PROBLEM AND ALGORITHM FINE-TUNING
A Case of Study using Bridge Club and Simulated Annealing

Nelson Rangel-Valdez, José Torres-Jiménez, Josue Bracho-Ríos and Pedro Quiz-Ramos
Laboratory of Information Technologies, CINVESTAV, Cd. Victoria, Tamaulipas, Mexico

Keywords: Parameter tuning, Covering arrays, Bridge club, Simulated annealing.

Abstract: Sometimes, it is difficult to cope with a good set of values for the parameters of an algorithm that solves an specific optimization problem. This work presents a methodology for fine-tuning the parameters of a Simulated Annealing (SA) algorithm solving the Bridge Club (BC) problem. The methodology uses Covering Arrays as a tool that evaluates a set of values for the parameters of the SA so that it achieves its best performance when solving BC. The results in the experiments performed show that, using this methodology, the SA reached the optimal solution of the BC problem with a relatively small number of evaluations, in comparison with other strategies that solves BC.

1 INTRODUCTION

The Bridge Club (BC) problem has been considered as a case of study in discrete optimization. This problem, in appearance simple, arises as a complex problem when solved by an exact approach. The Bridge Club problem and its optimal solution are defined in (Elenbogen and Maxim, 1992).

BC has been solved using several approximated approaches: greedy algorithm (Elenbogen and Maxim, 1992), partial branch & bound (Elenbogen and Maxim, 1992), steepest descent (Elenbogen and Maxim, 1992), annealed search (Elenbogen and Maxim, 1992), Kreher, et. al. (Kreher et al., 1996), Genetic Algorithm (Simpson, 1997), tabu search (Morales, 1997). Even though some of these techniques are sensitive to the values of its parameters, few is known about how good such values are (Simpson, 1997).

Table 1 compares the reported algorithms that solve the BC problem. First column shows the name of the methods. The second and third columns show the best solution and time spent by each algorithm, respectively. The last column shows the number of evaluations of the cost function used by each algorithm to reach their best reported value.

Table 1: Comparison between the different algorithms that solve BC.

Method	Cost	Time	Evaluations
Greedy Algorithm	22	3 hrs.	40,397
Partial Branch & Bound	20	8 hrs.	100,000
Steepest Descent	14	2 hrs.	75,000
Annealed Search	14	1 hr.	50,000
Kreher, et. al.	12	-	1,000,000
Genetic Algorithm	12	-	375,000
Tabu Search	12	-	3,456 - 153,600

The best values for the parameters of an algorithm are estimated according with the problem that is solved. Several strategies to choose these values can be found in the literature. Generally, the best values for an algorithm are determined through an experimental design. Sometimes, these values are taken from the literature (Laarhoven and Aarts, 1987). Some other are just chosen without validating its effectiveness. More recent approaches try new techniques based on factorial designs (Díaz and Laguna, 2006).

In the search for the best solution of a problem we try to find the best values for its parameters and the parameters of the algorithm that solves it. The parameters of a problem are the ones that are defined according with its definition. Among these parameters are the neighborhood function, the evaluation function, initial solution and internal representation. The parameters of an algorithm are those that do not need the definition of the problem to be specified. Some examples of algorithm parameters are temperature and population. Most of the time, the selection of the values for the parameters of the problem is independent from the parameters of the algorithm. This way of choosing values represents an area of opportunity in the sense that, combining both kind of parameters we can choose their values in one experimental design and we can exploit the relationship that can exist between the parameters of the algorithm and the parameters of the problem.

In order to show the effectiveness of a methodology for fine-tuning parameters of both the algorithm and the problem, this paper proposes a methodology based on Covering Arrays (Lopez-Escogido et al., 2008). The proposed methodology fine-tunes the parameters of a solution for the Bridge Club problem via Simulated Annealing.

This document is organized as follows. Section 2 shows the details about the implementation of the Simulated Annealing that solves the Bridge Club problem. Section 3 describes the methodology used to tune the parameters of the Simulated Annealing as well as the parameters of the problem. Section 4 shows the results of applying the tuning process in the solution of the Bridge Club problem through the Simulated Annealing algorithm. Finally, section 5 presents the conclusions derived from this research work.

2 SIMULATED ANNEALING

The Simulated Annealing Algorithm(SA) first introduced by S.Kirkpatrick (Kirkpatrick, 1983) is inspired in statistical mechanics. The parameters of the SA are the cooling schedule and the stop criterion. The parameters of the BC problem are the internal representation of the solution, the initial solution, the neighborhood function and the evaluation function.

