20

Performance Study of Cultural Algorithms Based on Genetic Algorithm with Single and Multi Population for the MKP

Deam James Azevedo da Silva[1], Otávio Noura Teixeira[2] and Roberto Célio Limão de Oliveira[1]
[1]*Universidade Federal do Pará (UFPA),*
[2]*Centro Universitário do Estado do Pará (CESUPA)*
Brazil

1. Introduction

Evolutionary Computation (EC) is inspired from by evolution that explores the solution space by gene inheritance, mutation, and selection of the fittest candidate solutions. Since their inception in the 1960s, Evolutionary Computation has been used in various hard and complex optimization problems in search and optimization such as: combinatorial optimization, functions optimization with and without constraints, engineering problems and others (Adeyemo, 2011). This success is in part due to the unbiased nature of their operations, which can still perform well in situations with little or no domain knowledge (Reynolds, 1999). The basic EC framework consists of fairly simple steps like definition of encoding scheme, population generation method, objective function, selection strategy, crossover and mutation (Ahmed & Younas, 2011). In addition, the same procedures utilized by EC can be applied to diverse problems with relatively little reprogramming.

Cultural Algorithms (CAs), as well as Genetic Algorithm (GA), are evolutionary models that are frequently employed in optimization problems. Cultural Algorithms (CAs) are based on knowledge of an evolutionary system and were introduced by Reynolds as a means of simulating cultural evolution (Reynolds, 1994). CAs algorithms implements a dual mechanism of inheritance where are inherited characteristics of both the level of the population as well as the level of the area of belief space (culture). Algorithms that use social learning are higher than those using individual learning, because they present a better and faster convergence in the search for solutions (Reynolds, 1994). In CAs the characteristics and behaviors of individuals are represented in the Population Space. This representation can support any population-based computational model such as Genetic Algorithms, Evolutionary Programming, Genetic Programming, Differential Evolution, Immune Systems, among others (Jin & Reynolds, 1999).

Multidimensional Knapsack Problem (MKP) is a well-known nondeterministic-polynomial time-hard combinatorial optimization problem, with a wide range of applications, such as cargo loading, cutting stock problems, resource allocation in computer systems, and

economics (Tavares et al., 2008). MKP has received wide attention from the operations research community, because it embraces many practical problems. In addition, the MKP can be seen as a general model for any kind of binary problems with positive coefficients (Glover & Kochenberger, 1996).

Many researchers have proposed the high potential of the hybrid-model for the solution of problems (Gallardo et al., 2007). The algorithms presented in this work to solve MKP are a combination of CAs with a Multi Population model. The Multi Population model is the division of a population into several smaller ones, usually called the island model. Each sub-population runs a standard sequential evolution proceeds, as if it were isolated from the rest, with occasional migration of individuals between sub-populations (Tomassini, 2005).

In order to conduct an investigation to discover improvements for MKP, this work is centered in the knowledge produced from CAs through the evolutionary process that utilizes a population-based Genetic Algorithm model, using various MKP benchmarks found in the literature. In addition, there is an interest in investigating how to deal with the Cultural Algorithms considering a population-based in Genetic Algorithms.

So as to compare test results, we implemented the follows algorithms: the standard cultural algorithm with Single Population (also known as standard CA or CA-S) and Cultural Algorithm with Multi Population defined as CA-IM with two versions: CA-IM_1 which has fixed values for genetic operators (recombination and mutation) and CA-IM_2 which does not have fixed values for genetic operators because these values are generated randomly. In order to evaluate the performance of the CA-IM algorithms, some comparison testing will be conducted with other two algorithms based on Distributed GA, called DGA and DGA-SRM (Aguirre et al., 2000).

The outline of the paper is as follows: in Section 2, a description with formal definition of the MKP problem and an overview of Cultural Algorithms are presented. Section 3 shows an alternative approach that explores the multi population model with Cultural Algorithms and explores how the interaction process occurs among various sub-populations. Our experimental results are shown in Section 4 and finally we show some conclusions in Section 5.

2. Background

Since the introduction of the Knapsack problems some algorithm techniques such as brute force, conventional algorithms, dynamic programming, greedy approach and approximation algorithm have been proposed (Ahmed & Younas, 2011).

Evolutionary algorithms (EAs) have been widely applied to the MKP and have shown to be effective for searching and finding good quality solutions (Chu & Beasley, 1998). It is important to note that MKP is considered a NP hard problem; hence any dynamic programming solution will produce results in exponential time. In the last few years, Genetic Algorithms (GAs) have been used to solve the NP-complete problems and have shown to be very well suited for solving larger Knapsack Problems (Fukunaga & Tazoe, 2009; Gunther, 1998; Sivaraj & Ravichandran, 2011). For larger knapsack problems, the efficiency of approximation algorithms is limited in both solution quality and computational

cost (Ahmed & Younas, 2011). Spillman's experiment, which applies the GA to the knapsack problem, shows that the GA does not have a good performance in relatively small size problem, but works quite well in problems that include a huge number of elements (Spillman, 1995). There are many packing problems where evolutionary methods have been applied. The simplest optimization problem and one of the most studied is the one-dimensional (zero–one or 0-1) knapsack problem (Ahmed & Younas, 2011), which given a knapsack of a certain capacity, and a set of items, each one having a particular size and value, finds the set of items with maximum value which can be accommodated in the knapsack. Various real-world problems are of this type: for example, the allocation of communication channels to customers who are charged at different rates (Back et al., 1997).

During a study of 0-1 knapsack, a number of extensions and variants have been developed such as (Ahmed & Younas, 2011): Multiple Knapsack Problems (MKP), Multidimensional Knapsack Problems (MDKP), Multi Choice Knapsack Problems (MCKP) and Multiple Choice Multidimensional Knapsack Problems (MMKP). It is also important to consider other extensions such as (Chu & Beasley, 1998): Multiconstraint Knapsack Problem, and also the term "Multidimensional Zero-one Knapsack Problem". Using alternative names for the same problem is potentially confusing, but since, historically, the designation **MKP** has been the most widely used (Chu & Beasley, 1998). Consequently, Multidimensional Knapsack Problem (MKP) is the designation selected for this work. In our previous research it was introduced a Multi Population Model on the cultural structure identified as "Multi Population Cultural Genetic Algorithm" (MCGA) (Silva & Oliveira, 2009). In MCGA model several sub-populations are connected with as ring structure, where the migration of individuals occurs after a generation interval (according to the migration based on parameter interval) with best-worst migration policy implementation. The results were satisfactory in relation to other algorithms in the literature. In another research two versions of Distributed GA (DGA) are presented as follows: standard Distributed GA (DGA) and an improved DGA (DGA-SRM), which two genetic operators are applied in parallel mode to create offspring. The term SRM represents "Self-Reproduction with Mutation", that is applied to various 0/1 multiple knapsack problems so as to improve the search performance (Aguire et al., 2000). Hybridization of memetic algorithms with Branch-and-Bound techniques (BnB) is also utilized for solving combinatorial optimization problems (Gallardo et al., 2007). BnB techniques use an implicit enumeration scheme for exploring the search space in an "intelligent" way. Yet another research utilizes adaptive GA for 0/1 Knapsack problems where special consideration is given to the penalty function where constant and self-adaptive penalty functions are adopted (Zoheir, 2002). Fitness landscape analysis techniques are used to better understand the properties of different representations that are commonly adopted when evolutionary algorithms are applied to MKP (Tavares et al., 2008). Other investigation utilizes multiple representations in a GA for the MKP (Representation-Switching GA) know as RSGA (Fukunaga, 2009). Other recent works consider two heuristics and utilize them for making comparisons to the well-known multiobjective evolutionary algorithms (MOEAs) (Kumar & Singh, 2010). While comparing MOEAs with the two heuristics, it was observed that the solutions obtained by the heuristics are far superior for larger problem instances than those obtained by MOEAs.

