry basilosaurus whose paws would look like A's. But they all dream to meet D one day.

The fitness is easy to compute : we just have to give one point to each gene corresponding to the ideal. The perfect genome will then get four points. The probability of reproduction of a given subject will directly depend on this value. In our case, we'll get the following results :

Subject	Fitness	Reproduction probability
A	1	1/7 = 0.143
B	1	1/7 = 0.143
C	2	2/7 = 0.286
D	3	3/7 = 0.428
Total	7	7/7=1

We'll consider a cycle of reproduction with for descendants, i.e. four mating concerning height subjects. D will be selected four times and will then get four descendants. C will be selected twice and will get two descendants. Finally A and B will only be selected once.

The reproduction pattern is the following :

Subject	Received genes	Genome	Fitness	Reproduction probability
A'	A : ⊙⊠ D : ⊙⊠	⊙⊠⊙⊠	2	2/10=0.2
B'	B : ⊠⊙ D : ⊙⊠	⊠⊙⊙⊠	2	2/10=0.2

C'	D : ☒☒ C : ☒◯	☒☒☒◯	3	3/10=0.3
D'	C : ☒ D : ☒◯☒	☒☒◯☒	3	3/10=0.3
Total			10	10/10=1

During reproduction crossovers occur at a random place (center of the genome for A', B' and C', just after the first gene for D'). The link existing between the degree of adaptation and the probability of reproduction leads to a trend to the rise of the average fitness of the population. In our case, it jumps from 7 to 10.

During the following cycle of reproduction, C' and D' will have a common descendant :

D' : ☒☒◯ + C' : ◯ = ☒☒◯◯

The new subject has inherited the intended genome : his paws have become flippers.

We can then see that the principle of genetic algorithms is simple :

1. Encoding of the problem in a binary string.
2. Random generation of a population. This one includes a genetic pool representing a group of possible solutions.
3. Reckoning of a fitness value for each subject. It will directly depend on the distance to the optimum.
4. Selection of the subjects that will mate according to their share in the population global fitness.
5. Genomes crossover and mutations.
6. And then start again from point 3.

The functioning of a genetic algorithm can also be described in reference to genotype (GTYPE) and phenotype (PTYPE) notions[10].

1. Select pairs of GTYPE according to their PTYPE fitness.
2. Apply the genetic operators (crossover, mutation...) to create new GTYPE.
3. Develop GTYPE to get the PTYPE of a new generation and start again from 1.

Crossover is the basis of genetic algorithms, there is nevertheless other operators like mutation. In fact, the desired solution may happen not to be present inside a given genetic pool, even a large one. Mutations allow the emergence of new genetic configurations which, by widening the pool improve the chances to find the optimal solution. Other operators like inversion are also possible, but we won't deal with them here.

D- Adaptation and Selection : the scaling problem

We saw before that in a genetic algorithm, the probability of reproduction directly depends on the fitness of each subject. We simulate that way the adaptive pressure of the environment.

The use of this method nevertheless set two types of problems :

1. A "super-subject" being too often selected the whole population tends to converge towards his genome. The diversity of the genetic pool is then too reduced to allow the genetic algorithm to progress.
2. With the progression of the genetic algorithm, the differences between fitness are reduced. The best ones then get quite the same selection probability as the others and the genetic algorithm stops progressing.

In order to palliate these problems, it's possible to transform the fitness values. Here are the four main methods :

1- Windowing : For each subject, reduce its fitness by the fitness of the worse subject. This permits to strengthen the strongest subject and to obtain a zero based distribution.

2- Exponential : This method, proposed by S.R. Ladd[11], consists in taking the square roots of the fitness plus one. This permits to reduce the influence of the strongest subjects.

3- Linear Transformation : Apply a linear transformation to each fitness, i.e.
f ' = a.f + b. The strongest subjects are once again reduced.

4- Linear normalization : Fitness are linearized. For example over a population of 10 subjects, the first will get 100, the second 90, 80 ... The last will get 10. You then avoid the constraint of direct reckoning. Even if the differences between the subjects are very strong, or weak, the difference between probabilities of reproduction only depends on the ranking of the subjects.

To illustrate these methods, let's consider a population of four subjects to check the effect of scaling. For each subject, we give the fitness and the corresponding selection probability.

Subjects	1	2	3	4
Rough Fitness	50/50%	25/25%	15/15%	10/10%
Windowing	40/66.7%	15/25%	5/8.3%	0/0%
Exponential	7.14/36.5%	5.1/26.1%	4.0/20.5%	3.32/16.9%
Linear transfo.	53.3/44.4%	33.3/27.8%	20/16.7	13.3/11.1%
Linear normalization	40/40%	30/30%	20/20%	10/10%

Windowing eliminates the weakest subject - the probability comes to zero - and stimulates the strongest ones (the best one jumps from 50 % to 67 %).

Exponential flattens the distribution. It's very useful when a super-subject induces an excessively fast convergence.

Linear transformation plays slightly the same role than exponential.

At last, linear normalization is neutral towards the distribution of the fitness and only depends on the ranking. It avoids as well super-subjects as a too homogeneous distribution.

Conclusion

Genetic algorithms are original systems based on the supposed functioning of the Living[12]. The method is very different from classical optimization algorithms[13].

1. Use of the encoding of the parameters, not the parameters themselves.
2. Work on a population of points, not a unique one.
3. Use the only values of the function to optimize, not their derived function or other auxiliary knowledge.
4. Use probabilistic transition function not determinist ones.

It's important to understand that the functioning of such an algorithm does not guarantee success. We are in a stochastic system and a genetic pool may be too far from the solution, or for example, a too fast convergence may halt the process of evolution. These algorithms are nevertheless extremely efficient, and are used in fields as diverse as stock exchange, production scheduling or programming of assembly robots in the automotive industry.

1- The main paradigm of Economy (neo-classical) is largely just a wonderful ode to optimization mathematics.

2- That's what the recent "non linearity" revolution learned us.

3- S.J. Gould et al., *Le livre de la vie,* Seuil, Science ouverte, 1993, pp.186 ss.

4- It is said that basilosaurus reproduced on earth. In that case, posterior members were useful. Harrison R, Bryden M.M. dir., *Baleines, dauphins et marsouins*. Bordas, 1989.

5- This a very common phenomenon. In that case, we could also speak of the Ichthyosaure, marine reptile of the Mesozoic era whose morphology was closed to shark and dolphin.

6- It's important to understand that Darwinian process doesn't have to lead to optimum. It improves fitness but has nothing to do with optimum.

7- Holland J.H., *Adaptation in natural and artificial system,* Ann Arbor, The University of Michigan Press, 1975.

8- Goldberg D., *Genetic Algorithms,* Addison Wesley, 1988.

9- Emmeche C., *Garden in the Machine. The Emerging Science of Artificial Life,* Princeton University Press, 1994, pp. 114 ss.

10- Heudin J.C., *La Vie Artificielle,* Hermès, 1994, pp. 91 ss.

11- S.R. Ladd, *Genetic Algorithm in C++*, 1999-2000. Downloadable book. http://www.coyotegulch.com.

12- "Biological programming" is not limited to AG, another well-known case is neural networks.

13- Goldberg D, *idem*, pp. 8 ss.

Jean-Philippe Rennard. Ph.D. May 2000
http://www.rennard.org/alife
alife@rennard.org