The cooling schedule refers to an initial temperature T_0, a final temperature T_f, a cooling factor α, the frozen parameter Γ (number of consecutive Markov chains without improvement) and the length of a Markov chain L. In addition, in order to determine the moment in which the SA ends, a number of evaluations I is used. Once that the evaluation function is performed I times, the SA exits and reports the best solution achieved.

The solution representations for the BC problem are based in the matrix representations proposed in (Simpson, 1997) (representation $\mathcal{R}_b$), in (Elenbogen and Maxim, 1992) (representation $\mathcal{R}_a$) and a new representation (representation $\mathcal{R}_c$ that fixes the group A in the initial solution in the first column of the representation proposed by (Elenbogen and Maxim, 1992).

The initial solution for all the representations is randomly generated. The values of the matrix $\mathcal{M}$ with representation $\mathcal{R}_a$ will be randomly chosen between $\{1,2,...,12\}$ or between $\{A,B,C\}$ otherwise.

In this approach we tested three evaluation functions. The first one $\Pi_1(\mathcal{M})$ shown in Equation 1 and defined in Elenbogen, et al. (Elenbogen and Maxim, 1992).

$$\Pi_1(\mathcal{M}) = \sum_{i=1}^{11} \sum_{j=i+1}^{12} (m_{ij} - 2)^2 \tag{1}$$

Two other evaluation functions are analyzed in the solution of the BC problem. The evaluation function $\Pi_2(\mathcal{M})$ rises to 4 instead of 2 the difference $(m_{ij} - 2)$. The evaluation function $\Pi_2(\mathcal{M})$ obtains the absolute difference instead of a power. Any evaluation function Π must be minimized by the SA algorithm to achieve the optimal solution.

This research work proposes three methods to build new solutions $\mathcal{M}'$ during the SA. The neighborhood function $\mathcal{N}_1$ creates $\mathcal{M}'$ by exchanging values between two elements $m_{i,j}, m_{i,k}$ of a given a solution matrix $\mathcal{M}$. The inversion strategy $\mathcal{N}_2$ can be defined as a set of exchanges; given two elements $m_{i,j}, m_{i,k} \in \mathcal{M}$, $j < k$, a new solution $\mathcal{M}'$ is constructed by applying $\mathcal{N}_1$ to $m_{i,\iota}, m_{i,\kappa}$, for all $j \le \iota, \kappa \le k$.

The neighborhood function based in rotations $\mathcal{N}_3$ creates $\mathcal{M}'$ using a solution $\mathcal{M}$ and two elements $m_{i,j}, m_{i,k} \in \mathcal{M}$, $j < k$. This function assigns the value of each element $m_{i,\iota}$, for every $j \le \iota < k$, to the element $m_{i,\iota+1}$. The element $m_{i,k}$ assigns its value to $m_{i,j}$.

Every neighborhood function $\mathcal{N}$ requires two elements $m_{i,j}, m_{i,k} \in \mathcal{M}$. The elements can be defined by choosing the values for i, j, k. The value i represents a row in $\mathcal{M}$ while the values j, k represent two different columns. This research work proposes three methods χ_a, χ_b, χ_c to choose the values of i, j, k to test the performance of the SA algorithm.

The method χ_a randomly chooses the values i, j, k every time that a neighborhood function $\mathcal{N}$ build a new solution $\mathcal{M}'$. The method χ_b is just like χ_a but it randomly selects the value of j once. After that, its value is increased by one at every call of a neighborhood function $\mathcal{N}$. The method χ_c, differs in which values i, j are randomly chosen once and after that their values are increased by one each time a neighborhood function $\mathcal{N}$ constructs a new solution $\mathcal{M}'$.

The SA stops the search for a solution for the BC problem when one of the following criteria are met: the system get frozen, the final temperature T_f has been reached, the maximum number of evaluations I has been exceeded or the optimal solution has been found, i.e., $\Pi(\mathcal{M}) = 12$, for a particular value of $\mathcal{M}$.

The next section describes the tuning process followed to find the best values for the parameters of the SA to solve the BC problem.

3 FINE-TUNING OF THE SA

Fine-tuning of parameters is the process of adjusting the parameters of an algorithm to solve a problem. Several works like (Barr et al., 1995), (Fink and S., 2002) have addressed the importance of performing a calibration of the parameters in heuristic and meta-heuristic approaches. Related works about methodologies that can be followed to perform a fine-tuning are shown in (Díaz and Laguna, 2006), (Fidanova et al., 2009), (Taguchi, 1994).

