A EVOLUTIONARY APPROACH TO SOLVE SET COVERING

Broderick Crawford, Carolina Lagos
Escuela de Ingeniería Informática, Pontificia Universidad Católica de Valparaíso, Valparaíso, Chile

Carlos Castro
Departamento de Informática, Universidad Técnica Federico Santa María, Valparaíso, Chile

Fernando Paredes
Escuela de Ingeniería Industrial, Universidad Diego Portales, Santiago, Chile

Keywords: Set Covering Problem, Cultural Algorithm, Genetic and Evolutionary Computation.

Abstract: In this paper we solve the classical Set Covering Problem comparing two evolutive techniques: Genetic Algorithms and Cultural Algorithms. We solve this problem with a Cultural Evolutionary Architecture maintaining knowledge of Diversity and Fitness learned over each generation during the search process and we compare it with a Genetic Algorithm using the same crossover and mutation mechanisms. Our results indicate that the approach is able to produce very competitive results in compare with other Metaheuristics and Approximation Algorithms.

1 INTRODUCTION

Cultural algorithms are a technique that incorporates knowledge obtained during the evolutionary process trying to make the search process more efficient. Cultural algorithms have been successfully applied to several types of optimization problems (Coello and Landa, 2002; Landa and Coello, 2005b). However, nobody had proposed a cultural algorithm for the Set Covering Problem.

Set partitioning problem (SPP) and set covering problem (SCP) are two types of problems that can model several real life situations (Feo and Resende, 1989; Chu and Beasley, 1998). In this work, we solve some benchmarks of SCP with a new evolutive approach: cultural algorithms (Reynolds, 1994; Reynolds, 1999; Reynolds and Peng, 2004).

This paper is organized as follows: In Section 2, we formally describe SCP using mathematical programming models. In section 3 we present the cultural evolutionary architecture. In sections 4 we show the population space and the belief space considered in cultural algorithms. In Section 5, we present experimental results obtained when applying the algorithm for solving some standard benchmarks taken from the ORLIB (Beasley, 1990). Finally, in Section 6 we conclude the paper and give some perspectives for future research.

2 PROBLEM DESCRIPTION

SPP is the NP-complete problem of partitioning a given set into mutually independent subsets while minimizing a cost function defined as the sum of the costs associated to each of the eligible subsets (Chu and Beasley, 1998). In the SPP linear programming formulation we are given a m rows and n columns incidence matrix $A = (a_{ij})$ in which all the matrix elements are either zero or one. Additionally, each column is given a non-negative real cost c_j. We say that a column j can cover a row i if $a_{ij} = 1$. Let J denotes the set of the columns and x_j a binary variable which is one if column j is chosen and zero otherwise. The SPP can be defined formally as follows:

$$Minimize \qquad f(x) = \sum_{j=1}^{n} c_j \, x_j \qquad (1)$$

Crawford B., Lagos C., Castro C. and Paredes F. (2007).
A EVOLUTIONARY APPROACH TO SOLVE SET COVERING.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - AIDSS*, pages 356-360
DOI: 10.5220/0002406703560360

$$Subject\ \ to \qquad \sum_{j=1}^{n} a_{ij}\, x_j = 1; \qquad \forall i = 1,\ldots,m \quad (2)$$

$$x_j \in 0,1 \qquad i = 1,\ldots,m \qquad (3)$$

These constraints enforce that each row is covered by exactly one column. The SPP has been studied extensively over the years because of its many real world applications. The SCP is a SPP relaxation (replacing the equation 2 by 4. The goal in the SCP is to choose a subset of the columns of minimal weight which covers every row. The SCP can be defined formally using constraints to enforce that each row is covered by at least one column as follows:

$$\sum_{j=1}^{n} a_{ij}\, x_j \geq 1; \qquad \forall i = 1,\ldots,m \qquad (4)$$

3 CULTURAL EVOLUTIONARY ARCHITECTURE

The cultural algorithms were developed by Robert G. Reynolds (Reynolds, 1994; Reynolds and Peng, 2004) as a complement to the metaphor used by evolutionary algorithms that are mainly focused on natural selection and genetic concepts. The cultural algorithms are based on some theories which try to model cultural as an inheritance process operating at two levels: a micro-evolutionary level, which consists of the genetic material that an offspring inherits from its parent, and a macro-evolutionary level, which is the knowledge acquired by the individuals through generations. This knowledge, once encoded and stored, it serves to guide the behavior of the individuals that belong to a population.

