A VERY SIMPLE STEADY-STATE EVOLUTIONARY ALGORITHM

L.Gacôgne * **

* LIP6 - Université Paris VI 104 av. du pdt Kennedy 75016 Paris
tel : 01 44 27 87 49 mail : Louis.Gacogne@lip6.fr
** ENSIIE 1 square de la résistance 91025 Evry

Abstract
A particular steady-state strategy of evolution with a small sized population is studied in this paper. We specially focus our attention on two ways to attempt a compromise with best parameters, we observe improvements in order to optimise classical problems when exploration is directly function of the homogeneity of the population.

Keywords *evolutionary algorithms - steady state genetic algorithms*

I INTRODUCTION

Evolutionary computation comprises different techniques that have been inspired by biological mechanisms. Beyond the canonical genetic algorithm (GA) [Holland 75], [Goldberg 82, 89], many ways of research intend to accelerate evolution in view of optimization problems with elitist heuristics [Schwefel 1995] or small population [Coello, Pulido 2000]. It is well known that a too homogeneous population must be avoid. A lot of different ideas have been proposed (clearing, changing the fitness, parallel evolution…) In previous works [Gacogne 2002, 2006] we showed that a small population with various operators is really better. We, now, focus on a simple way to favour heterogeneous population.
As a lot of heuristics in artificial intelligence, the tuning of parameters is difficult, but for the proposed strategy, we can reduce them to the population size, the updating rate and finally a migration rate which is a solution to the clearing way or a variable rate according to similarity inside the population.

II THE ALGORITHM SSGA(μ, τ, ξ)

The very simple option we chose, according to the steady-state updating, is to take the same idea to remove some individuals. To go on exploration searching in the same time exploitation of real-time results of optimum, we decide a rate ξ of exploration. Thus, each generation, in the population of μ individuals, the τ best children and a number ξ of random individuals remove the $\tau + \xi$ worst parents under the condition $0 < \tau$ and $\tau + \xi < \mu$.
Let us assume that for any positive function f from $[a, b]^m$ to R, in order to find the global minimum of f, we could choose a vector of m components $x = (x_1, x_2, \dots, x_m)$ evaluated by $f(x)$.

μ is the number of individuals in the population
ε is the threshold we want to reach to stop the run if we want $f(x) < \varepsilon$
$eval_{max}$ is the maximum number of evaluation of f to avoid infinite course

n_v is the number of evaluation (returned by the algorithm) to reach $f(x) < \varepsilon$ and we give the average of it on 100 runs

```
1) $t \leftarrow 0$, A population $P_0$ of $\mu$ random float-vector in $[0, 1]^m$ is built.
Each x of $P_0$ is evaluated by $f(x)$, $v_m$ is the best value of the $\mu$ individuals and $n_v \leftarrow \mu$
2) Loop while $v_m > \varepsilon$ and $n_v < eval_{max}$ do
        for $i = 1$ to $\mu$ do let $C_t(i) = op(P_t(i))$ where $op$ is a genetic operator randomly chosen
        between mutation or a crossover with another random $P_t(j)$
                ($C_t$ is called the population offspring and $P_t$ the parents)
        - Evaluations by $f$ of the $\mu$ children and sorting of $C_t$,
        -   Updating : let $P_{t+1}$ the sorted union of the best $\mu - \tau - \xi$ parents of $P_t$, the best
            $\tau$ elements of $C_t$ and $\xi$ new evaluated individuals picked in the space of research.
        - Sort of $P_{t+1}$ according to $f$ and
        - Incrementation $n_v \leftarrow n_v + \mu + \xi$
```

At the end of the course, the algorithm returns all information about the best value $f(x) < \varepsilon$ reached by the best x and the number of evaluation of f to reach it. A simple average for 100 runs of n_v gives a good idea for the parameters tuning.
More precisely we do experiments for $\mu = 3$ to 15, for $\tau = 1$ to $\mu - 2$, for $\xi = 0$ to $\mu - \tau - 1$, on three problems.

III RESULTS

To make tests with various problems, it is necessary to average runs which have a quite great dispersion. So, experimentally two parameters must be fixed. We chose to average on 100 runs, the number of evaluations of the function to be optimised and we chose a maximal number $eval_{max}$ = 5000 evaluations to the following functions the symbolic problem, according with experiments.