2.1 Multidimensional Knapsack Problem

As mentioned earlier, the MKP is a well-known nondeterministic-polynomial time-hard combinatorial optimization problem, with a wide range of applications (Tavares et al., 2008). The classical 0-1 knapsack problem is one of the most studied optimization and involves the selection of a subset of available items having maximum profit so that the total weight of the chosen subset does not exceed the knapsack capacity. The problem can be described as follows: given two sets of n items and m knapsacks constraints (or resources), for each item j, a profit p_j is assigned, and for each constraint i, a consumption value r_{ij} is designated. The goal is to determine a set of items that maximizes the total profit, not exceeding the given constraint capacities c_i. Formally, this is stated as follows (Tavares et al., 2008):

Maximize
$$\sum_{j=1}^{n} p_j x_j \,, \tag{1}$$

Subject to
$$\sum_{j=1}^{n} r_{i,j} x_j \le c_{i,} \quad \text{i =1,…,}m \tag{2}$$

$$x_j \in \{0,1\}, \quad \text{j =1,….,}n \tag{3}$$

With
$$p_j > 0 \,,\; r_{i,j} \ge 0 \;\text{ and }\; c_i \ge 0 \tag{4}$$

The knapsack constraint is represented by each of the m constraints described in Eq. (2). The decision variable is the binary vector x =(x_1,...,x_n). Each item j is mapped to a bit and when x_j **=1**, the corresponding item is considered to be part of the solution. The special case of m ***=1*** is generally known as the Knapsack Problem or the Unidimensional Knapsack Problem.

For single constraint the problem is not strongly NP-hard and effective approximation algorithms have been developed for obtaining near-optimal solutions. A review of the single knapsack problem and heuristic algorithms is given by Martello and Toth (Martello & Toth, 1990). Exact techniques and exhaustive search algorithms, such as branch-and-bound, are only of practical use in solving MKP instances of small size since they are, in general, too time-consuming (e.g., instances with 100 items or less, and depending on the constraints).

2.2 Evolutionary approach for the MKP

In a resolution of specific problems that implements an Evolutionary Algorithm, as for example, a simple Genetic Algorithm (GA), it is necessary the definition of five components (Agapie et al., 1997). The first component is the genotype or a genetic representation of the potential problem (individual representation scheme). The second is a method for creating an initial population of solutions. The third is a function verifying the fitness of the solution (*objective function* or *fitness function*). The fourth are genetic operators and the fifth are some

constant values for parameters that are used by the algorithm (such as population size, probability of applying an operator, etc.).

2.2.1 Genotype

The natural representation of the MKP would be the binary representation, in which every bit represents the existence or not of a certain element in the Knapsack. A bit set to 1 indicates that the corresponding item is packed into the knapsack and a bit set to 0 indicates that it is not packed. Hence a typical population of two individuals for a six elements in Knapsack would be represented as showed in Figure 1. Thus, each element has an identification that is given by the bit index.

In Figure 1 (a) there are three elements in the knapsack, corresponding to the following positions: 1, 4 and 6. In Figure 1 (b) there are four elements in the knapsack, whose positions are: 2, 3, 5 and 6.

Fig. 1. Knapsack example for two chromosomes.

2.2.2 Initial population

The population is the solution representation that consists of a set of codified chromosomes. There are many ways to generate the initial population such as random chromosome or chromosome with the solution closer to the optimum. In most applications the initial population is generated at random.

2.2.3 Evaluation function

In GA each individual is evaluated by fitness function. Some individuals produce more children than others due to their fitness. By this mechanism, individuals that have chromosomes with better fitness have better chances of leaving their genes. This leads to better average performance of the whole population as generations proceed (Ku & Lee, 2001). A feasible vector solution $\boldsymbol{x}$ needs to satisfy constraint (2), otherwise it is infeasible. Hence, a penalty is applied to all infeasible solutions in order to decrease their corresponding "fitness". Therefore, the two types of evaluation functions used in this research are based on static (constant) and adaptive penalty functions. The standard evaluation function for each individual is given by the following expressions:

$$\text{Evaluation (x)} = \sum_{i=1}^{i=n} (x[i] \times p[i]) - Pen(x) \tag{5}$$

$$\text{Maximum Profit Possible (MaxP)} = \sum_{i=1}^{i=n} p[i] \tag{6}$$

A vector solution x is optimal when Evaluation (x) =MaxP.

2.2.4 Genetic operators

To implement the GA process, many factors should be considered such as the representation scheme of chromosomes, the mating strategy, the size of population, and the design of the genetic operators such as selection, mutation and recombination (Ku & Lee, 2001).

i. **Selection** - is an operator that prevents low fitness individuals from reproduction and permits high fitness individuals to offspring more children to improve average fitness of population over generations. There are various selections types, such as stochastic remainder, elitism, crowding factor model, tournament, and roulette wheel.
ii. **Recombination or Crossover** - is an operator that mixes the chromosomes of two individuals. Typically two children are generated by applying this operator, which are similar to the parents but not same. Crossover causes a structured, yet randomized exchange of genetic material between solutions, with the possibility that the "fittest" solutions generate "better" ones. A crossover operator should preserve as much as possible from the parents while creating an offspring.
iii. **Mutation** - introduces totally new individuals to population. It helps extend the domain of search and will restrain the diversity of the population. Mutation involves the modification of each bit of an individual with some probability Pm. Although the mutation operator has the effect of destroying the structure of a potential solution, chances are it will yield a better solution. Mutation in GAs restores lost or unexplored genetic material into the population to prevent the premature convergence of the GA.

The tournament is the selection type chosen for this work since it is more used and it presents good performance. For a binary representation, classical crossover and mutation operators can be used, such as n-point crossover or uniform crossover, and bit-flip mutation. In CAs the influence of information from Belief Space on recombination and mutation process such as: best chromosome or set of best chromosomes information is expected.

