Parameter Tuning: Simulated Annealing for Function Optimization

Daniel Cuéllar, Elva Díaz and Eunice Ponce de León

Universidad Autónoma de Aguascalientes (UAA), Computer Sciences Department, Basic Sciences Center, 940 Universidad Ave., Ciudad Universitaria, Aguascalientes, Aguascalientes, México
cuellar_garrido@hotmail.com, elvitad@yahoo.com, eponce@correo.uaa.mx
(Paper received on August 30, 2010, accepted on October 20, 2010)

Abstract. This work proposes an experimental strategy for calibrating algorithms. A function maximization problem is used to test the strategy. Simulated annealing is selected to this research because it is a heuristic method which requires more initial parameters for its efficient operation than other heuristics. In many cases the success or failure of the algorithm depends greatly on doing this initialization task correctly and despite this difficulty, simulated annealing is a powerful tool, which has provided good results in solving problems of optimization in bioinformatics, medicine, engineering, physics, etc. The calibrating experiment obtains the best combination of parameters, reducing the search space only to the most promising interval, where the optimum is located. Each problem presents special characteristics and needs its proper tuning in concordance with the "No Free Lunch" theorem. The simulating annealing has as training part, the automatic tuning of parameters.

Keywords: Simulated annealing, Function optimization, No free lunch theorem, Algorithms tuning, Parameters.

1 Introduction

The "no free lunch" theorem (NFLT) explicitly demonstrates that all algorithms have the same averaged performance over all classes of problems. The only way that this can happens is that the gaining of performance in one class is paid with the loss of performance in all other classes [9]. If we want to use one algorithm to solve a particular problem, the NFLT gives a foundation to analyze and to select the best algorithm performance for this problem. If it is possible to select the best combination of the algorithm parameters, this particular set of parameter values is named the calibration values, and the algorithm with calibrated values receives the name of calibrated algorithm. The problem of tuning a metaheuristic can be profitably formalized and solved as a machine learning problem [7], so a tuning routine or strategy can be implemented as a training step in the algorithm; for this research, the optimization of functions has been selected because the results can be easily drown, and the simulating annealing algorithm has been selected because has many parameters which makes it difficult to calibrate.

©G. Arroyo-Figueroa (Ed)
Special Issue in Advances in Artificial Intelligence and Applications
Research in Computing Science 51, 2010, pp. 38-45

2 The Simulated Annealing Algorithm

The simulated annealing is an optimization technique that simulates the thermodynamic phenomenon of annealing. It uses concepts directly brought from physics, which needs to be adapted to the physical model of annealing to the case at hand. The simple simulated annealing algorithm is presented below:

Algorithm 1. Pseudocode for the simple simulated annealing algorithm

1. Material selection to anneal with (objective function definition to solve).
2. Parameter initialization (initial temperature, rate of cooling, etc).
3. Modification and evaluation of the actual system energetic state (Acceptance criteria and Metropolis routine).
4. If **thermal equilibrium*** doesn't exists return to step four.
5. If the system isn't **frozen**** decrease temperature based on the cooling schedule and return to step four.
6. End.

* **Thermal Equilibrium:** In simulated annealing, the thermal equilibrium is achieved when enough number of consecutive results is accepted during a certain temperature and new changes produce insignificant results in the optimization function.

** **System Frozen:** In simulated annealing, the system is frozen when no more results are accepted after several temperature decreases during the cooling schedule, or when a certain stop criteria is meet, this circumstances occur generally when the temperature reach or approach to zero.

The principal characteristic of the simulated annealing consists in the implementation of a routine known as Metropolis algorithm that employs a "Boltzmann probability" to prevent the system from falling into a local optimum [4]. While it is easy to run a simulated annealing program, is hard to make it work well [1].

3 Function Selection

The problem we want to solve is to find the global maximum value of an arbitrary mathematical function. The proposed function called M2, is multiplied for a coefficient of 100 and contains several local maximum and a single global optimum. These characteristics make it interesting to test the effectiveness of the algorithm and the tuning strategy. The domain of this function is between -2 to 2 and we can observe a high fluctuation in the values in this range. Its equation and graph are shown next:

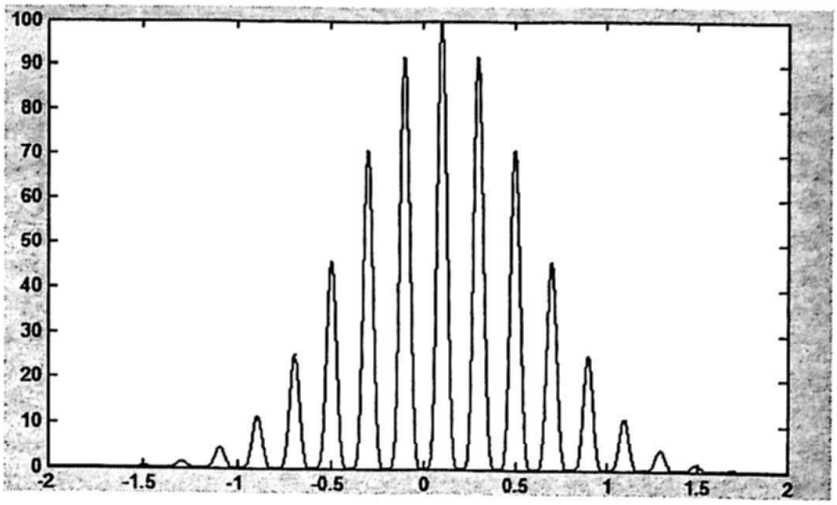

Fig. 1. *M2 Evaluated from -2 to 2 and multiplied for a coefficient of 100*

4 The Tuning Problem

The simulated annealing algorithm is very sensitive to any value assigned to its parameters and the relationship between these, determining the success or failure of it. We can say that this is a crucial and difficult stage to achieve. The first and most natural approach for this task is to perform this initialization manually and get the right combination to achieve a result in an empirical way, but sometimes this can be a hard work. Another problem lies in the resolution and selection range of the parameters, so it is necessary to delimit the potential range of the general parameters of the algorithm (study of the sensitivity of the function) [2].

5 Experimental Approach

5.1 Tuning Experiment

For the computer, this is a black box problem, without information a priori; the only way to know when the tuning is better is when it finds those combinations of parameters that yield best results with each new implementation. For this, we have to try all the possible combinations of parameter values, which is a combinatorial problem of exponential nature, however, it is possible to restrain the number of combinations within a calculable amount and assess their tendency to close up the gap until the range of effect for the desirable result. The results of these combinations should reflect variability in the results, whether these are good or bad results, so that you can see a trend toward what parameters are affecting the problem and what not.

5.2 Experimental Design

In this experiment, three values were combined for each of the five parameters as shown in table 1, this for reducing the number of combinations and thus avoid combinatorial explosion while having a pivot value and two more to tell us whether there is any bias towards one of its sides:

Table 1. Initial value selection for the parameter tuning experiment

Value	Low	Medium	High
Temperature	10	20	50
Equilibrium	2.5	5	10
Cooling	.3	.6	.9
Iterations	10	50	100
Freezing	.000001	.001	1

Then we evaluate all possible combinations of parameters with a total of $(n)^p$ combinations, where p is the number of parameters and n the number of values for each, in other words $(3)^5 = 243$ and in turn make 10 replicas of each combination for a total of $10 * (3)^5 = 2430$ results (simulated annealing results vary randomly because of its stochastic nature so is appropriate to make replicas for each possible combination.) The design of the experiment is run within a nested structure of FORS:

Algorithm 2. General data extraction procedure for the tuning experimental design

```
FOR (i = 1; i < n; i++)
{
        if (i == 1) {Parameter1 = Value^P1_1;}
        if (i == 2) {Parameter1 = Value^P1_2;}
        if (i == n) {Parameter1 = Value^P1n;}
FOR (j = 1; j < n; j++)
{
        if (j == 1) {Parameter2 = Value^P2_1;}
        if (j == 2) {Parameter2 = Value^P2_2;}
        if (j == n) {Parameter2 = Value^P2n;}
        FOR (X = 1; X < Replics; X++)
        {
                Heuristic (Annealing) Program
                Save Actual Parameters Combination in a File (TXT, XLS, etc.)
                Save Results for Parameters Combination in a File (TXT, XLS, etc.)
        }
}
}
```

6 Results and Discussion

Figure 2 shows the results for each of the 2430 parameter combinations, it is interesting to see how the program has some combinations of parameters that yield bad results which allow us to see that in fact there is variability in the results, which means that the parameter combinations affect the results, however, by the nature of simulated annealing these are concentrated in the local maximums and the optimal of the function. In addition, the data were subjected to an analysis of a linear stepwise regression model; the analysis of the problem was performed in SPSS and produced the results shown in the table 2. Regression analysis was used to detect the order of importance of each parameter; the most important parameter that most directly affects the outcome under the analysis was the rate of cooling followed by freezing, the initial temperature and the equilibrium parameter. In turn, the analysis also gives us results about which parameters are insignificant as is the case of iterations, which even ran in the results.