III-1 The "tripod" function.

The first test holds on this function [Gacôgne 2000] which is very fast to compute but hard to optimize the real minimum 0, because of three minima :

tripod(x, y) = if $y < 0$ then $|x| + |y + 50|$
        else if $x < 0$ then $1 + |x + 50| + |y - 50|$
        else $2 + |x - 50| + |y - 50|$

defined on $[-100, 100]^2$.

About the representation, if m = 4, an individual (a, b, c, d) is a pair :

(x, y) = (20a+2b-100, 20c+2d-100) to get :
(x, y) = (0, -50) that is tripod(5 0 2 5) = 0
(x, y) = (-50, 50), tripod(2 5 7 5) = 1
(x, y) = (50, 50), tripod(7 5 7 5) = 2

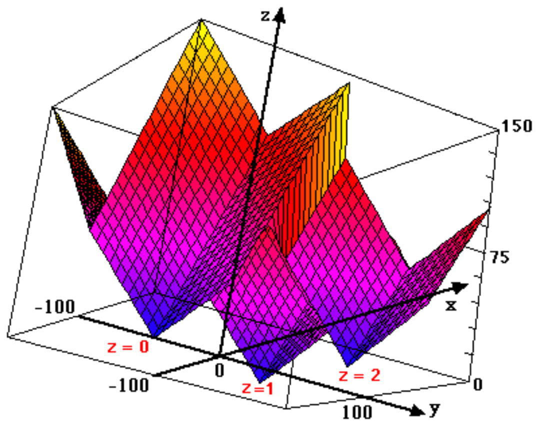

Figure 1 The tripod function

From the tests, we selected the rows $\tau = 1$ where best results are located and we show below that results are not very relevant, but we can say that $\mu = 6$ is a good result. When we choose $\xi = 1$, the best results for τ are as usual reached if τ is roughly the third part of μ (figure 3).

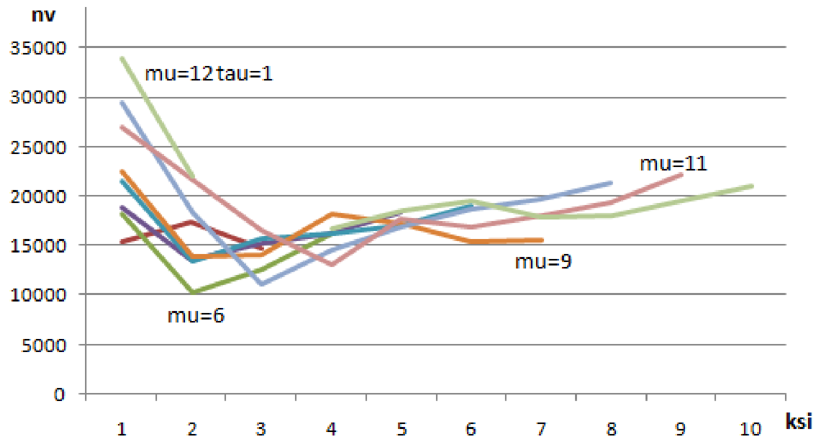

Figure 2 Averaging number of evaluation to reach 10^{-4} with $\tau = 1$, according to ξ.

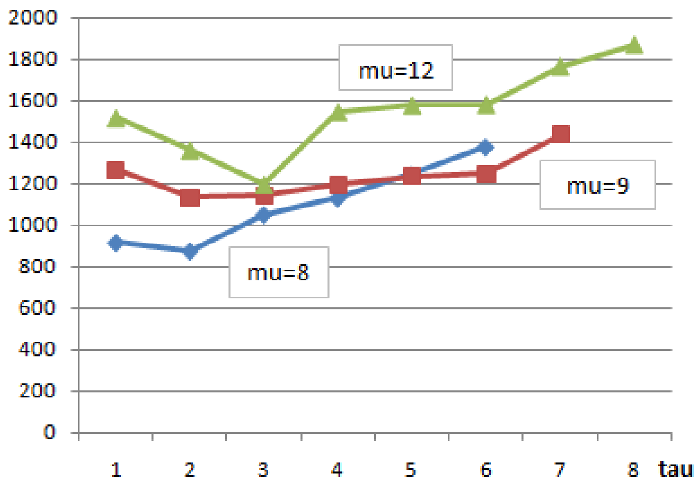

Figure 3 Averaging number of evaluation to reach 10^{-4} with $\xi = 1$, according to τ.