2.2.5 Constant values parameters

An Evolutionary Algorithm involves different strategy parameters, e.g.: mutation rate, crossover rate, selective pressure (e.g., tournament size) and population size. Good parameter values lead to good performance. There are three major types of parameter control (Eiben & Smith, 2008):

- **deterministic**: a rule modifies strategy parameter without feedback from the search (based on some type of a counter);
- **adaptive**: a feedback rule based on some measure monitoring search progress (quality);
- **self-adaptative**: parameter values evolve along with the solutions; encoded onto chromosomes they undergo variation and selection.

The implementation of a deterministic parameter control is easier, provided that the parameter values used are tested to verify the best performance.

2.3 Cultural algorithms

Cultural Algorithms (CAs) have been developed so as to model the evolution of the cultural component of an evolutionary computational system over time as it accumulates experience

(Reynolds & Chung, 1996). As a result, CAs can provide an explicit mechanism for global knowledge and a useful framework within which to model self-adaptation in an EC system. The CAs are based on knowledge of an evolutionary system that implements a dual mechanism of inheritance. This mechanism allows the CAs to explore as much microevolution as macroevolution. Microevolution is the evolution that happens in the population level. Macroevolution occurs on the culture itself, i.e. the belief space evolution. The belief space is the place where the information on the solution of the problem is refined and stored. It is acquired through the population space over the evolutionary process. The belief space has the goal to guide individuals in search of better regions. In the CAs evolution occurs more quickly than in population without the mechanism of macroevolution. The characteristics and behaviors of individuals are represented in the Population Space and as mentioned earlier the population space can support any population-based computational model such as Genetic Algorithms among others (Jin & Reynolds, 1999). The communications protocols dictate the rules about individuals that can contribute to knowledge in the Belief Space (function of acceptance) and how the Belief Space will influence new individuals (Function of Influence), as shown in Figure 2.

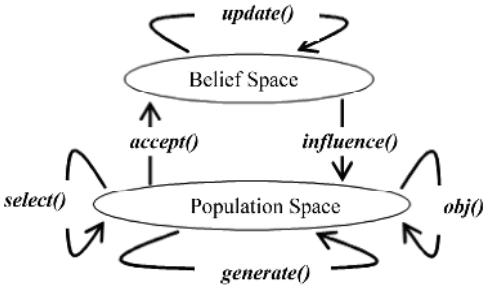


Fig. 2. Framework of Cultural Algorithm (Reynolds & Peng, 2004).

The two most used ways to represent knowledge in the belief space are (Reynolds & Peng, 2004): Situational Knowledge and Normative Knowledge. Situational Knowledge represents the best individuals found at a certain time of evolution and it contains a number of individuals considered as a set of exemplars to the rest of the population. The number of exemplars may vary according to the implementation, but it is usually small. For example, the structure used to represent this type of knowledge is shown in Figure 3. Each individual is stored within its parameters and fitness value (Iacoban et al. 2003).

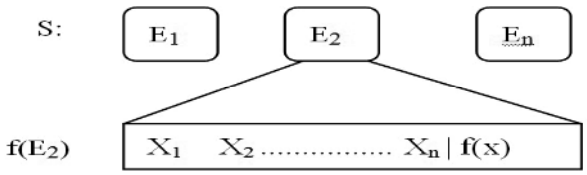


Fig. 3. Representation of Situational Knowledge.

The Situational Knowledge is updated when the best individual of the population is found. This occurs when its fitness value exceeds the fitness value of the worst individual stored.

Normative Knowledge represents a set of intervals that characterize the range of values given by the features that make the best solutions (Iacoban et al., 2003).

Figure 4 shows the structure used by Reynolds and his students, where are stored the minimum and maximum values on the individual's characteristics.

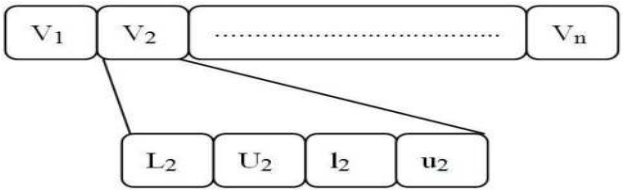


Fig. 4. Representation of Normative knowledge

These intervals are used to guide the adjustments (mutations) that occur in individuals. With these minimum values, (li) and maximum (ui), the fitness values are also stored. This value results from the individuals that produced each extreme Li and Ui respectively.

The adjustment of the range of Normative Knowledge varies according to the best individual. That is, if the individual was accepted by the *acceptance function* and its range is less than the range stored in the belief space, the range is adjusted, and vice versa.

The resolution of problems produces experiences from individuals in the population space, which are selected to contribute to the acceptance by the belief space, where the knowledge is generalized and stored. In the initial population, the individuals are evaluated by the fitness function. Then, the information on the performance of the function is used as a basis for the production of generalizations for next generations. The experiences of the individuals selected will be used to make the necessary adjustments on the knowledge of the current belief space.

2.4 Parallel Genetic Algorithms

The definition of Parallel Genetic Algorithms (PGAs) is related with execution of various GAs in parallel mode. The main goal of PGAs is to reduce the large execution times that are associated with simple genetic algorithms for finding near-optimal solutions in large search spaces and to find better solutions.

The PGAs can be implemented through two approaches (Sivanandam, 2007): standard parallel approach and the decomposition approach. In the first approach, the sequential GA model is implemented on a parallel computer by dividing the task of implementation among the processors. The standard parallel approaches are also known as *master-slave GAs.* In the decomposition approach, the full population exists in distributed form. Other characteristic in the decomposition approach is that the population is divided into a number of sub-populations called demes. Demes are separated from one another and individuals compete only within a deme. An additional operator called migration is used to move the individuals from one deme to another. If the individuals can migrate to any other deme, the model is called *island model* or *Multiple-population GAs* when implemented in parallel or distributed environments (Braun, 1991). Migration can be controlled by various parameters

like migration rate, topology, migration scheme like best/worst/random individuals to migrate and the frequency of migrations (Sinvanadam, 2007).

Other authors classify Parallel Genetic Algorithm in four main categories (Aguirre & Tanaka, 2006): global master-slave, island, cellular, and hierarchical parallel GAs. In a global master-slave GA there is a single population and the evaluation of fitness is distributed among several processors. The important characteristic in a global master-slave GA is that the entire population is considered by genetic operators as selection, crossover and mutation. An island GA, also known as coarse-grained or distributed GA, consists of several sub-populations evolving separately with occasional migration of individuals between sub-populations. A cellular category also known as "fine-grained GA" consists of one spatially structured population, whose selection and mating are restricted to a small neighborhood. The neighborhoods are allowed to overlap permitting some interaction among individuals. Finally, a hierarchical parallel GA category, combines an island model with either a master-slave or cellular GA. The global master-slave GA does not affect the behavior of the algorithm and can be considered only as a hardware accelerator. However, the other parallel formulations of GAs are very different from canonical GAs, especially, with regard to population structure and selection mechanisms. These modifications change the way the GA works, affecting its dynamics and the trajectory of evolution. For example, the utilization of parameters as sub-population size, migration rate and migration frequency are crucial to the performance of island models. Cellular, island and hierarchical models perform as well as or better than canonical versions and have the potential of being more than just hardware accelerators (Aguirre & Tanaka, 2006). A new taxonomy about PGAs is also presented by Nowostawski and Poli (Nowostawski & Poli, 1999).

