Critical control of a genetic algorithm

Raphaël Cerf

Université Paris Sud and IUF

May 20, 2010

Abstract

Based on speculations coming from statistical mechanics and the conjectured existence of critical states, I propose a simple heuristic in order to control the mutation probability and the population size of a genetic algorithm.

Genetic algorithms are widely used nowadays, as well as their cousins evolutionary algorithms. The most cited initial references on genetic algorithms are the beautiful books of Holland [10], who tried to initiate a theoretical analysis of these processes, and of Goldberg [9], who made a very attractive exposition of these algorithms. The literature on genetic algorithms is now so huge that it is beyond my ability to compile a decent reasonable review. For years, there has been an urgent and growing demand for guidelines to operate a genetic algorithm on a practical problem. On the theoretical side, progress is quite slow and somehow disappointing for practitioners. The theoretical works often deal with a simple toy problem, otherwise the behavior of the genetic algorithm is too complex to be amenable to rigorous mathematical analysis. Here I propose a simple heuristic in order to control efficiently the genetic algorithm, based on speculations coming from statistical mechanics and the conjectured existence of critical states. Although it is quite simple, I have not been able to locate this heuristic in the literature, and I hope it will be useful. Apart from my own belief, it is supported by several empirical studies, the most notable one being the work of Ochoa [13], and it is in accordance with several conclusions and ideas appearing in the work of van Nimwegen and Crutchfield [17].

Error threshold. The fundamental notion on which the heuristic is based is the notion of error threshold, introduced by Manfred Eigen in 1971 [6]. Eigen analyzed a simple system of replicating molecules and demonstrated the existence of a critical mutation rate. Above this mutation rate, the information carried by the molecules is destroyed by the mutations. This fundamental result lead to the notion of quasispecies developed by Eigen, McCaskill and Schuster [8], which plays an important role in evolutionary biology. The critical mutation rate can be explicitly computed on the simplest fitness landscape: the space $\{0,1\}^n$ and the fitness function equal to 1 everywhere except one point where it is equal to $\sigma > 1$. In this situation, the critical mutation rate per bit is

$$p_m = \frac{\ln \sigma}{n}\,.$$

This is the sort of result we dream of in the context of genetic algorithms: an explicit simple expression for the critical mutation probability. Several researchers have already argued that the notion of error threshold plays a role in the dynamics of a genetic algorithm. This is far from obvious, because Eigen's model is formulated for an infinite population model. However there is evidence that a similar phenomenon occurs in finite populations as well, and also in genetic algorithms. In her PhD thesis [13], Ochoa demonstrated the occurrence of error thresholds in genetic algorithms over a wide range of problems and landscapes. This very interesting work is published in a series of conference papers.

Optimal population size and mutation rate. The dream of a practitioner is to have a set of optimal parameters to solve his specific problem. On a few simple examples it is possible to find empirically the optimal rates, by sampling several runs of the genetic algorithm with different parameters. This was done by Ochoa in her PhD thesis. She concluded that there exists a relationship between the optimal mutation rate and the error threshold. An important contribution of Ochoa's work is to try to relate quantitatively the optimal mutation rate with the error threshold. Cervantes and Stephens investigated further this idea [4]. One of the most interesting and inspiring work on the theory of genetic algorithms I have read over the last years is the series of papers by van Nimwegen, Crutchfield and Mitchell [18, 19, 15, 16, 17]. In these papers, the authors perform a theoretical and experimental study of a genetic algorithm on a specific class of fitness functions. Their analysis rely on techniques from mathematical population genetics, molecular evolution theory and statistical physics. Among the fundamental ingredients guiding the analysis are the quasispecies model, the error threshold and metastability. In the last work of the series [17], van Nimwegen and Crutchfield describe an entire search effort surface and they introduce a generalized error threshold in the space of the population size and mutation probability delimiting a set of parameters where the genetic algorithm proceeds efficiently.