III-2 Tests for Rastrigin' s function

Our second test is on the well known function $F_R(x) = \Sigma_{i=1..n}[x_i^2 + 10 - 10\cos(2\pi x_i)] / 100$
For example, for the 3-dimensional Rastrigin's function, individuals could be 3*5 digits and we use a "decode" function to arrive in the space $[0, 1]^3$ and next in $[-30, 30]^3$ with the dilatation ϕ from *[0, 1]* to *[a, b]* defined by $\phi(x) = a + x(b - a)$.
Thus in one dimension *decode (3, 4, 5, 6, 7)* $\rightarrow$ *0.34567*

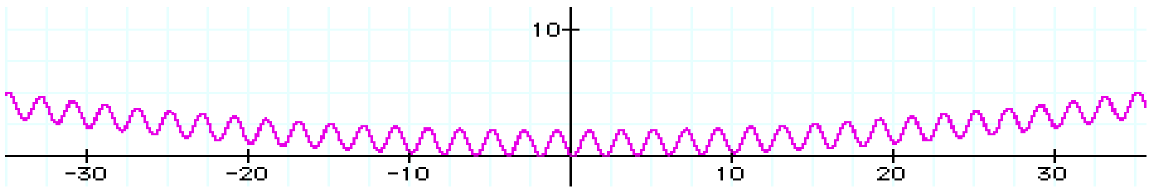

Figure 4 The Rastrigin's function in one dimension

In one dimension we first test the algorithm with all possibilities on small values for $\mu = 3$ to 15, for $\tau = 1$ to $\mu - 2$, for $\xi = 0$ to $\mu - \tau - 1$. For the value $\xi = 0$ (no exploration) results are roughly twice worse than for $\xi = 1$. 1 is very often but not always the best elimination rate.
That is why, we present on figure 5 and 6 all possibilities for $2 < \mu < 16$, $\tau < \mu - 2$. In any case the updating rate τ is at the better choice when being roughly the third part of μ.

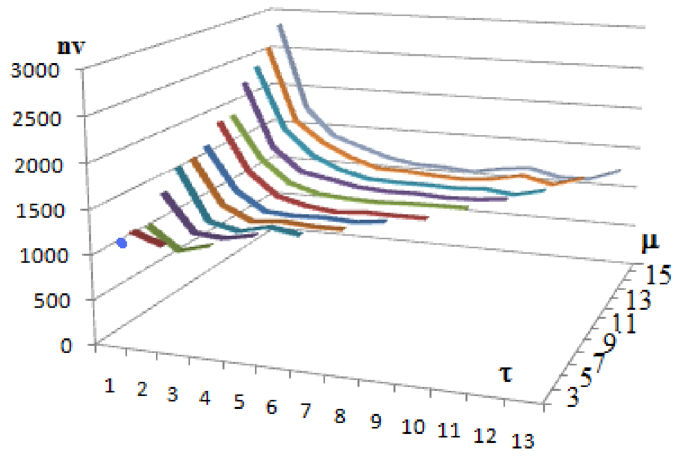

Figure 5 Average number of evaluation to reach 10^{-4} without exploration $\xi = 0$ with $\mu = 3$ to 15, $\tau = 1$ to $\mu - 2$

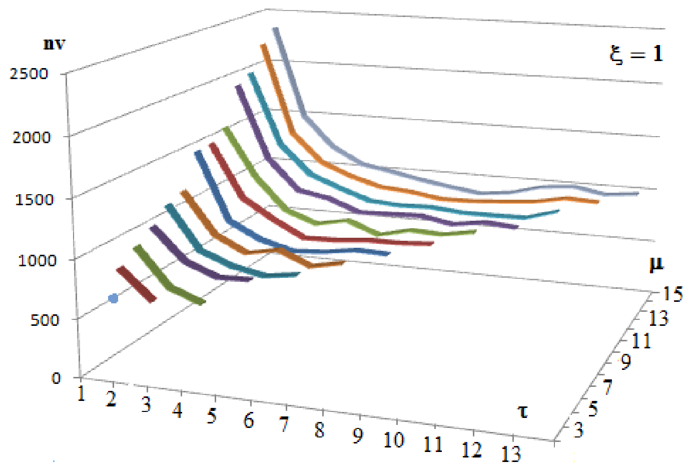

Figure 6 Average number of evaluations to reach 10^{-4} for $\xi = 1$ with $\mu = 3$ to 15, $\tau = 1$ to $\mu - 2$