In recent studies about MKP Silva and Oliveira (Silva & Oliveira, 2009) have shown that good results are reached in the benchmark tests when taking into consideration the implementation of sub-populations and the migration process from the island model. The results presented were better than canonical version of Cultural Algorithm in most cases.

2.5 Island model (Multi Population Genetic Algorithms)

Multi population Genetic Algorithms (MGAs) or Island Model, is an extension of traditional single-population Genetic Algorithms (SGAs) by dividing a population into several sub-populations within which the evolution proceeds and individuals are allowed to migrate from one sub-population to another. Different values for parameters such as selection, recombination and mutation rate can be chosen for each sub-population. Normally, the basic island model uses the same values for these parameters in all sub-populations.

In order to control the migration of individuals, several parameters were defined such as: (i) the communication topology that defines the connections between sub-populations, (ii) a migration rate that controls how many individuals migrate, and (iii) a migration interval that affects the frequency of migration. In addition, migration must include strategies for migrant selection and for their inclusion in their new sub-populations (Aguire, 2000).

The sub-populations size, communication topology (its degree of connectivity), migration rate and migration frequency are important factors related to the performance of distributed GAS. In general, it has been shown that distributed GAs can produce solutions with similar or better quality than single population GAs, while reducing the overall time

to completion in a factor that is almost in reciprocal proportion to the number of processors (Aguire, 2000).

In the island model GA, the sub-populations are isolated during selection, breeding and evaluation. Islands typically focus on the evolutionary process within sub-populations before migrating individuals to other islands, or conceptual processors, which also carry out an evolutionary process. At predetermined times, during the search process, islands send and receive migrants to other islands. There are many variations of distributed models, e.g. islands, demes, and niching methods, where each requires numerous parameters to be defined and tuned (Gustafson, 2006).

An example of the communication topology, can be defined as a graph in which the sub-populations *Pi* (i = 0, 1,..., K - 1) are the vertices and each defined edge Li,j specifies a communication link between the incident vertices Pi and Pj (neighbor sub-populations) (Aguire, 2000). In general, assuming a directed graph for each defined link Li,j we can indicate the number of individuals Ri,j that will migrate from P to Pj (migration rate) and the number of generations M between migration events (migration interval). The communication topology and migration rates could be static or dynamic and migration could be asynchronous or synchronous.

Various strategies for choosing migrants have been applied. Two strategies often used to select migrants are selection of the best and random selection. For example, the migration can implement a synchronous elitist broadcast strategy occurring every M generation. Each sub-population broadcasts a copy of its R best individuals to all its neighbor sub-populations.

Hence, every sub-population in every migration event receives migrants. Figure 5 illustrates a communication topology +l+2 island model in which each sub-population is linked to two neighbors (L = 2). In this example, the sub-population P0 can send individuals only to P1 and P2 and receive migrants only from P4 and P5.

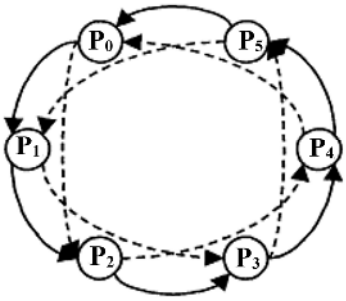


Fig. 5. +1+2 communication topology.

3. Cultural Island Model (CA-IM)

In this section is presented an approach about the communication topology for migration process implemented in a Cultural Algorithm based on the island model. As noted earlier in the classical island model implementation, there are sub-populations connected with as ring

structure. Individuals in classical island model are migrated after every migration-interval (M) among generations and the best-worst migration policy is used.

The approach utilized in this work is an adaption and implementation of the island model on the cultural structure here identified as "Cultural Island Model" (CA-IM), briefly introduced in Silva & Oliveira (Silva & Oliveira, 2009). The implementations have become simple because the same CAs structures were used as much the evolutionary structure as the belief space that is the main characteristic present in CAs.

The main characteristic present in CA-IM is the link between main belief spaces (from main population) and secondary belief space (from multi population). They store information about independent evolution for main population and sub-populations respectively, i.e. the cultural evolutions occur in parallel among the main population and the sub-populations of the islands. The link of communication between two *Belief Spaces,* allows migration between the best individuals stored in the cultural knowledge structure implemented. Figure 6 shows the framework correspondent to the proposed structure.

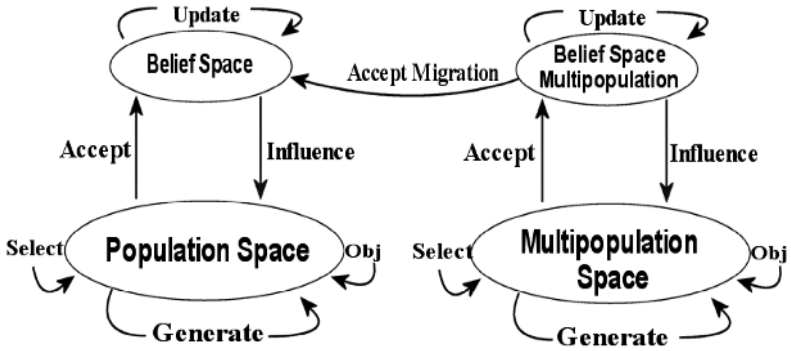


Fig. 6. Framework of model proposed

Migrations from islands occur through *Belief Space Multipopulation* structure that perform the communication process among sub-populations and send the best individuals through *Accept Migration*. It occurs in a predefined interval whose parameter is *M* (every *M* generation) where the best individuals are evaluated by acceptance function and updated in each belief space. The migration from *Belief Space Multipopulation* to *Main Belief Space* is implemented as a number of individuals which are considered as a set of exemplars to the rest of the population (Situational Knowledge).

It is important to note that CA-IM provides a continuous verification between the last solution (optimum value) found and the current solution. Then, it computes the number of generations where don't occur improvements. Thus, if the distance between the last generation, where the current solution was found, and the current generation is high then the sub-populations are eliminated and recreated randomly. As for CA-IM, there is a fixed difference for this occurrence in the range of 60 to 100 generations. If a new solution is not found in this range, then the sub-populations of the islands (Multipopulation Space) as well as the cultural information about all sub-populations (Belief Space Multipopulation) are recreated randomly by algorithm.