Phase transitions. A basic goal of statistical mechanics is to understand the collective behavior of particles governed by simple microscopic rules. Typically, the particles are driven by two antagonistic effects: entropy and energy. Interesting models present a phenomenon of phase transition: there exists a critical point or a critical curve in the parameter space separating a region where energy effects dominate from a region where entropy effects dominate. The system is most interesting at criticality, where both forces compete equally. Perhaps the most studied model is the Ising model. There has been remarkable progress recently in the rigorous understanding of the critical Ising and percolation models in two dimensions [20]. The error threshold is in fact a particular type of phase transition [7]. The antagonistic forces in presence are mutation and selection and this threshold separates a regime where selection dominates from a regime where mutation dominates. In a genetic algorithm, the crossover operator complicates the dynamics and either it shifts the critical points or it creates new ones. This phenomenon has been observed independently by Rogers, Prügel–Bennett and Jennings [14] and by Nilsson Jacobi and Nordahl [11].

Efficient search. An efficient search procedure should realize a delicate balance between an exploration mechanism and a selection mechanism. This general idea is present in numerous works dealing with random optimization. I have believed for several years that a search procedure works best if it is close to a critical point, which realizes an optimal balance between the exploration and the selection mechanisms. This view is supported by the general knowledge coming from statistical mechanics, and it is also expressed in several previous works [13, 17, 14]. Unfortunately, phase transitions and critical points are sharply defined only for infinite systems. Moreover the computation of the critical points is very hard and complex, it can be achieved only for specific models, like Eigen's model or the two dimensional Ising model, and it requires great mathematical skills. For the three dimensional Ising model, the critical point can only be estimated numerically. Hence the task of computing the optimal parameters of a genetic algorithm on a specific problem is a formidable one, clearly much harder than solving the problem itself.

Self–organized criticality. My next hope is to try to adapt the parameters of the genetic algorithm during the search in order to reach a critical regime. Systems which are driven naturally towards a critical state have attracted a lot of interest since the seminal work of Bak, Tang and Wiesenfeld [1]. These systems are said to exhibit self–organized criticality. Several researchers have tried to incorporate such mechanisms to design optimization procedures. An interesting example is the extremal optimization [2]. In his PhD thesis [21], Whitacre investigates the occurrence of self–organized criticality in evolutionary algorithms. Krink, Thomsen and

Rickers used successfully mechanisms inspired by self–organized criticality to control evolutionary algorithms [12]. My aim here is to propose a very simple heuristic to achieve a critical control of a genetic algorithm.

