

BPSO Versions with Chi-squared Distribution for MKP Resolution

Raida KTARI^{1,2}, Habib CHABCHOUB¹

¹ Sfax University, L.O.G.I.Q, Tunisia

² Aix-Marseille Université, CNRS LIF UMR 7279, France

(E-mail : raidaktari@yahoo.com, habib.chabchoub@fsegs.rnu.tn)

Abstract. This paper deals with the application of BPSO (Binary Particle Swarm Optimization), EPSO (Essential Particle Swarm Optimization) and EPSOq (Essential Particle Swarm Optimization queen) to the Multidimensional Knapsack Problem (MKP) which is well-known to be NP-hard Combinatorial Optimization problem. The particularity of this paper consists in proposing a novel random number generator based on the Chi-squared distribution for the particles' positions and velocities in the initialization step. A repair operator is also utilized to change an unfeasible solution to a feasible one. The performance assessment of the BPSO, EPSO and EPSOq is a critical point in this paper. That is why, we experiment these approaches on a variety of MKP instances from OR-Library. Then, we compare our results with those of other previous works and with the best known results in the literature.

Keywords: MKP, PSO, Chi-squared distribution, randomness, repair operator, performance.

1 Introduction

Particle Swarm Optimization (PSO) is one of the evolutionary optimization methods inspired by nature which include evolutionary strategy (ES), evolutionary programming (EP), genetic algorithm (GA), and genetic programming (GP). PSO was originally designed and introduced by Eberhart and Kennedy [7], [9] in 1995. It is a population based search algorithm based on the simulation of the social behaviour of birds, bees or a school of fishes.

The PSO algorithm is based on the exchange of information between individuals, so called particles, of the population, so called swarm. Indeed, each particle adjusts its own position towards its previous experience and towards the best previous position obtained in the swarm. Memorizing its best own position establishes the particle's experience implying a local search along with global search emerging from the neighboring experience or the experience of the whole swarm.

PSO has gained widespread appeal amongst researchers and has been shown to offer good performance and efficiency in a variety of application domains such as power and voltage control (Abido [1]), mass-spring system (Brandstatter and Baumgartner [4]), and task assignment (Salman et al. [13]). The comprehensive survey of the PSO algorithms and applications can be found in Kennedy and Eberhart [11].

Particle swarm optimization was first introduced as an optimization method for solving continuous problem. Later, Kennedy and Eberhart [10], proposed a

binary version of PSO (BPSO) to accommodate discrete binary variables and allow it to operate in a binary problem space. A particle moves in a search space restricted to 0 or 1 on each dimension.

Since the performance ability of BPSO is not good enough, several improved versions of BPSO have been proposed to modify the velocity vector when binary variables are involved. Recently, Chen et al. [5] proposed a modified version of BPSO, so called Essential Particle Swarm Optimization (EPSO). In this research work, authors dismantle the BPSO algorithm qualitatively, breaking it into its essential components and then reinterpreting it in another ways as new program. After, as pheromone array in Ant Colony Optimization (Alaya et al. [2]), they introduce the queen informant particle in EPSO. They identify this implementation of this idea with the acronym EPSOq that is considered as another improved version of BPSO.

This paper deals with the application of BPSO, EPSO and EPSOq in the field of Combinatorial Optimization (CO) problems, which is a quite rare field tackled by PSO. The constrained problem discussed in this paper is the well-known to be NP-hard CO problem. The 0-1 multidimensional knapsack problem (MKP), which consists in selecting a subset of n given objects (or items) in such a way that the total profit of the selected objects is maximized while a set of knapsack constraints are satisfied. More formally, the MKP01 can be stated as follows:

$$\begin{aligned}
 \text{(MKP)} \quad & \left\{ \begin{array}{l} \text{Max } \sum_{j=1}^n c_j x_j \quad (1) \\ \text{s.s } \sum_{j=1}^n a_{ij} x_j \leq b_i \quad ; \quad \forall i \in M = \{1, \dots, m\} \quad (2) \\ x_j \in \{0, 1\} \quad \forall j \in N = \{1, \dots, n\} \quad (3) \end{array} \right.
 \end{aligned}$$

Equation (1) describes the objective function for the MKP. Each of the m constraints described in condition (2) is called a knapsack constraint, so the MKP is also called the m -dimensional knapsack problem. Let $M = \{1, 2, \dots, m\}$ and $N = \{1, 2, \dots, n\}$, with $b_i > 0$ for all $i \in M$ and $a_{ij} \geq 0$ for all $i \in M, j \in N$, a well-stated MKP assumes that $c_j > 0$ and $a_{ij} \leq b_i < \sum_{j=1}^n a_{ij}$ for all $i \in M, j \in N$.