3.1 Mutation and recombination

In mutation operation the cultural knowledge (such as situational knowledge) as well as the standard binary mutation operation (known as "bit-flip mutation") is utilized. If the cultural knowledge is utilized during the mutation process, the mutated chromosome genes are replaced by the best genes from chromosome stored in situational knowledge with P_M probability, otherwise, the genes are inverted by bit-flip mutation. The chromosome chosen among a set of chromosomes from situational knowledge can be the best chromosome or a random chromosome.

The bit-flip mutation is a common operation applied in evolutionary algorithms to solve a problem with binary representation. Consequently, each bit from current mutated chromosome is flipped, i.e. the value of the chosen gene is inverted also with probability of mutation P_M. Figure 7 shows the pseudo-code of mutation utilized by CA-IM.

```
CA-IM_Mutation
1- Get initial Parameters:
• PM (Probability of Mutation);
• S [ ] (Situational Knowledge);
• Srandom= random chromosome of S[ ];
• C (Current Chromossome) ;
• Sbest (best chromosome of S[ ]);
2- Create C '(Chromossome Mutation)
  If (random<=0.5)
  {
      for (int i=0; i<genotypeLength; i++)
      {
          if (random <=PM)
          {
              if (random <=0.5)
                  C'[i]=Sbest[i];
              else
                  C'[i]=Srandom[i];
          }
      }
  }
  else
  {
     for (int i=0; i<genotypeLength; i++)
     {
        If (random <=PM)
           C'= flip(C, i);
     }
  }
3- Return C';
```

Fig. 7. Mutation pseudo-code.

In recombination operation the cultural knowledge as well as the standard binary recombination operation (known as "uniform recombination") is also utilized. In the uniform recombination the bits are randomly copied from the first or from the second parent to genes in the offspring chromosomes, in any sequence of ones and zeros. Figure 8 shows the pseudo-code of CA-IM recombination.

If the cultural knowledge is utilized during the recombination process, the chromosome genes are replaced by the best genes from chromosome stored in situational knowledge with P_R probability. Otherwise, the genes are replaced with genes from their parents. Here only the best chromosome is chosen from situational knowledge during the recombination process.

```
CA-IM_Recombination
1- Get initial Parameters:
  • PR (Probability of Recombination);
  • C1 (Current Chromosome -1);          // Parents Chromosomes
  • C2 (Current Chromosome -2);          // Parents Chromosomes
  • S [ ] (Situational Knowledge);
  • Sbest=best chromosome of S [ ];
2- Create C 1' and C2' // Offsprings Chromosome Recombination )
   for (int i=0; i<genotypeLength; i++)
   {
      If (random<= PR)
      {
         If (random <=0.5)
         {
            if (random<= 0.5)
            {
               C1'[i] = Sbest[i];
               C2'[i] = Sbest[i];
            }
            else
            {
               C1'[i] = C2[i];
               C2'[i] = Sbest[i];
            }
         }
         else
         {
            C1'[i] = C2[i];
            C2'[i] = C1[i];
         }
      }
      else
      {
         C1'[i] = C1[i];
         C2'[i] = C2[i];
      }
   }
3- Return Sons (C1' and C2');
```

Fig. 8. Recombination pseudo-code.

4. Experimental results and discussion

To evaluate the performance of the proposed algorithm CA-IM, a comparison of various tests with Distributed Genetic Algorithms utilizing the same knapsack problems was carried out. To make a comparison two kinds of algorithms based in Distributed GAs (Aguirre et al., 2000): (i) A Distributed canonical GA (denoted as DGA), and (ii) a Distributed GA-SRM (denoted as DGA-SRM) were utilized. The SRM term means "Self-Reproduction with Mutation", and introduces diversity by means of mutation inducing the appearance of beneficial mutations.

For the CA-IM algorithm there are two versions: CA-IM_1 and CA-IM_2. The only difference is that CA-IM_1 has a fixed rate for mutation and recombination, while CA-IM_2 has a random rate for mutation and recombination. The standard CA (CAs) is the Cultural Algorithm with single population.

4.1 DGA and DGA-SRM

The DGA works with various 0/1 multiple knapsack problems (NP hard combinatorial) which from previous efforts seem to be fairly difficult for GAs (Aguirre et al., 2000). Those algorithms were evaluated on test problems which are taken from the literature. The problem sizes range from 15 objects to 105 and from 2 to 30 knapsacks and can be found in OR-Library (Beasley, 1990). The knapsack problems are defined by: problem *(n, m)* where *n* represents the number of objects and *m* represents the number of knapsacks. Each knapsack

has a specific capacity as well each object has a specific weight. For example, Weing7 (105, 2) represents a MKP with 105 objects and 2 knapsacks.

Every experiment presented here has a similar capacity to the work described in DGA and DGA-SRM (Aguire et al., 2000) such as: population size, number of function evaluations in each run and a total of 100 independent runs. Each run uses a different seed for the random initial population. To improve understanding of DGA and DGA-SRM algorithms, some parameters and symbols are presented:

- The maximum size of the population is represented by λ_{total} (fixed in 800);
- The parent and offspring population sizes are represented by μ and λ respectively;
- The parameter K represents the number of sub-populations (partitions). Hence, $\lambda * \boldsymbol{K} = \lambda_{total}$ (maximum=800);
- The parameter ***M*** is the number of generations between migration events (migration interval) ;
- The symbol ***N*** represents the number of times the global optimum was found in the 100 runs;
- The symbol τ represents a threshold (utilized for control of a normalized mutant's survival ratio).
- The symbol ***T*** represents the number of function evaluations in each run;
- *Average* is the average of the best solutions and *Stdev* is the standard deviation around *Average*, respectively;

In DGA and DGA-SRM, each sub-population broadcasts a copy of its ***R*** best individuals to all of its neighbor sub-populations. Hence, every sub-population in every migration event receives $\lambda_m = L \times R$ migrants, where L is the number of links. When there is no migration and the sub-populations evolve in total isolation, the values corresponding to such a characteristic are denoted by X in the table. The results for knapsack problem Weing7 for DGA and DGA-SRM is shown in the Table 1 (Aguirre et al., 2000).

K	λ_m / λ	DGA						DGA-SRM					
		L R	λ	**M**	**N**	**Average**	**Stdev**	μ	λ	**M**	**N**	**Average**	**Stdev**
8	0.10	5 2	100	5	0	**1094423.4**	433.38	50	100	80	63	1095421.44	30.84
8	0.05	5 1	100	5	0	1093284.95	733.24	50	100	100	66	1095423.58	29.84
8	0.01	1 1	100	5	0	1089452.96	1082.41	50	100	80	77	**1095430.51**	26.51
8	X	X	100	X	0	1087385.56	1729.4	50	100	X	60	1095419.80	30.86

Table 1.The best results for Weing7 (105, 2) by DGA and DGA–SRM (λ_{total} =800; T=8x10^5).