Critical control of the mutation. When running a genetic algorithm, we are not looking for the optimal mutation probability, rather we look for a control of the mutation probability which allows to explore efficiently the space. Several researchers have already worked on this idea and proposed different possible schemes to adapt the parameters of the genetic algorithm. A review is presented in [5]. Here I propose an adaptive procedure which receives a simple feedback from the search. The conjectural picture I have in mind is the following. The genetic algorithm running on a fitness landscape is a finite population model, approximating an infinite population model. This infinite model presents several phase transitions, depending on the geometry of the fitness landscape. In a way, there is a phase transition associated to each local maximum. The parameters of a genetic algorithm should be adjusted in order to be close to the phase transition corresponding to its current position. When the algorithm escapes from a local maximum and finds a better point, the parameters should be completely readjusted from scratch. In practice, we need a simple criterion to decide whether the mutation parameter is above or beneath the local critical value. Considering Eigen's model of quasispecies and keeping in mind that we are dealing with a finite population, I propose the following simple criterion. If the best fitness observed in the population decreases, then the mutation probability is above the critical value. If the best fitness observed in the population is constant, then the mutation probability is below the critical value. These speculations lead to the algorithm presented on page 7. As for the procedure to control the mutation, there are plenty of choices. A very simple possibility is to use a dichotomy procedure. We use two extremal values α and β which bound the abstract critical mutation probability and we do as follows. To initialize the mutation control, we set $\alpha = 0$ and β to a reasonable upper bound on the critical mutation probability. In the case of a genetic algorithm working with binary words of length n, the initial value of β should be of order $(\ln c)/n$, where c is larger than the ratio between the maximum and the minimum of the fitness function. One should not take β too large, otherwise the mutations will destroy all the relevant information in the population, without hope of recovering the interesting points during the subsequent steps. The initial probability mutation is then set to $p_m = (\alpha + \beta)/2$. To increase the mutation probability, we set successively $\alpha = p_m$ and $p_m = (\alpha + \beta)/2$. To decrease the mutation probability, we set successively $\beta = p_m$ and $p_m = (\alpha + \beta)/2$. When the algorithm decreases the mutation, it is because the best fitness has decreased, and the hope is to recover the best points of the previous generations with reversed mutations. This is likely to happen only if the mutation rate is not too high, hence it seems important to be able to adjust adequately the initial upper bound β. Another possibility is to use elitism, which leads to the variant of the previous algorithm presented on page 8. The idea is natural: whenever the best fitness has decreased, we reintroduce by force the lost best solution, and we decrease the mutation probability. I think that this algorithm is more interesting than a standard search procedure incorporating elitism, because it is likely to escape from a local maximum much quicker. When exploring the landscape around a local maximum, it is reasonable to explore incrementally the neighborhood, starting with the points which are close in a mutational sense to the current best solution and proceeding then with further and further points. The progressive increase of the mutation rate implements this strategy to some extent. Another good reason to increase the mutation rate is to avoid premature convergence of the population, a problem that has been observed since the early days of genetic algorithms. The algorithm I propose tries to run with the largest reasonable mutation probability, which maintains the widest diversity in the population without losing the best fit individual.

Critical control of the population size. A very interesting conclusion of [17] is the existence of a critical population size below which it is practically impossible to reach the global optimum. A similar conclusion was obtained in the simpler framework of generalized simulated annealing [3]: within a specific asymptotic regime of low mutations and high selection pressure, the convergence to the global maximum could be guaranteed only above a critical population size. These works support the idea of the existence of a critical population size. From now on, let us suppose that this idea is correct. If we adhere to the belief that a search procedure is more efficient when it is running close to a critical state, we should also try to adjust the population size close to the critical one, in the same way as we did with the mutation. It is of course clear that we want a population size larger than the critical one, because an algorithm with too small a population will not reach the global maximum. One may however wonder why it would be a problem to use a population size much larger than the critical one. I imagine that it would lead to a waste of computational resources. Indeed it does not make sense to use a genetic algorithm with a large population size to find the maximum of a concave function, because a population of a few individuals will certainly suffice and do the job much faster. Another reason is that it seems best to have only a very small fraction of the population sitting on the best current points, in order to use most of the computational resources to explore the vicinity of these points. This is also a very interesting conclusion of [17]. With this in mind, I propose the more elaborate algorithm presented page 9.

What to do next. I hope that the critical mutation control of the genetic algorithm will lead to real practical improvement. The benefit of controlling the population size is more conjectural, but I believe it is a very interesting direction to try: if the genetic algorithm is still stuck at a local maximum after convergence of the mutation probability, then one should increase the population size in order to escape and find a better solution. A further possibility is to perform a simultaneous critical control of the mutation probability and the population size during the run of the algorithm. To do that will require slightly more complicated methods, because two parameters have to be optimized simultaneously, but this seems really to be a promising direction to explore. From an algorithmic point of view, it seems easier to vary first the mutation probability and then the population size. Another issue is the use of elitism. Elitism is a straightforward mechanism guaranteeing the asymptotic convergence. Instead of using an elitist algorithm, one can try to exert an adequate control on the parameters of the algorithm in order to enforce elitism.