MKP is one of the most intensively studied discrete programming problems, mainly because its simple structure which can be seen as a general model for any kind of binary problems with positive coefficients.

In this paper, we propose an algorithm which applies the BPSO, EPSO and EPSOq approaches for solving the 0-1MKP. This is achieved by using a generator based on the Chi-squared distribution in order to generate particles well-varied in the research space during the initialization phase.

This algorithm has also the advantage of using a repair operator based on the pseudo-utility ratios derived from the surrogate duality approach to guarantee generating feasible solutions.

The performance assessment of the BPSO, EPSO and EPSOq is a critical point in this paper. That is why, we experiment these approaches on a very large variety of bigger size MKP01 instances from OR-Library, which are considered

to be rather difficult for optimization approaches. Then, we compare our results with the best known ones in the literature (Angelelli et al. [3], Chu and Beasley [6], Vasquez and Vimont [14]).

2 BPSO Methods for MKP Resolution

Since we have found any literature concerning neither the EPSO nor EPSOq algorithm applied to the 0-1 MKP problems, we select some bigger size MKP instances from OR-Library to assess their performance, and we compare their results not only with that of the BPSO but also with the best known profits in the literature. To make a fair comparison, we set the same ring topology as the neighborhood structure with number of neighbors set to 2 for the three PSO improved versions. The parameters given in (Chen et al. [5]) are regarded as optimal for our algorithm. That is why; we conserve their values even for BPSO. We also use the same initialization generator for the particles position and the same repair method for the unfeasible solutions. These methods are described as follows:

Initialization method: Monte Carlo methods use the computer together with the generation of random numbers and mathematical models to generate statistical results that be able to simulate and experiment with the behavior of various business, engineering and scientific systems. Monte Carlo simulations usually use the application of random numbers that are uniformly distributed over the interval $[0, 1]$. These uniformly distributed random numbers are employed in order to obtain stochastic variables from various probability distributions. These stochastic variables can then be useful to approximate the behavior of worthwhile system variables.

Indeed, most computer languages have form of random number generator that generates uniformly distributed random numbers between 0 and 1. Almost these random number generators use modulo-arithmetic so as to generate numbers that appear to be uniformly distributed. As a result, the random number generators are called pseudorandom number generators since they are not really random. They only simulate the behavior of a uniformly distributed random number on the interval $[0, 1]$.

As an example, we apply in the present paper the chi-square goodness of fit test to the random number generator associated with JAVA our computer programming language in order to initialize the particles positions. We try to build a computer program to generate, for instance, 1000 random numbers between 0 and 1. We can then divided the interval 0 to 1 into 10 classes using the intervals $(0, .1)$, $(.1, .2)$, ..., $(.9, 1.0)$ and then we sort the 1000 random numbers to precise the number in each class. These values are the observed frequencies designed by the experiment. If the pseudorandom number generator is truly uniform, then the theoretical frequency associated with each class would have a value of 100.

So, for each position bit, a number will be generated thanks to this approach and consequently, it will be rounded either 0 or 1. This method would be very helpful in creating a swarm whose particles are well-scattered in the search

space and even quick not only for the particles positions initialization step but also for the particles velocities start up.

Reparation method: Usually, in PSO, a given particle is dynamically attracted by the social and the cognitive components. Therefore, during evolution towards the best global solution, the algorithm can pass through regions of unfeasible solutions.

Indeed, an unfeasible particle now can be changed to a feasible one later. A particle representing an unfeasible solution is allowed to exist in the swarm. PSO usually makes use of penalty function technique in order to reduce the constrained problem to an unconstrained problem by imposing a penalty to the fitness of such particle (Hu and Eberhart [8]).

Instead of this methodology, we incorporate the heuristic repair operator suggested in (Kong and Tian [12]) specially designed for MKP. The repair operator utilizes the notion of the pseudo-utility ratios u_j that is explained in the following equation:

$$u_j = \frac{c_j}{\sum_{i=1}^m w_i a_{ij}} \quad (4)$$

where $w=(w_1, w_2, \dots, w_m)$ is a set of surrogate multipliers (or weights) of some positive real numbers. To obtain reasonably good surrogate weights we can solve the LP relaxation of the original MKP and utilize the values of the dual variables as the weights. Otherwise, w_i is set equal to the shadow price of the i^{th} constraint in the LP relaxation of the MKP.