According to Table 1 the best value found in *Average* is equal to 1094423.4, for DGA and 1095430.51 for DGA-SRM. Table 1 also indicates that the DGA-SRM improves the results in relation to DGA. Table 2 shows the results found for others knapsack problems by DGA and DGA-SRM. In order to simplify the results shown in Table 2, the following configuration parameters should be considered: K = 16 sub-populations and $\mu = 25$ (Aguirre et al., 2000).

Problem (n, m)	λ_m / λ	DGA						DGA-SRM (τ =0.35)				
		LR	λ	M	N	Average	Stdev	λ	M	N	Average	Stdev
Petersen6 (39,5)	0.01	1 1	50	5	0	**10506.90**	26.11	50	140	77	10614.82	5.82
Petersen7 (50,5)	0.10	5 1	100	5	0	1093284.95	733.24	50	40	89	16535	5.94
Sento1 (60, 30)	0.10	5 1	100	5	0	1089452.96	1082.41	50	40	98	7771.78	1.54
Sento2 (60, 30)	0.10	5 1	100	5	0	1087385.56	1729.4	50	40	84	8721.32	2.11

Table 2. The best results for other problems by DGA and DGA-SRM (λ_{total} = 800; T=4x10^5).

4.2 CA-IM_1

For the algorithm proposed (CA-IM) various parameters and symbols are also considered such as:

- The parameter ***P*** is the size of main population;
- The parameter P_M is the probability of mutation and P_R probability of recombination.
- The number of islands is ***K*** (number of sub-populations);
- The parameter α is the percentage which defines the size of the population of each island at function of ***P***.
- The sub-population size in each island is ***SI***, since ***SI*** = α ***P**.
- The percentage of best individuals in Situational Knowledge on population space is represented by SK_P and the percentage of best individuals in Situational Knowledge on multi population space is represented by SK_M.
- The parameter ***M*** is the number of generations between migration events (migration interval). Here ***M*** determines the interval of influence from the islands population through the Situational Knowledge.
- The symbol ***T*** represents the number of function evaluations in each run;
- The symbol ***N*** represents the number of times the global optimum was found in the 100 runs.
- *Average* is the average of the best solutions and *Stdev* is the standard deviation around *Average*;
- *Average of generations* is the average of the generations whose best solution was found in each run.

For the tests carried out for CA-IM_1, the selection chosen was tournament, whose value is 3, the mutation rate (P_M) is 0.025 and recombination rate (P_R) is 0.6. The situational knowledge configurations are: SK_P=*0.2* and SK_M=0.5. Table 3 shows the results found by CA-IM_1, whose best value found in *Average* is 1095445 (the optimal value) and in the *Average of Generations* is 44.49. All values reached have optimum value. However, if *Average of Generations* is low in relation to total of generations, then this means that the optimum is found in few generations.

As it is shown in Table 3, it is possible to observe that CA-IM outperforms DGA-SRM for any configuration such as the number of sub-populations (islands) and size of sub-population. Similarly, CA-IM also exhibits higher convergence reliability than DGA-SRM with higher values for *N* and *Average* with smaller *Stdev*. These results show that the CA-IM produces higher performance for all utilized parameters.

P	K	α	SI	M	N	Average of Generations	Average	Stdev
400	8	0.125	50	20	100	52.9	1095445	0.0
400	8	0,125	50	05	100	**44.49**	**1095445**	0.0
100	7	1.0	100	05	100	68.87	1095445	0.0

Table 3. The best results for **Weing7** (105, 2) by CA-IM_1 (λ_{total} =800 and T=8x10^5).

A new result "Average of Generations" was introduced so as to evaluate other type other type of performance whose value represents the average of generations that the optimum value was found for 100 independent runs for each problem presented. Particularly, it occurs when *M* is low and *K* is high (see result for Average of Generations). This means that a larger number of islands with small populations produce better convergence.

According to Table 3 the best value found in *Average* is 1095445 (the optimal value) while the *Average of generations* is 44.49 that means a low value, considering that 500 generations was utilized in each run which T=4x10^5. This represents 500 generations with a population size equal to 800 (including all subpopulations). Table 4 shows the results for others MKPs found by algorithm CA-IM_1.

Problem (n, m)	P	K	α	SI	M	N	Average of Generations	Average	Stdev.
Petersen6 (39,5)	400	8	0.125	50	20	100	30.22	10618.0	0.0
Petersen6 (39,5)	400	4	0,25	100	05	100	**26.29**	10618.0	0.0
Petersen7 (50,5)	400	8	0.125	50	20	100	78.49	16537.0	0.0
Petersen7 (50,5)	400	4	0,25	100	05	100	**71.51**	16537.0	0.0
Sento1 (60,30)	400	8	0.125	50	20	100	100.21	7772.0	0.0
Sento1 (60,30)	400	4	0,25	100	05	100	**87.44**	7772.0	0.0
Sento2 (60,30)	400	8	0.125	50	20	99	185.19	8721.81	0.099
Sento2 (60,30)	400	4	0,25	100	05	100	**166.12**	87722.0	0.0

Table 4. The best results for other problems by CA-IM_1 (λ_{total} = 800, T=4x10^5).

Thereby, it is possible to observe that CA-IM_1 outperforms DGA-SRM. Similarly, CA-IM_1 also exhibits higher convergence reliability (higher values of *N* and *Average* with smaller *Stdev*) than DGA-SRM. These results show that the CA-IM_1 is able to find global optimal for MKP, taking into consideration the tests results with 100% success.

The problem that presented greater difficulty was Sento2, that presented in some cases optimal values near to 100% such as N=98 and N=99. Even with results of $N < 100$ they are still better than the results obtained in the chosen benchmarks. In the meantime, the implementation of some adjustments allows CA-IM_1 to reach N=100 for Sento2.

4.3 CA-IM_2

For the tests carried out for CA-IM_2 the selection chosen was tournament whose value is 3. The mutation rate (P_M) is a random value in a specific interval: P_M= [0.01, 0.5]. The Recombination rate (P_R) is also a random value in an interval: P_R= *[0.1,* 0.99]. The situational knowledge configurations are: SK_P=*0.2* and SK_M=0.5. The CA-IM_2 results are presented in

Table 5 that shows the results for Weing7 and in Table 6 that shows the results for others knapsack problems.

P	K	α	SI	M	N	Average of Generations	Average	Stdev
400	8	0.125	50	20	100	**70.48**	1095445	0.0
400	8	0,125	50	05	100	72.72	1095445	0.0
100	7	1.0	100	05	100	107.11	1095445	0.0

Table 5. The best results for Weing7 (105,2) by CA-IM_2 (λ_{total} =800, T=8x10^5).