Summary of the heuristic. The heuristic I propose is quite simple. The parameters of a genetic algorithm should be set close to a conjectured critical line describing the phase transition associated with its current localization in the fitness landscape. In order to adjust the parameters, we observe the behavior of the best fit individual from one generation to another and we proceed as follows:

- If the best fitness decreases strictly, then the genetic algorithm is operating in the regime where mutation dominates. Thus the mutation probability should be decreased or the population size should be increased.
- If the best fitness is constant, then the genetic algorithm is operating in the regime where selection dominates. Thus the mutation probability should be increased or the population size should be decreased.
- If the best fitness increases strictly, then the genetic algorithm has successfully escaped from a local maxima, the mutation control should be reinitialized and the population size should be reset to two.

The idea of adapting the parameters of the genetic algorithm is not a new one and it has already been seriously investigated by many researchers [5]. However the philosophy behind the control described above is quite different from what I have seen in the literature. For example, a common approach is to reward operators which lead to good solutions, with the hope that they will create even better solutions. My suggestion comes from a different principle, which is in fact quite opposite. The crucial idea is that a delicate interaction between exploration and selection is the key to an efficient search. In order to create the critical equilibrium, I suggest to favor the mechanism which currently does not produce good results!

Critical mutation control

```
Initialization
  • Initialize the population
  • Find the best fit individual $\widehat{y}$
  • Initialize the mutation control
Main loop
  • Set $\widehat{x} = \widehat{y}$
  • Apply selection
  • Apply mutation
  • Apply crossover
  • Find the best fit individual $\widehat{y}$
  If fitness($\widehat{x}$) $=$ fitness($\widehat{y}$) then
      Increase mutation probability
      Goto Main loop
  If fitness($\widehat{x}$) $>$ fitness($\widehat{y}$) then
      Decrease mutation probability
      Goto Main loop
  If fitness($\widehat{x}$) $<$ fitness($\widehat{y}$) then
      Reinitialize the mutation control
      Goto Main loop
```

Dichotomy procedure to change mutation

Initialization: $\alpha = 0$, $\beta = (\ln c)/n,\ p_m = (\alpha + \beta)/2$

Mutation increase: $\alpha = p_m,\ p_m = (\alpha + \beta)/2$

Mutation decrease: $\beta = p_m,\ p_m = (\alpha + \beta)/2$

Critical mutation control with elitism

```
Initialization
  • Initialize the population
  • Find the best fit individual $\widehat{y}$
  • Initialize the mutation control
Main loop
  • Set $\widehat{x} = \widehat{y}$
  • Apply selection
  • Apply mutation
  • Apply crossover
  • Find the best fit individual $\widehat{y}$
  If fitness($\widehat{x}$) = fitness($\widehat{y}$) then
      Increase mutation probability
      Goto Main loop
  If fitness($\widehat{x}$) > fitness($\widehat{y}$) then
      Decrease mutation probability
      Reintroduce $\widehat{x}$ in the population
      Set $\widehat{y} = \widehat{x}$
      Goto Main loop
  If fitness($\widehat{x}$) < fitness($\widehat{y}$) then
      Reinitialize the mutation control
      Goto Main loop
```

Critical mutation and size control with elitism

```
Initialization
    • Set the population size to two
    • Initialize the population
    • Find the best fit individual $\widehat{y}$
    • Initialize the mutation control
Main loop
    • Set $\widehat{x} = \widehat{y}$
    • Apply selection
    • Apply mutation
    • Apply crossover
    • Find the best fit individual $\widehat{y}$
    If fitness($\widehat{x}$) $<$ fitness($\widehat{y}$) then
        Reinitialize the mutation control
        Set the population size to two
        Initialize the population with two copies of $\widehat{y}$
        Goto Main loop
    If the mutation probability has converged then
        Increase the population size
        Reinitialize the mutation control
        Goto Main loop
    If fitness($\widehat{x}$) $=$ fitness($\widehat{y}$) then
        Increase mutation probability
        Goto Main loop
    If fitness($\widehat{x}$) $>$ fitness($\widehat{y}$) then
        Decrease mutation probability
        Reintroduce $\widehat{x}$ in the population
        Set $\widehat{y} = \widehat{x}$
        Goto Main loop
```

Acknowledgements: This work was completed during a visit to the mathematics department of the University of Padova. I warmly thank Carlo Mariconda and Paolo Dai Pra for their hospitality.