A *Covering Array*(CA) (Lopez-Escogido et al., 2008) is an array M of size $N \times k$ consisting of N vectors of length k with entries from 0, 1, ..., $v-1$ (v is the size of the alphabet) such that every one of the v^t possible vectors of size t occurs at least once in every possible selection of t elements from the vectors. The parameter t is referred to as the *covering strength*.

This process is based in a set of instances β of a problem $\mathcal{P}$ and in a Covering Array $CA(N;t,k,v)$ to study the effect of the interaction between parameters in the solution of $\mathcal{P}$ using an algorithm $\mathcal{A}$.

In the CA, the value k is the number of parameters of the algorithm $\mathcal{A}$ and the problem $\mathcal{P}$ object of the fine-tuning. The value of the alphabet v will correspond to the cardinality of the set of values for the parameters considered in the tuning process. The level of interaction between parameters is initialized to $t = 2$. During the process, the performance φ_t of the algorithm solving the problem $\mathcal{P}$ with a level of interaction t is measured. If the performance is improved, the values of the parameters of $\mathcal{P}$ and $\mathcal{A}$ are adjusted in order to find better solutions. If the performance φ_t is no longer improved, the interaction level t is increased by one only if $\varphi_t - \varphi_{t-1} > \varepsilon$. ε is a threshold selected so that the fine-tuning process halts in a suitable amount of time. Every time that the level of interaction t is increased then a new $CA(N;t,k,v)$ must be constructed.

The next section presents the experiments done to tune the values of the parameters of the Simulated Annealing to solve the BC problem using the methodology described in this section.

4 EXPERIMENTAL RESULTS

The methodology described in Section 3 was used to fine-tune the parameters of the Simulated Annealing in order to solve the BC problem.

The initial values selected for the parameters during the tuning process are shown in Table 2. The first column shows the parameter, the last 3 columns are the values selected for that parameter.

Table 2: Values for the parameters of the Simulated Annealing.

Type of Parameter	**Parameter**	1^{st} **value (0)**	2^{nd} **value (1)**	3^{rd} **value (2)**
Problem	$\mathcal{R}$	$\mathcal{R}_a$	$\mathcal{R}_b$	$\mathcal{R}_c$
Problem	χ	χ_a	χ_b	χ_c
Problem	Π	Π_1	Π_2	Π_3
Algorithm	T_0	1.0	0.5	0.25
Algorithm	α	0.99	0.90	0.85
Algorithm	T_f	0.001	0.000001	0.000000001
Algorithm	I	10000	100000	1000000
Algorithm	L	500	800	1000
Algorithm	Γ	5	10	15
Problem	$\mathcal{N}$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$

Table 3: Summary of the results obtained during the fine-tuning process using a $CA(14;2,10,3)$.

		Num.	**Avg.**	**Avg.**	**Avg.**	**Avg.**
Conf.	**Sol.**	**Opt.**	**Sol.**	**Time**	**Eval.**	**Opt. Eval.**
0002010112	14.00	0.00	37.29	0.00	20129.03	0.00
0101221200	12.00	6.00	32.26	0.90	97483.81	4666.67
0120112011	22.00	0.00	49.03	0.00	127741.88	0.00
0210002222	18.00	0.00	53.68	21.94	1331612.75	0.00
1001101122	14.00	0.00	38.84	0.00	91354.82	0.00
1011212100	12.00	2.00	32.77	0.58	47380.64	4800.00
1021020221	18.00	0.00	46.45	0.00	19354.84	0.00
1112000020	12.00	4.00	32.90	0.00	18322.58	4250.00
1210120210	12.00	2.00	37.16	0.00	18903.23	3000.00
2000021002	14.00	0.00	33.29	1.94	193548.45	0.00
2120200102	14.00	0.00	38.45	0.00	20129.03	0.00
2202122101	28.00	0.00	72.00	3.87	284903.31	0.00
2211201011	30.00	0.00	112.52	0.00	37741.95	0.00
2222211220	12.00	2.00	37.03	0.00	80903.23	45000.00

Table 3, in the first column, shows the first $CA(14;2,10,3)$ used for the tuning process. The alphabet for this CA is shown in the Table 2. Each digit from left to right correspond to a parameter shown in Table 2. The values 0, 1 and 2 correspond to values in the columns 2, 3 and 4, in Table 2, respectively.

