Solving Congress Timetabling with Genetic Algorithms and Adaptive Penalty Weights

Daniel Ángel Huerta-Amante and Hugo Terashima-Marín

Center for Intelligent Systems, ITESM, Campus Monterrey
Ave. Eugenio Garza Sada 2501 Sur, C. P. 64849
Monterrey, Nuevo León, Mexico Tel. +52 (81) 8328 4379
dhuerta@csi.mty.itesm.mx, terashima@itesm.mx

Abstract When a Genetic Algorithm is used to tackle a constrained problem, it is necessary to set a penalty weight for each constraint type, so that, if the individual violates a given constraint it will be penalized accordingly. Traditionally, penalty weights remain static throughout the generations. This paper presents an approach to allow the adaptation of weights, where the penalty function takes feedback from the search process. Although, the idea is not new since other related approaches have been reported in the literature, the work presented here considers problems which contain several kinds of constraints. The method is successfully tested for the congress timetabling problem, a difficult problem and with many practical applications. Further analysis is presented to support the efficiency of the technique.

1 Introduction

Genetic Algorithms (GAs) [1, 2] are an optimization technique that has been successful for a variety of combinatorial and constrained optimization problems [3]. When a Genetic Algorithm is used to tackle a constrained problem in combination with a penalty function, it is necessary to set a penalty weight for each constraint type, so that, if the individual violates a given constraint it will be penalized accordingly. Traditionally, penalty weights remain static throughout the generations. This paper presents an approach to allow the adaptation of weights, where the penalty function takes feedback from the search process. Although, the idea is not new since other related approaches have been reported in the literature, the work presented here considers problems which contain several kinds of constraints. A complete survey related to adpatation strategies is presented and summarized by Michalewicz and Schmidt [4]. Hamida and Schoenauer [5] have proposed a technique which is based on the feasibility percentage. This idea was extended in this paper to solve a more constrained problem such as the congress timetabling [6, 7]. The constraints considered for our investigation are PRESET(x,y) (event x should be scheduled at exactly y hours), EXCLUDE(x,$*a$) (event x must not happen at certain times given in array a), ORDER(x,w) (an event x to be scheduled before some other event w), and TIME(t) (the sum of all scheduled events should not be greater than t).

The paper is organizaed as follows. Section 2 presents the technical details of the adaptation algorithms, the experimentation and results. Section 3 reports the conclusions and future work.

2 Adaptation algorithms and experimentation

The original method suggested by Hamida and Schoenauer [5] for adapting penalty weights modifies the penalty coefficients according to the proportion of feasible individuals in the population. Their strategy is used to combine feasible with infeasible individuals in order to explore the region around the feasible domain. The original technique adapted for our problem with several kinds of constraints and with adaptation of weights was called CGAA1. A more refined algorithm is called CGAA2. Several other parameters were introduced for both algorithms CGAA1 and CGAA2 such as the following: $fact$ - constant to increase or decrease the current penalty; $tTarget$ - target portion of feasible individuals in the population; *error* the difference between the actual percentage of feasibles and the target; $tTrain$ - parameter to adjust the penalty in terms of penalty units; *relaxation* - parameter to relax the constraint TIME, and penalize accordingly. $tMax$ and $tRel$ were two parameters linked to *relaxation*, which is in fact the parameter where the difference between CGAA1 and CGAA2 is focused. *Lapse* and $tEnd$ are parameters to setting the number of generations for adaptation, and the termination criterion, respectively. The original algorithm but with static penalty weights is labeled CGA. A permutation-based representation was used to solve the problem with GAs. Each chromosome represents a complete solution with n activities to be scheduled from left to right. The objective (penalty) funcion to evaluate each chromosome x is given by:

$$f(x) = \frac{1}{1 + P(x)} \tag{1}$$

where

$$P(x) = c_{time}p_{time} + c_{preset}p_{preset} + c_{order}p_{order} + c_{exclude}p_{exclude} \tag{2}$$

where p_{time}, p_{preset}, p_{order} and $p_{exclude}$ are the penalty weights for each constraint type, and c_{time}, c_{preset}, c_{order} and $c_{exclude}$ are the degrees of violation for each constraint.

To include adaptation for constraints in the congress timetabling problem studied in this article, the following steps for tuning the CGAA1 and CGAA2 algorithms were followed: (1) tune parameters for TIME and PRESET, (2) introduce adptability to each of those constraints, (3) introduce adaptability to the rest of constraints, and finally (4) tune the $tEnd$ paramater. For doing this, three randomly generated problems were designed for each congress duration (3, 4, and 5 days).