III-3 Results for the Gauss' queens problem

We can solve the Gauss' queens problem taking the function *gauss* equal to the number of couple of queens in catching positions on a chessboard. For example, we minimize it to zero for the schema figure 7 in 7-dimension : (3 1 6 2 5 7 4)

queens 4 → (2 4 1 3)
queens 5 → (3 1 4 2 5)
queens 5 → (1 4 2 5 3)
queens 6 → (3 6 2 5 1 4)
queens 7 → (2 7 5 3 1 6 4)
queens 7 → (3 1 6 2 5 7 4)
queens 8 → (6 3 1 8 4 2 7 5)
queens 9 → (4 1 7 9 2 6 8 3 5)

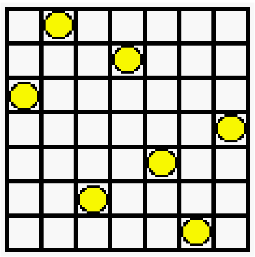

Figure 7 A solution : (3 1 6 2 5 7 4) for the Gauss queens' problem

In previous works, it is shown that dimension 6 is harder to solve that 7 or 8 dimensions. A lot of experiments for every possibilities from $\mu = 3$ to 15 have been performed and, the best results are always for $\xi = 1$. We eliminate the cases $\xi = 0$ which are always very worse than others.

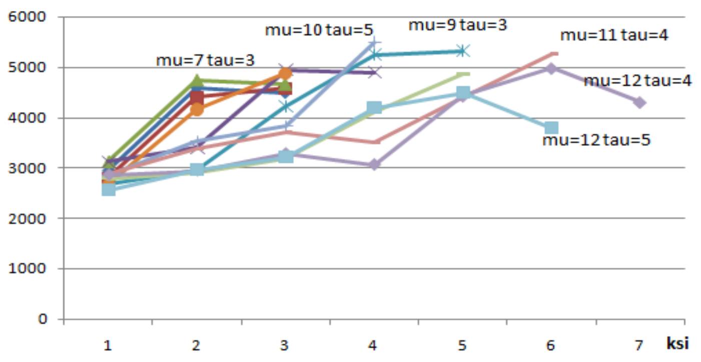

Figure 8 Average number of evaluations to reach a solution in the 6-Gauss queens problem

We can observe now that curves for dimension 7 or 8 are more regular :

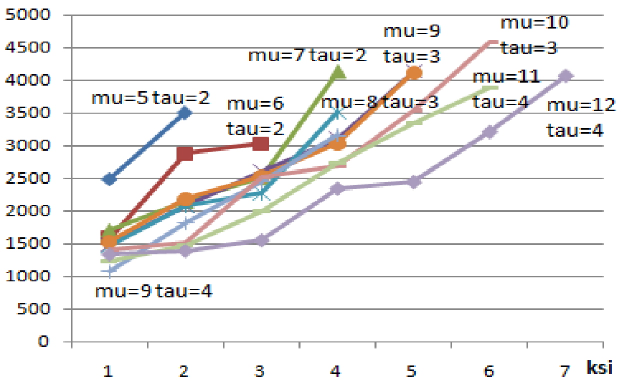

Figure 9 Average number of evaluations to reach a solution to 7-Gauss problem according to ξ

Comparison between SSGA(μ, τ, ξ) and SSGA(μ, τ, π)

In [Gacôgne 2002] we presented SSGA(μ, τ, π) as an algorithm where the rate of new random individuals is not constant but given by the state of the population each generation, that is to say, at the end of each generation, according to a similarity measure in the space of research, each time a pair of individual *i, j* with *f(i)* < *f(j)* and *prox(i, j)* > *π*, then j is killed and removed by a new individual.