Considering that evolution can be seen like an optimization process, Reynolds developed a computational model of cultural evolution that can have applications in optimization (Coello and Landa, 2002; Landa and Coello, 2005b). He considered the phenomenon of double inheritance with the purpose of increase the learning or convergence rates of an evolutionary algorithm. In this model each one of the levels is represented by a space. The *micro-evolutionary* level is represented by the population space and the *macro-evolutionary* level by the belief space.

The population space can be adopted by anyone of the paradigms of evolutionary computation, such as the genetic algorithms, the evolutionary strategies or the evolutionary programming. In all of them there is a set of individuals where each one has a set of independent characteristics with which it is possible to determine its fitness. Through time, such individuals could be replaced by some of their descendants, obtained from a set of operators (crossover and mutation, for example) applied to the population.

The belief space is the ”store of the knowledge” acquired by the individuals along the generations. The information in this space must be available for the population of individuals. There is a protocol of communication established to dictate rules about the type of information that it is necessary to interchange between the spaces. This protocol defines two functions: *acceptance*, this function extracts the information (or experience) from the individuals of a generation putting it into the belief space; and *Influence*, this function is in charge ”to influence” in the selection and the variation operators of the individuals (as the crossover and mutation in the case of the genetic algorithms). This means that this function exerts a type of pressure according to the information stored in the belief space.

In the belief space of a cultural evolution model there are different types of knowledge: Situational, Normative, Topographic, Historical or Temporal, and Domain Knowledge. According to Reynolds and Peng (Reynolds, 1999; Peng, 2005) these types conform a complete set, that is any other type of knowledge can be generated by means of a combination of two or more of the previous types of knowledge. The pseudo-code of a cultural algorithm appears later (Landa and Coello, 2005a). Most of the steps of a cultural algorithm correspond with the steps of a traditional evolutionary algorithm. It can be clearly seen that the main difference lies in the fact that cultural algorithm use a belief space. In the main loop of the algorithm, we have the update of the belief space, at this point the belief space incorporates the individual experiences of a select group of members with the acceptance function, which is applied to the entire population.

Pseudo-code of a Cultural Algorithm

```
Generate the initial population
Initialize the space of beliefs
Evaluate the initial population
Repeat
   Update the space of beliefs
   (with the individuals accepted)
   Apply the variation operators(under
    the influence of the space of beliefs)
   Evaluate each child
   Perform selection
While the end condition is not satisfied
```

4 POPULATION SPACE AND BELIEF SPACE

In the design and development of our cultural algorithm solving SCP we considered in the population space a genetic algorithm with binary representation. An individual, solution or chromosome is an n-bit string, where a value 1 in the bit indicates that the column is considered in the solution and zero in another case (value in j-bit corresponds to value of in the linear programming model). The initial population was generated with n selected individuals randomly with a repair process in order to as-sure the feasibility of the individuals. For the selection of parents we used binary tournament and the method of the roulette. For the process of variation we used the operator of basic crossover and the fusion operator proposed by Beasley and Chu (Beasley and Chu, 1996), for mutation we used interchange and multibit. For the treatment of unfeasible individuals we applied the repairing heuristic proposed by Beasley and Chu too. In the re-placement of individuals we use the strategy steady state and the heuristic proposed by Lozano et al. (Lozano et al., 2003), which is based on the level of diversity contribution of the new offspring. The genetic diversity was calculated by the Hamming distance, which is defined as the number of bit differences between two solutions. The fitness function is determined for:

$$f_i = \sum_{j=1}^{n} c_j \, s_{ij} \tag{5}$$

Where S is the set of columns in the solutions, S_{ij} is the value of bit (or column) j in the string corresponding to individual i and c_j is the cost of the bit. The main idea is try to replace a solution with worse fitness and with lower contribution of diversity than the one provided by the offspring. In this way, we are working with two underlying objectives simultaneously: to optimize the fitness and to promote useful diversity.

The main idea is try to replace a solution with worse fitness and with lower contribution of diversity than the one provided by the offspring. In this way, we are working with two underlying objectives simultaneously: to optimize the fitness and to promote useful diversity.