Results for all three algorithms tested are shown in Table 1. Each algorithm reports results for problems with duration of 3, 4, and 5 days. 17 different problems for each duration were generated, with 30 runs for each instance. It is

observed that for constraints TIME, PRESET, and EXCLUDE, in both algorithms CGAA1 and CGAA2, the adaptive process produces a decrease in the constraint violation. For instance, for constraint TIME the average number of minutes is 10.00 for CGA (static penalty weights) in congresses lasting 4 days, while for CGAA1 decreases to 9.67, and, for CGAA2 there is still a slight improvement decreasing to 6.06. The same kind of behavior is noticed for the other constraints. We found out that PRESET is the most difficult constraint for the adaptation process, eventhough there is also a decrease in the result for both algorithms CGAA1 and CGAA2 when compared to CGA. In the last column of the table FITNESS is reported. In order to have a point for comparison against the CGA, the best individual in the adaptation algorithms for a particular run was evaluated using the static weights. Table 2 shows results for a GA with static penalties, compared against the adpating models such as the CGAA1 and CGAA2 for 50 randomly generated problems with 30 runs for each of them. It is observed that the algorithm CGAA1 behaves better in general than the static model, and algorithm CGAA2 behaves slightly better than both. The degree of violation is shown for each kind of constraint (minimizing) and in the right column the average fitness (maximizing). These results also confirm those conclusions derived from the previous table.

	Days	TIME	PRESET	ORDER	EXCLUDE	FITNESS
CGA	3	3.43	68.83	0.03	7.33	0.726058
	4	10.00	102.70	0.03	5.80	0.654762
	5	17.10	91.60	0.40	6.40	0.618476
CGAA1	3	2.47	67.60	0.00	2.73	0.736826
	4	9.67	88.87	0.00	5.16	0.680868
	5	12.36	86.40	0.26	5.73	0.647106
CGAA2	3	1.63	67.60	0.00	4.60	0.760208
	4	6.06	88.86	0.00	5.63	0.671713
	5	12.93	91.66	0.00	3.10	0.639005

Table 1. Table comparing algorithms CGA, CGAA1, and CGAA2 for different durations of the congress.

3 Conclusions and Future Work

It has been demonstrated through an empirical study that adapting penalty weights in a GA for a constrained problem with multiple types of constraints performs better than a GA with static penalty weights. Despite this fact, adapting weights requires to set and tune various parameters. Future work is suggested

Algorithm	TIME	PRESET	ORDER	EXCLUDE	FITNESS
CGA	10.29	95.75	0.56	6.21	0.64570
CGAA1	8.47	93.65	0.37	1.92	0.66239
CGAA2	7.85	98.16	0.19	1.20	0.66287

Table 2. Table comparing performance of algorithms CGA, CGAA1, and CGAA2 for 50 randomly generated problems.

to investigate more about this trade-off. Several lessons were learned with this experimentation that could help to tackle similar problems in the future.

Acknowledgements

This research is supported by ITESM under the Research Chair CAT-010 and by CONACyT project 41515.

References

1. D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Publishing Company, Inc., 1989.
2. Melanie Mitchell. *An Introduction to Genetic Algorithms: Complex Adaptive Systems.* MIT Press, 1998.
3. R. Sarker, M. Mohammadian, and Y. Xin. *Evolutionary Optimization.* Kluwer International Series. 2002.
4. Zbigniew Michalewicz and Martin Schmidt. Evolutionary algorithms and constrained optimization. *Evolutionary Optimization*, pages 57–86, 2002. Sarker, R. and Mohammadian, M. and Xin, Y., editors.
5. S. Ben Hamida and Marc Schoenauer. An Adaptive Algorithm for Constrained Optimization Problems. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J.J. Merelo, and H.-P. Schwefel, editors, *Proceedings of 6th PPSN*, pages 529–538, Heidelberg, Germany, September 2000. Paris, Springer-Verlag. LNCS Vol. 1917.
6. H. Terashima. *Combinations of GAS and CSP strategies for solving examination timetabling problems.* PhD thesis, Instituto Tecnologico y de Estudios Superiores de Monterrey, October 1998.
7. A. Schaerf. A Survey of Automated Timetabling. *Artificial Intelligence Review*, 13:87–127, 1999.