It is natural to think that this way is better than a fixed rate ξ of removing, because this last option is not a security of homogeneity. A first experiment shows that it is not right; the very simple function *sum(a, b, c, ...)* = *a* + *b* + *c* .... for a sequence of digits a, b, c, ... in 0..9, which 0 as mimimum shows similar results (figure 10).

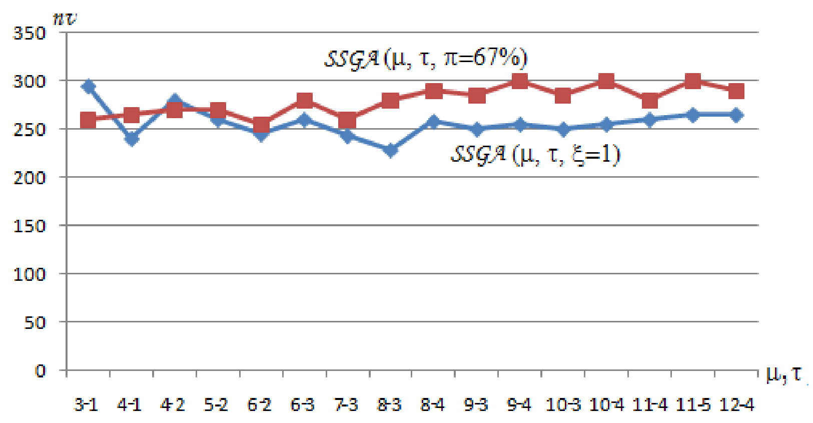

Figure 10 Average number of evaluations to reach a solution 0 for the sum of digits of a gene (a, b, c, d,...) with the best pairs (μ, τ) from (3, 1) untill (12, 4)

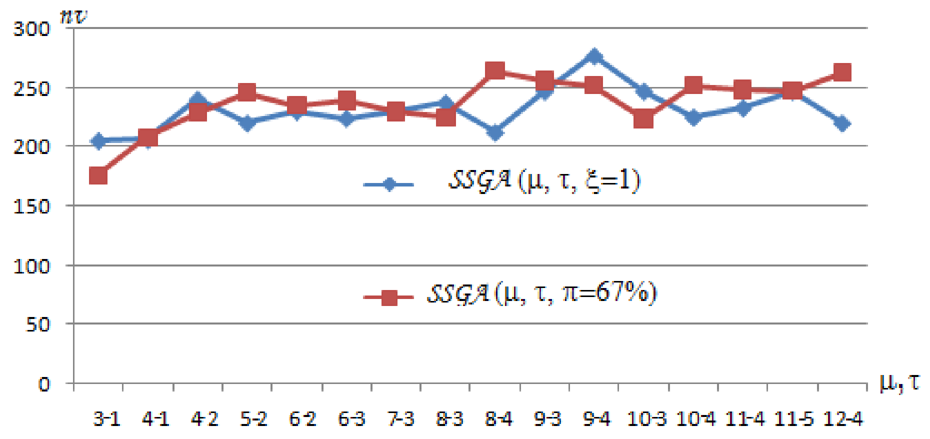

Figure 11 Average number of evaluations to reach 10^{-4} for the Rastrigin function in one dimension with the same (μ, τ) from (3, 1) untill (12, 4)

As we see, in spite of the last experiment, results in the two ways we tried are very closed for the Rastrigin function, so we show now what appens for the tripod function and the Gauss'queens problem 8*8, results are mostly better with elimination, that is to say with SSGA(μ, τ, π) but as we said in previous work, the rate of elimination π is not really relevant between 33% and 75%.

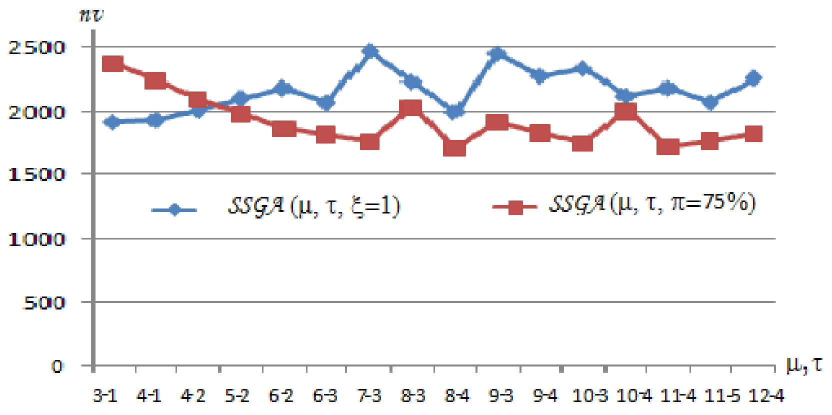

Figure 12 Average number of evaluations to reach 0 for the tripod function in two dimensions with the same pairs (μ, τ) from (3, 1) to (12, 4)