In a cultural algorithm, the shared belief space is the foundation supporting the efficiency of the search process. In order to find better solutions and improve the con-vergence speed we incorporated information about the diversity in the belief space. We stored in the belief space the individual with better fitness of the current generation and the individual who delivers major diversity to the population, which will be considered leaders in the space of beliefs. With this type of knowledge situational, each of the new individuals generated try to follow a leader stored in the space of beliefs.

A situational-fitness knowledge procedure selects from the initial population the individual with better fitness, which will be a leader in the situational-fitness space of beliefs. A situational-diverse knowledge procedure selects from the initial population the most diverse individual of the population, which will be a leader in the situational-diverse space of beliefs.

In this work, we implemented the influence of situational-fitness knowledge in the operator of crossover. The influence initially appears at the moment of the parental selection, the first father will be chosen with the method of binary tournament and the second father will be the individual with better fitness stored in the space of beliefs. In relation with the influence of situational-diverse knowledge in the operator of cross-over, this procedure works recombining the individual with better fitness of every generation with the most diverse stored in the space of beliefs, with this option we expect to deliver diversity to the population.

The updating the situational belief space procedure implies that the situational space of beliefs will be updated in all generations of the evolutionary process. The update of the situational space of beliefs consists in the replacement of the individuals by current generation individuals if they are better considering fitness and diversity.

5 COMPARISON OF RESULTS

The following results present the cost obtained when applying different operators in our algorithm for solving SCP41, SCP42, SCP48, SCP51, SCP61, SCP62, SCP63, SCPa1, SCPb1, SCPc1 benchmarks, these test problem sets were obtained electronically from OR-Library (Beasley, 1990). The first table shows the optimal values and the number of rows(m), number of columns(n) for diverse instances of the Set Covering Problem. Following the same sequence that the table 1, the table 2 shows the cost from a Genetic Algorithm using the basic proposal described in section 4 not considering diversity. The next column presents the best cost obtained when applying our Cultural Algorithm. The table 3 shows the results applying Ant System (AS) and Ant Colony System (ACS) taken from (Crawford and Castro, 2006) and Round, Dual-LP, Primal-Dual, Greedy taken from (Gomes et al.,

Table 1: Test problem details for different instances of the SCP.

Problem	Optimal	Rows(m)	Columns(n)
SCP41	429	200	1000
SCP42	512	200	1000
SCP48	492	200	1000
SCP51	253	200	2000
SCP61	138	200	1000
SCP62	146	200	1000
SCP63	145	200	1000
SCPa1	253	300	3000
SCPb1	69	300	3000
SCPc1	227	400	4000

Table 2: Obtained costs using Genetic Algorithm (GA) and Cultural Algorithm (CA) with Basic Crossover and Mutation Interchange.

Problem	GA	CA
SCP41	519	446
SCP42	627	569
SCP48	642	636
SCP51	303	295
SCP61	179	151
SCP62	183	162
SCP63	198	196
SCPa1	426	300
SCPb1	97	95
SCPc1	238	277

2006). The algorithm has been run with the following parameters setting: size of the population (n) =100, size of the tournament (t) =3, number of generations (g) =30, probability of crossing (pc) =0.6 and probability of mutation (pm) =0.2. The algorithm was implemented using ANSI C, GCC 3.3.6, under Microsoft Windows XP Professional version 2002.

Table 3: Obtained costs using algorithms to solve SCP from different authors.

Problem	AS	ACS	Round	Dual	Primal -LP	Greedy -Dual
SCP41	473	463	429	505	521	463
SCP42	594	590	*	*	*	*
SCP48	524	522	*	*	*	*
SCP51	289	280	405	324	334	293
SCP61	157	154	301	210	204	155
SCP62	169	163	347	209	232	170
SCP63	161	157	*	*	*	*
SCPa1	*	*	592	331	348	288
SCPb1	*	*	196	115	101	75
SCPc1	*	*	592	331	348	288

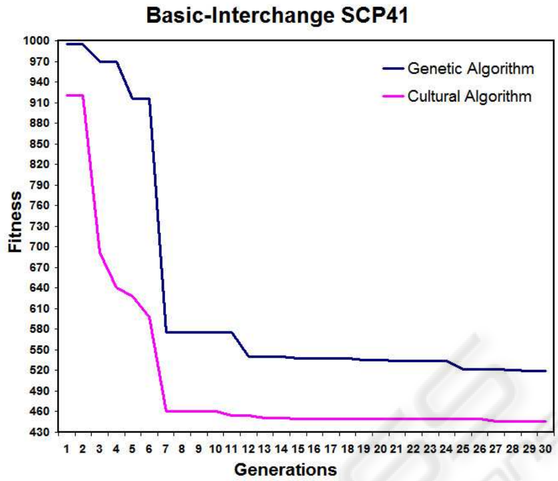

Figure 1: Fitness versus generations are showed for SCP41 instances, using Basic Crossover and Mutation Interchange.