Problem (n, m)	P	K	α	SI	M	N	Average of Generations	Average	Stdev.
Petersen6 (39,5)	400	8	0.125	50	20	100	37.89	10618.0	0.0
Petersen6 (39,5)	400	4	0,25	100	05	100	**33.39**	10618.0	0.0
Petersen7(50,5)	400	8	0.125	50	20	100	81.46	16537.0	0.0
Petersen7(50,5)	400	4	0,25	100	05	100	**74.38**	16537.0	0.0
Sento1(60,30)	400	8	0,25	50	20	98	**112.55**	7771.75	1.7717
Sento1(60,30)	400	4	0,25	100	05	100	126.46	7772.0	0.0
Sento2(60,30)	400	8	0.125	50	20	71	183.35	8720.0	3.7199
Sento2(60,30)	400	4	0,25	100	05	88	**173.53**	8721.38	2.1732

Table 6. The best results for other problems by CA-IM_2 (λ_{total} = 800, T=4x10^5).

The implementation of random rate for mutation and recombination in CA-IM_2 doesn't produce satisfactory results in comparison to CA-IM_1, as it is shown in Table 6. In addition, the *Average of Generations* from algorithm CA-IM_2 is greater than CA-IM_1 for all knapsack problems. However, in comparison to CA-IM_1, there are few differences in results for Weing7 as is shown in Table 3 and Table 5.

4.4 CA-S (Standard CA)

For CA-S we also utilized the same configuration such as: tournament value=3, P_M= 0.025 and P_R = 0.6. The situational knowledge configuration is equal to 0.2 (SK_P=*0.2).* Every experiment presented here also consists of 100 independent runs and each run uses a different seed for the random initial population.

Problem (, m)	P	N	Average	Stdev.
Petersen6 (39,5)	800	97	10617.58	2.4002
Petersen7 (50,5)	800	81	16533.7	6.8703
Sento1 (60,30)	800	100	7772.0	0.0
Sento2 (60,30)	800	82	8721.14	2.4495
Weing7 (105,2)	800	100	1095445.0	0.0

Table 7. The best results for all knapsack problems by CA-S (T=4x10^5).

Table 7 shows the results from standard Cultural Algorithm (CA-S) that utilizes single population. According to results, the CA-S reaches optimum average for 100 runs only for Sento1 and Weing7. However, the results from CA-S for Petersen6, Pertersen7 and Sento2 outperform the results presented by DGA-SRM.

5. Conclusion

This work presented a Cultural Algorithm (CA) with single population (CA-S) and multi population (CA-IM) in order to improve the search performance on MKP. It was observed that CA-S improves the convergence reliability and search speed. However, CA-S is not enough to reach global optimum for most problems presented. Our cultural algorithm implementation with island model (CA-IM_1 and CA-IM_2) allows the migration among islands sub-populations and main population through belief space structures that represent the cultural knowledge available in Cultural Algorithms.

The results have shown that the CA-IM_1 is better than CA-IM_2 for the benchmarks selected. The results have also shown that the CA-IM_1 and CA-IM_2 perform the optimum search and reach optimum values equally or above the ones reached by algorithms DGA and DGA-SRM that were chosen for comparison. The positive results obtained, give support the idea that this is a desirable approach for tackling highly constrained NP-complete problems such as the MKP. In addition, it is possible that the hybridization of cultural algorithms based on population of GA with local search techniques improves the results obtained by standard CAs. In a future work, a study will be done about the behavior of the sub-populations that are eliminated and recreated randomly. In addition a local search will be implemented to CAs as much for standard CA (single population) as for CA-IM (multi population) so as to verify improvements on these algorithms.

6. Acknowledgments

This research was supported by the CAPES (Coordenação de Aperfeiçoamento do Pessoal de Ensino Superior, Brazil) and by FAPESPA (Fundação de Amparo à Pesquisa do Estado do Pará, Brazil).

7. References

Adeyemo, J.A. (2011). Reservoir Operation using Multi-objective Evolutionary Algorithms-A Review, In: *Asian Journal of Scientific Research*, Vol.4, No. 1, pp.16-27, February 2011, ISSN 19921454.

Agapie, A., Fagarasan, F. & Stanciulescu, B. (1997). A Genetic Algorithm for a Fitting Problem, In: *Nuclear Instruments & Methods in Physics Research Section A*, Vol. 389, No. 1-2, April 1997, pp. 288-292, ISSN 0168-9002.

Aguirre, H. E. & Tanaka, K. (2006). A Model for Parallel Operators in Genetic Algorithms, In: *Springer Book Series Studies in Computational Intelligence, Parallel Evolutionary Computations* , Nedjah, N., Alba, E. & Macedo M., L., pp.3-31, Springer, ISBN 9783540328391, Berlin Heidelberg.

Aguirre, H. E., Tanaka, K., Sugimara, T. & Oshita, S. (2000). Improved Distributed Genetic Algorithm with Cooperative-Competitive Genetic Operators, In: *Proc. IEEE Int.*

Conf. on Systems, Man, and Cybernetics, Vol.5, ISBN 0-7803-6583-6, pp. 3816-3822, Nashville, TN, USA, October 8-11 2000.

Aguirre, H. E., Tanaka, K. & Sugimura, T. (1999). Cooperative Model for Genetic Operators to Improve GAs In: *International Conference on Information Intelligence and Systems*, ISBN 0-7695-0446-9, pp. 98–106, Bethesda, MD, USA, 31 Oct. - 03 Nov., 1999.

Ahmed, Z. & Younas I. (2011). A Dynamic Programming based GA for 0-1 Modified Knapsack Problem, In: *International Journal of Computer Applications*, Vol. 16, No.7, February, 2011, pp. 1–6, ISSN 09758887.

Back, T., Fogel D. B. & Michalewicz Z., (Ed(s).). (1997). *Handbook of Evolutionary Computation*, Oxford University Press, ISBN 0-7503-0392-1, UK.

Beasley, J. E. (1990). Multi- Dimensional Knapsack Problems, In: *OR-library,* Date of Access: September 2011, Available from: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/mknapinfo.html.

Braun, H. (1991). On Solving Traveling Salesman Problems by Genetic Algorithms, In: *Parallel Problem Solving from Nature – Proceedings of 1st Workshop*, Vol. 496 of Lecture Notes in Computer Science, H.P. Schwefel and R. Manner, Vol. 496, pp. 129-133, Springer, ISBN 3-540-54148-9, Dortmund, FRG, October 1-3 1990.

Chu, P. C. & Beasley J. E. (1998). A Genetic Algorithm for the Multidimensional Knapsack Problem, In: *Journal of Heuristics*, vol. 4, no. 1, June 1998, pp. 63–86, ISSN:1381-1231.

Eiben, A. E. & Smith, J.E. (2008). *Introduction to Evolutionary Computing*, Springer, Second edition, ISBN 978-3-540-40184-1, Amsterdam, NL.

Fukunaga, A. S. & Tazoe, S. (2009). Combining Multiple Representations in a Genetic Algorithm for the Multiple Knapsack Problem, In: *IEEE Congress on Evolutionary Computation*, ISBN 978-1-4244-2958-5, pp. 2423 – 2430, Trondheim, 18-21 May, 2009.