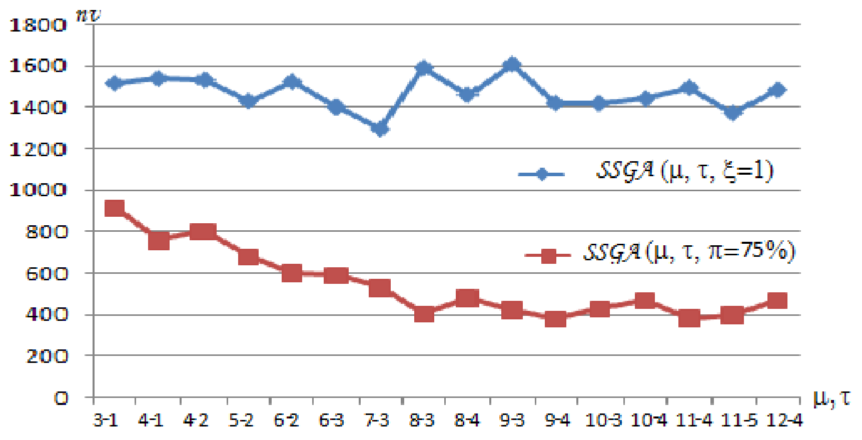

Figure 13 Average number of evaluations to reach 0 for the Gauss problem 8*8 with the pairs (μ, τ) from (3, 1) to (12, 4)

Conclusion

Experiments to find the averaging number of evaluation to reach global optimum always displays about the same results. First, best results are very often for a population size μ between 3 and 12 more precisely with $\mu = 7$ or 8 individuals and with an updating rate τ from 1/3 to 1/2 of it. Secondly, this number of evaluations is always increasing with ξ for the same μ, τ, it is, nevertheless better than when $\xi = 0$. So $\xi = 1$ is tested face to an other option which is a variable elimination rate π. According to the representation of the problem, if a similarity measure is defined on the genotypes, then clearing the population at each generation and replacement with new individuals is a satisfying way to keep the best individual and to go on the exploration

References

Bäck T. Fogel D.B. Schwefel H.P. *Handbook of evolutionary computation,* Oxford University Press, 1997

Coello C., Pulido G. A micro genetic algortihm for multiobjective optimization, Intrenal report Laboratorio National de Informatica Avanzada 2000

Davis L. *Adaptating operator probabilities in genetic algorithm,* Proc. 5th Int. Conf. on GA p61-69, Morgan K. 1993

De Jong K. Sarma J., *Generation Gaps Revisited*, in Foundations of G.A. Morgan Kaufmann, p5-17, 1993

Gacôgne L. *Benefit of a steady state genetic algorithm with adaptive operators,* Mendel Brno p236-242, 2000

Gacôgne L. *Steady-state evolutionary algorithm with an operators family,* EISCI Kosice p173-182, 2002

Gacôgne L. *Methods to apply operators in a steady-state evolutionary algortithm* IPMU conf. Paris 2006

Goldberg D.E. *Genetic algorithms in search, optimization and machine learning,* Addison Wesley, 1989

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

Michalewicz Z. *Genetic algorithms + data structures= evolution programs*, Springer Verlag 1992

Schwefel H.P. *Evolution and optimum seeking*. Sixth generation computer technology series. Wiley, 1995

Tvrdik J. Misik L. Krivy I. *Competing heuristics in evolutionary algorithms,* Intelligent Technology, Theory and Applications IOSpress vol 76 p159-165, 2002

Whitley D. Kauth J. *Genitor : a different genetic algorithm,* Proc. of Rocky Mountain Conf. on A.I. p118, 1988