The BC problem was solved using the SA and the CA shown in the first column of Table 3. The BC problem was solved 31 times by each configuration. Table 3 summarizes the results obtained during the fine-tuning process using a level of interaction of two $t = 2$. Column 2 shows the best solution achieved by each configuration. Column 3 shows the number of times that the optimal solution was reached. Column 4 presents the average cost of a solution using a specific configuration. Column 5 presents the average time measured in seconds that the SA spent in finding the optimal solution. Column 6 is the average number of times that the evaluation function was computed during the SA. Column 7 shows the average number of times that the SA needs to compute the evaluation function in order to find the optimal solution.

The results presented in Table 3 show a low quality in the solution achieved by the SA using the combination of the values described by $CA(14;2,10,3)$. The best configuration just achieved the optimum value 19% of the times. The solutions of the SA averaged a distance of 168% over the optimal solution, in the best case.

In order to improve the performance of the SA when solving the BC, the level of interaction between parameters t was increased. A new Covering Array $CA(54;3,10,3)$ was used. The results obtained were significantly improved. The best configuration achieved the optimum value in 54% of the cases. The solutions of the SA averaged a distance of 6% over the optimal solution. And with respect to the number of evaluations performed by the SA to find the optimal solution, it averages 25458 evaluations. In general, the time spent by the SA in finding a solution was around 0.81 seconds. The improvement achieved in the next level of interaction $t = 4$ did not surpass the threshold ε. The fine-tuning process halted at this point. The SA algorithm performed better with the configuration 2010002110 for the SA algorithm and BC parameters.

5 CONCLUSIONS

This document presents a methodology for fine-tuning the parameters of the SA and the BC problem. The methodology relies on the use of a CA to perform the parameters tuning.

The fine-tuning process adjusts the values of the set of parameters formed by the parameters of the SA and the parameters of the Bridge Club problem. An initial interaction of level two (a CA of strength $t = 2$) was initially proven, the best configuration achieved the optimal solution 19% of the times it was runned. In general, it averaged a 168% above the optimal solution. When increasing the level of interaction to three (a CA of strength $t = 3$), the best configuration found the optimal solution in 54% of the times and its average distance from the optimal was reduced to 6%, a dramatical change with respect to the interaction of level $t = 2$. No significative change in the performance of the SA were achieved with greater level of interaction.

The SA algorithm improves the reported results by other approaches. The methodology shows that the performance of an algorithm can be dramatically affected by the interaction level between the parameters involved. In our case of study, an interaction level of value three $t = 3$ was sufficient to find satisfactory values for the parameters of the SA in order to get a good performance in solving the BC problem. Finally, we can conclude that the Covering Arrays, as part of experimental designs, are valuable tools for fine-tune parameters associated with an algorithm and the problem being solved.

ACKNOWLEDGEMENTS

This research was partially funded by the following projects: CONACyT 58554-Cálculo de Covering Arrays, 51623-Fondo Mixto CONACyT y Gobierno del Estado de Tamaulipas.

REFERENCES

Barr, R., Golden, B., Kelly, J., Resende, M., W.R., and Jr., S. (1995). Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics*, 1(1):9–32.

Díaz, B. A. and Laguna, M. (2006). Fine-tuning of algorithms using fractional experimental designs and local search. *Operations Research*, 54(1):99–114.

Elenbogen, B. and Maxim, B. (1992). Scheduling a bridge club (a case study in discrete optimization). *Mathematics Magazine*, 65(1):18–26.

Fidanova, S., Alba, E., and Molina, G. (2009). Memetic simulated annealing for the gps surveying problem. pages 281–288.

Fink, A. and S., V. (2002). *Optimization Software Class Libraries*. Operations Research/Computer Science Interfaces Series. Boston, Massachusetts.

Kirkpatrick, S. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Kreher, D., Royle, G., and Wallis, W. (1996). A family of resolvable regular graph designs. *Discrete Math.*, 156(1-3):269–275.

Laarhoven, P. J. M. and Aarts, E. H. L., editors (1987). *Simulated annealing: theory and applications*. Kluwer Academic Publishers, Norwell, MA, USA.

Lopez-Escogido, D., Torres-Jimenez, J., Rodriguez-Tello, E., and Rangel-Valdez, N. (2008). Strength two covering arrays construction using a sat representation. In *LNCS 5317/2008*, pages 44–53.

Morales, L. (1997). Scheduling a bridge club by tabu search. *Mathematics Magazine*, 70(4):287–290.

Simpson, R. (1997). Scheduling a bridge club using a genethic algorithm. *Mathematics Magazine*, 70(4):281–286.

Taguchi, G. (1994). *Taguchi Methods: Design of Experiments*. American Supplier Institute.