6 CONCLUSIONS

In this paper we have introduced the first proposal to solve SCP with Cultural Algorithms. The main idea of the Cultural Algorithms is to incorporate knowledge acquired during the search process, integrating inside of an evolutionary algorithm the so called Belief Space. The objective is to do more robust algorithms with greater rates of convergence. Our computational results confirm that incorporating information about the diversity of solutions we can obtain good results in the majority of the experiments. Our main conclusion from this work is that we can improve the performance of Genetic Algorithms considering additional information in the evolutionary process. Genetic Algorithms tends to lose diversity very quickly. In order to deal with this problem, we have shown that maintaining diversity in the Belief Space we can improve the computational efficiency. Evolutionary Algorithms can be seen as two phase process. The first phase explores for good solutions, while the second phase exploits the better solutions. Both phases appear to require different diversity. Then, in order to improve further the performance of our algorithm, we are now exploring the possibilities for employing knowledge about diversity from the Belief Space only in the first generations of the Population Space. In this way, considering that population diversity and selective pressure (that gives individuals with higher fitness a higher chance of being selected for reproduction) are inversely related, we also plan to work on a non-deterministic use of the two functions (accept and influence) communication. The main idea behind this approach is an adaptive solving way that it evaluates the trade-offs between exploration and explotation during the process.

REFERENCES

Beasley, J. E. (1990). Or-library:distributing test problem by electronic mail. *Journal of Operational Research Society*, 41(11):1069–1072. Available at http://people.brunel.ac.uk/ mastjjb/jeb/info.html.

Beasley, J. E. and Chu, P. C. (1996). A genetic algorithm for the set covering problem. *European Journal of Operational Research*, 94(2):392–404.

Chu, P. C. and Beasley, J. E. (1998). Constraint handling in genetic algorithms: The set partitioning problem. *Journal of Heuristics*, 4(4):323–357.

Coello, C. A. and Landa, R. (2002). Constrained Optimization Using an Evolutionary Programming-Based Cultural Algorithm. In Parmee, I., editor, *Proceedings of the Fifth International Conference on Adaptive Computing Design and Manufacture (ACDM 2002)*, volume 5, pages 317–328, University of Exeter, Devon, UK. Springer-Verlag.

Crawford, B. and Castro, C. (2006). Integrating lookahead and post processing procedures with aco for solving set partitioning and covering problems. In *ICAISC*, pages 1082–1090.

Feo, T. and Resende, M. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8:67–71.

Gomes, F. C., Meneses, C. N., Pardalos, P. M., and Viana, G. V. R. (2006). Experimental analysis of approximation algorithms for the vertex cover and set covering problems. *Comput. Oper. Res.*, 33(12):3520–3534.

Landa, R. and Coello, C. A. (2005a). Optimization with constraints using a cultured differential evolution approach. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 27–34, New York, NY, USA. ACM Press.

Landa, R. and Coello, C. A. (2005b). Use of domain information to improve the performance of an evolutionary algorithm. In *GECCO '05: Proceedings of the 2005 workshops on Genetic and evolutionary computation*, pages 362–365, New York, NY, USA. ACM Press.

Lozano, M., Herrera, F., and Cano, J. R. (2003). Replacement strategies to preserve useful diversity in steady-state genetic algorithms. In *Proceedings of the 8th Online World Conference on Soft Computing in Industrial Applications*.

Peng, B. (2005). *Knowledge and population swarms in cultural algorithms for dynamic environments*. PhD thesis, Detroit, MI, USA. Adviser-Robert G. Reynolds.

Reynolds, R. (1994). An introduction to cultural algorithms. In *Third Annual Conference on Evolutionary Programming*, pages 131–139.

Reynolds, R. G. (1999). Cultural algorithms: theory and applications. In *New ideas in optimization*, pages 367–378, Maidenhead, UK, England. McGraw-Hill Ltd., UK.

Reynolds, R. G. and Peng, B. (2004). Cultural algorithms: Modeling of how cultures learn to solve problems. In *ICTAI*, pages 66–172.