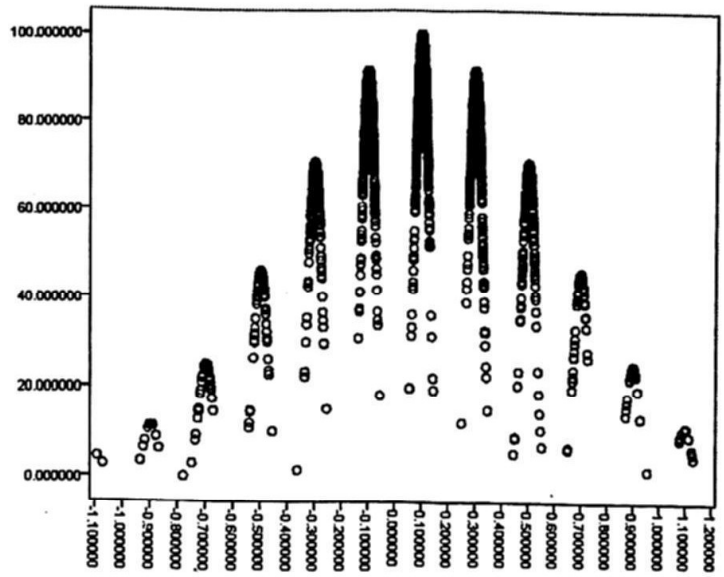

Fig. 2. Experimental results for the parameter combinations showed on table 1

Based on this study, we conclude that all are significant except for the number of iterations because it does not affect the variance. This is not necessarily true; the parameter may be that not really significant or the range used might not include the values that affects the results. To find out what is our case, is pertinent to repeat the experiment extending the range, if even changing the range of action we did not find any change, then the variable is insignificant and can be neglected in the algorithm, can be fixed in a comfortably value or can be eliminated if this one doesn't affect the algorithm behavior. Now that we know what parameters are significant, we need to know the range where they exercise action. For this, we took the 20 best results and the combinations of parameters where they were found and based on these results we made the table 3:

Table 2. Stepwise regression test results for the analysis of parameter significance

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	59.387	.688		86.307	.000
	Cooling	39.942	1.062	.403	37.619	.000
2	(Constant)	65.257	.661		98.750	.000
	Cooling	39.942	.985	.403	40.550	.000
	Freezing	-17.595	.512	-.342	-34.359	.000
3	(Constant)	70.311	.752		93.467	.000
	Cooling	39.942	.973	.403	41.052	.000
	Freezing	-17.595	.506	-.342	-34.785	.000
	Temperature	-.190	.014	-.133	-13.516	.000
4	(Constant)	67.166	.872		77.066	.000
	Cooling	39.942	.970	.403	41.190	.000
	Freezing	-17.595	.504	-.342	-34.902	.000
	Temperature	-.190	.014	-.133	-13.562	.000
	Equilibrium	.539	.076	.069	7.079	.000

a. Dependent Variable: MAXIMUM

Table 3. *Frequency table for individual parameter values and its impact in the 20 best results*

Parameters	*Values*	*Frecuencies*
Equilibrium	2.5	2
	5	6
	10	12
Cooling	0.3	0
	0.6	0
	0.9	20
Temperature	10	6
	20	7
	50	7
Iterations	10	10
	50	5
	100	5
Freezing	0.000001	15
	0.001	5
	1	0

As is thought, smaller freezing temperatures and the cooling parameter at .9 are better; With respect to the initial temperature and the number of iterations, there is an evident pattern, when one increases the other decreases, which suggests a correlation between these two parameters. Based on these observations and selecting the best parameters, the parameters were set at the following values in table 4, and then used to generate other 2430 new results with a calibrated algorithm shown in the figure 3:

Table 4. Best parameter combination for the M2 maximization problem

	Temperature	Equilibrium	Cooling	Iterations	Freezing
Value	50	20	.9	50	.000001

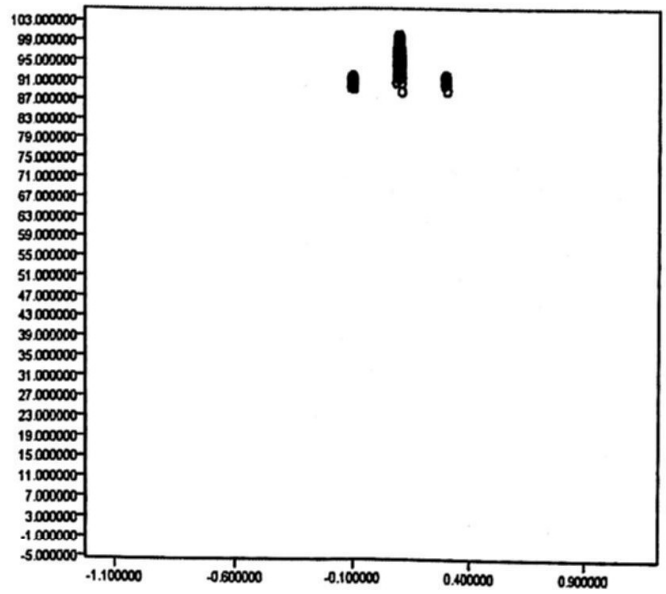

Fig. 3. *Experimental results for the parameter combination showed on table 4*

This is a very good convergence of the algorithm towards the global optimum; on the contrary, if we modify the specific parameters of the function, the tuning loses its effect, therefore requiring a new tuning, as shown in figure 4, with the same parameters but in a search space from -100 to 100 increasing the dispersion of results.

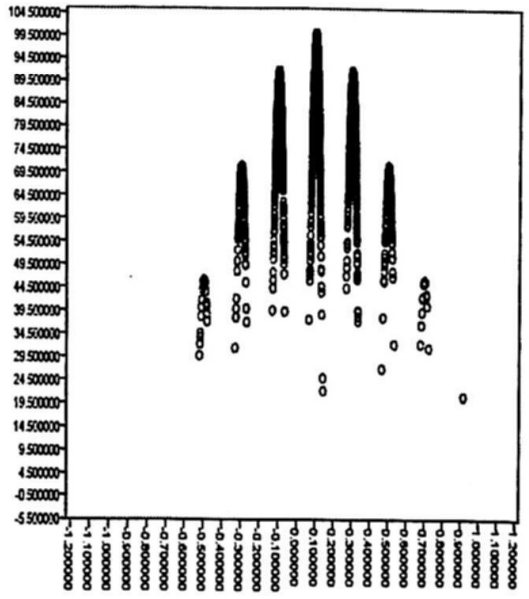

Fig. 4. *Experimental results for the previous combination with a search space from -100 to 100*

7 Conclusions and Further Work

As is shown, the calibrating experiment obtains the best combination of parameters, reducing the search space only to the most promising interval, where the optimum is located. The most important parameters are cooling, freezing and initial temperature, constituting the cooling scheduling. Each problem presents special characteristics and needs its proper tuning. This is according to the "No Free Lunch" theorem of Wolpert and Macready [9]. In the future, we will be implementing this proposed strategy for solving deceptive problems, if the tuning strategy proves to be efficient we will be making a general tuning program such as proposed in [2].

References

1. A. Dowsland, Kathryn and Berlamino Adenso Díaz. *Diseño de heurísticas y fundamentos del recocido simulado*. Revista Iberoamericana de inteligencia artificial 19 (2003): 93-102.
2. Adenso Díaz, Berlamino and Manuel Laguna. *Fine-tuning of algorithms using fractional experimental designs and local search*. Operations Research 54.1 (2006): 99-114.
3. Chelouah, R. and P. Siarry. *Tabu Search Applied to Global Optimization.* European Journal of Operational Research 123 (2000): 256-270.
4. Dréo, J., et. al. *Metaheuristics for hard optimization: Methods and case studies*. Springer, 2006.
5. Hajek, Bruce. *Cooling schedules for optimal annealing*. Mathematics of Operations Research 13.2 (1988): 311-329.
6. Kirkpatrick, S., C. D. Gelatt and M. P. Vecchi. *Optimization by simulated annealing*. Science 4598.220 (1983): 671-680.
7. Mauro, Birattari. *The problem of tuning metaheuristics as seen from a machine learning perspective.* Universite Libre de Bruxelles, 20 December 2004.
8. Siarry, P. and G. Dreyfus. *La méthode du recuit simulé: théorie et applications*. (1989).
9. Wolpert, D.H. and W.G. Macready. *No Free Lunch Theorems for Optimization*. IEEE Transactions on Evolutionary Computation 67.1 (1997).
10. Wolpert, D.H. and W.G. Macready. *No Free Lunch Theorems for Search.* Technical Report. Santa Fe Institute, 1995.