References

[1] Per Bak, Chao Tang, and Kurt Wiesenfeld. Self-organized criticality: An explanation of 1/f noise. *Phys. Rev. Lett.*, 59:381–384, 1987.

[2] Stefan Boettcher and Allon G. Percus. Extremal optimization: Methods derived from co-evolution. In *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 1, pages 825–832, Orlando, Florida, USA, 13-17 July 1999.

[3] Raphaël Cerf. Asymptotic convergence of genetic algorithms. *Adv. in Appl. Probab.*, 30(2):521–550, 1998.

[4] Jorge Cervantes and Christopher Rhodes Stephens. "Optimal" mutation rates for genetic search. In *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, volume 2, pages 1313–1320, Seattle, Washington, USA, 8-12 July 2006.

[5] Agoston Endre Eiben, Robert Hinterding, and Zbigniew Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141, July 1999.

[6] Manfred Eigen. Self-organization of matter and the evolution of biological macromolecules. *Naturwissenschaften*, 58(10):465–523, 1971.

[7] Manfred Eigen. Natural selection: a phase transition? *Biophysical Chemistry*, 85(2–3):101–123, 2000.

[8] Manfred Eigen, John McCaskill, and Peter Schuster. The molecular quasi-species. *Advances in Chemical Physics*, 75:149–263, 1989.

[9] David Goldberg. *Genetic algorithms in search, optimization and machine learning.* Addison–Wesley, 1989.

[10] John H. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, Mich., 1975.

[11] Martin Nilsson Jacobi and Mats Nordahl. Quasispecies and recombination. *Theoretical Population Biology*, 70(4):479–485, 2006.

[12] Thiemo Krink, Peter Rickers, and René Thomsen. Applying self-organised criticality to evolutionary algorithms. In *Parallel Problem Solving from Nature – PPSN VI*, volume 1, pages 375–384. Springer, 2000.

[13] Gabriela Ochoa. *Error Thresholds and Optimal Mutation Rates in Genetic Algorithms*. PhD thesis, The University of Sussex, Brighton, 2001.

[14] Alex Rogers, Adam Prügel-Bennett, and Nicholas R. Jennings. Phase transitions and symmetry breaking in genetic algorithms with crossover. *Theoret. Comput. Sci.*, 358(1):121–141, 2006.

[15] Erik van Nimwegen and James P. Crutchfield. Metastable evolutionary dynamics: Crossing fitness barriers or escaping via neutral paths? *Bulletin of Mathematical Biology*, 62(5):799–848, 2000.

[16] Erik van Nimwegen and James P. Crutchfield. Optimizing epochal evolutionary search: Population-size independent theory. *Computer Methods in Applied Mechanics and Engineering*, 186(2–4):799–848, 2000.

[17] Erik van Nimwegen and James P. Crutchfield. Optimizing epochal evolutionary search: Population-size dependent theory. *Machine Learning Journal*, 45:77–114, 2001.

[18] Erik van Nimwegen, James P. Crutchfield, and Melanie Mitchell. Finite populations induce metastability in evolutionary search. *Phys. Lett. A*, 229(3):144–150, 1997.

[19] Erik van Nimwegen, James P. Crutchfield, and Melanie Mitchell. Statistical dynamics of the royal road genetic algorithm. *Theoret. Comput. Sci.*, 229(1-2):41–102, 1999.

[20] Wendelin Werner. *Percolation et modèle d'Ising*, volume 16 of *Cours Spécialisés*. Société Mathématique de France, Paris, 2009.

[21] James M. Whitacre. *Adaptation and Self-Organization in Evolutionary Algorithms*. PhD thesis, School of Chemical Sciences and Engineering, The University of New South Wales, 2007.

Raphaël Cerf
Université Paris Sud
Mathématique, Bâtiment 425
91405 Orsay Cedex–France
rcerf@math.u-psud.fr