Gallardo, J. E., Cotta C. & Fernández A. J. (2007). On the Hybridization of Memetic Algorithms With Branch-and-Bound Techniques, In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B,* Vol. 37, No. 1, February 2007, pp. 77-83, ISSN 1083-4419.

Glover, F. & Kochenberger, G. A. (1996). Critical Event Tabu Search for Multidimensional Knapsack Problems, In: *Meta-Heuristics: Theory and Applications*, Osman, I.H. & Kelly, J.P., pp. 407-427, Springer, ISBN 978-0-7923-9700-7, Boston, USA.

Gunther, R. R. (1998). An Improved Genetic Algorithm for the Multiconstrained 0–1 Knapsack Problem, In: *Evolutionary Computation Proceedings*. *IEEE World Congress on Computational Intelligence*, ISBN 0-7803-4869-9, pp.207-211, Anchorage, AK, May 4-9 1998.

Gustafson, S. & Burke, E.K. (2006). The Speciating Island Model: An Alternative Parallel Evolutionary Algorithm, In: *Journal of Parallel and Distributed Computing*, Vol. 66, No. 8, August 2006, pp. 1025-1036, ISSN 07437315.

Iacoban, R., Reynolds, R. & Brewster, J. (2003). Cultural Swarms: Modeling the Impact of Culture on Social Interaction and Problem Solving, In: *IEEE Swarm Intelligence Symposium*. ISBN 0-7803-7914-4, pp. 205–211, University Place Hotel, Indianapolis, Indiana, USA, April 24-26 2003.

Jin, X., & Reynold, R. G. (1999). Using Knowledge-Based System with Hierarchical Architecture to Guide the Search of Evolutionary Computation, In: *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence,* ISBN 0-7695-0456-6, pp. 29–36, Chicago, Illinois, November 08-10 1999.

Ku, S. & Lee, B. (2001). A Set-Oriented Genetic Algorithm and the Knapsack Problem, In: *Proceedings of the Congress on Evolutionary Computation*, ISBN 0-7803-6657-3, pp. 650–654, Seoul, South Korea, May 27-30 2001.

Kumar, R. & Singh, P. K. (2010). Assessing Solution Quality of Biobjective 0-1 Knapsack Problem using Evolutionary and Heuristic Algorithms, In: *Applied Soft Computing*, Vol. 10, No 3, June 2010, pp. 711 – 718, ISSN 1568-4946.

Martello, S. & Toth, P. (1990). *Algorithms and Computer Implementations*, John Wiley & Sons, ISBN 0471924202, New York.

Nowostawski , M. & Poli , R. (1999). Parallel Genetic Algorithm Taxonomy, In: *Proceedings of the Third International conference on knowledge-based intelligent information engineering systems*, ISBN 0780355784, pp. 88-92, Adelaide, August 1999.

Reynolds, R. G., & Peng, B. (2004). Cultural Algorithms: Computational Modeling of How Cultures Learn to Solve Problems, In: *Seventeenth European Meeting on Cybernetics and Systems Research*, ISBN 3-85206-169-5, Vienna, Austria, April 13-16 2004.

Reynolds, R. G. & Chung C. (1996). The Use of Cultural Algorithms to Evolve Multiagent Cooperation, In: *Proc. Micro-Robot World Cup Soccer Tournament*, pp. 53–56. Taejon, Korea, 1996.

Reynolds, R. G. (1994), An Introduction to Cultural Algorithms, In: *Proceedings of the Third Annual Conference on Evolutionary Programming,* ISBN 9810218109, pp. 131-139, San Diego, California, February 24-26 1994.

Reynolds, R. G. (1999). Chapter Twenty-Four; Cultural Algorithms: Theory and Applications, In: *New Ideas in Optimization*, Corne, D., Dorigo, M. & Glover F., pp. 367-377, McGraw-Hill Ltd., ISBN 0-07-709506-5, UK, England.

Silva, D. J. A. & Oliveira R. C. L. (2009). A Multipopulation Cultural Algorithm Based on Genetic Algorithm for the MKP, In: *Proc. of the 11th Annual conference on Genetic and evolutionary computation,* ISBN 978-1-60558-325-9, pp. 1815-1816, Montreal, Québec, Canada, July 8-12 2009.

Sivanandam, S.N. & Deepa, S. N. (2007). *Introduction to Genetic Algorithms*, (1st), Springer, ISBN 978-3-540-73189-4, New York.

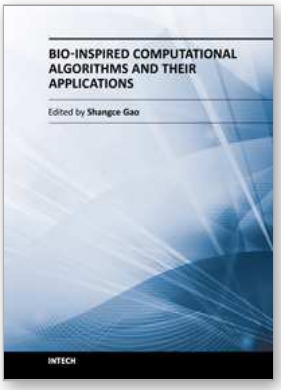
Sivaraj, R. & Ravichandran,T. (2011). An Improved Clustering Based Genetic Algorithm for Solving Complex NP Problems, In: *Journal of Computer Science,* Vol. 7, No. 7, May 2011, pp. 1033-1037, ISSN 15493636.

Spillman, R. (1995). Solving Large Knapsack Problems with a Genetic Algorithm, In: *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 1, ISBN 0-7803-2559-1, pp 632 -637, Vancouver, BC, Canada, October 22-25 1995.

Tavares, J., Pereira, F. B. & Costa, E. (2008). Multidimensional Knapsack Problem: A Fitness Landscape Analysis, In: *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, Vol. 38, No. 3, June 2008, pp.604-616, ISSN 1083-4419.

Tomassini, Marco (2005). *Spatially Structured Evolutionary Algorithms: Artificial Evolution, Space and Tim - Natural Computing Series* (1st), Springer, New York, Inc., ISBN 3540241930, Secaucus, NJ, USA.

Zoheir, E. (2002). Solving the 0/1 knapsack Problem Using an Adaptive Genetic Algorithm, In: *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol.16, No. 1, January 2002, pp.23-30, ISSN 08900604.



Bio-Inspired Computational Algorithms and Their Applications

Edited by Dr. Shangce Gao

ISBN 978-953-51-0214-4
Hard cover, 420 pages
Publisher InTech
Published online 07, March, 2012
Published in print edition March, 2012

Bio-inspired computational algorithms are always hot research topics in artificial intelligence communities. Biology is a bewildering source of inspiration for the design of intelligent artifacts that are capable of efficient and autonomous operation in unknown and changing environments. It is difficult to resist the fascination of creating artifacts that display elements of lifelike intelligence, thus needing techniques for control, optimization, prediction, security, design, and so on. Bio-Inspired Computational Algorithms and Their Applications is a compendium that addresses this need. It integrates contrasting techniques of genetic algorithms, artificial immune systems, particle swarm optimization, and hybrid models to solve many real-world problems. The works presented in this book give insights into the creation of innovative improvements over algorithm performance, potential applications on various practical tasks, and combination of different techniques. The book provides a reference to researchers, practitioners, and students in both artificial intelligence and engineering communities, forming a foundation for the development of the field.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821