A brief description behind this reparation method is given as follows: The first phase, which is called DROP phase, changes the value of each bit of the solution from one to zero in increasing order of u_j if feasibility is violated. The second phase, so called ADD phase, reverses the process by changing each bit from zero to one in decreasing order of u_j as long as feasibility is not violated. Adopting this procedure, we are able to solve the problem of an unfeasible particle existence in the search space.

3 Computational Results and Discussions

Combining all the ideas described above, we aggregate the BPSO, EPSO and EPSOq in a unique algorithm applied on a variety large MKP instances with maximum number of cycles of n , where n is the objects number. Indeed, we concentrated in MKP instances with $m \in \{5, 10, 30\}$ constraints, $n \in \{100, 250, 500\}$ variables and $\alpha \in \{0.25, 0.5, 0.75\}$ tightness ratios. Experiments were performed in an Intel® Core™ i5 with 4 Gb of RAM and 2.67 GHz CPU.

The following table reports just the test results of some instances.

The first column indicates the instance name, the second column is the best-known solutions and the next 3 columns record the fitness average and its CPU time average in seconds of BPSO, EPSO and EPSOq respectively over 30 runs.

Instance	Best-known	BPSO		EPSO		EPSOq	
		Fitness	CPU time	Fitness	CPU time	Fitness	CPU time
OR5x100-0.25_1	24381	21774	0.012	24071.5	0.0033	24381	0.0029
OR5x100-0.50_2	42545	41870.5	0.15	42320	0.009	42545.5	0.0099
OR5x100-0.75_3	59802	44294	0.67	59494.5	0.094	58889	0.093
OR10x250-0.25_4	61000	57550	9	60369	0.53	60662.5	0.52
OR10x250-0.50_5	108485	106255	6.23	107314	0.792	107314	0.69
OR10x250-0.75_6	152109	151619	4.012	151417.5	0.871	152109	0.792
OR30x500-0.25_7	114181	109086	327.004	112112	11.91	113895	11.021
OR30x500-0.50_8	216542	212662.5	221.95	212088.5	8.58	214150.5	7.95
OR30x500-0.75_9	303199	298667	123.01	300267	7.023	301447.5	6.4

Table.1: Comparison of results obtained by BPSO, EPSO, EPSOq and the optimal known solutions

From this table and for all the 270 instances of MKP that we tested and which are considered to be rather difficult for optimization approaches, Our experiments show that BPSO, EPSO and EPSOq are able to find good solutions. But, it is obvious that EPSOq outperforms BPSO and even EPSO with better solution quality, and with quick convergence to satisfied solution, as the size of the problem increases.

In fact, incorporating the queen informant in the EPSO doesn't increase the number of function evaluations because it was added just a new informer that only offers information to the other particles. Moreover, the initialization generator based on the Chi-squared distribution achieves a higher exploration of solutions at the start of algorithm and the repair operator plays a critical role in a higher exploitation near the global optimum solutions at the end of algorithm.

We also compare the EPSOq fitness with the best known solutions in the literature. Indeed, the average performance of EPSOq for the 270 instances considered was 1.028% of the known optimum. This reveals that PSO can perform well for this class of combinatorial problem, even for large instances. That is, EPSOq seems to be efficient in navigating the hyper-surface of the search space and finding good solutions (and, sometimes, the best solution). It is also as competitive as the existing discrete PSO approaches for the MKP thanks to the initialization generator, the repair operator and the queen informant.

4 Conclusions

This paper presents the application of the BPSO, EPSO and EPSOq on the 0-1 multidimensional knapsack problem.

The particles positions are initialized using a random number generator based on the Chi-squared distribution in the aim to obtain a swarm well-varied. To tackle

the problem of the movement to regions at very large distances from the main swarm or even outside the search space, we implement the reparation method using the notion of the pseudo-utility ratios. So as to evaluate the effectiveness and viability of these discrete binary PSO versions, we take many large MKP instances from OR-Library to test its performance. Indeed, its simulation results are compared with the best known solutions. The EPSOq results were close and sometimes equal to the optimal solution known, even considering that the parameters were not optimized. These experiments were not optimal. Little effort was taken to find the best parameters. In future versions, we will do our best in these aspects. Eventually, the EPSOq is seemed as a new path for building a fast and easy discrete PSO by its smart representation.

References

- 1.M. A. Abido. Optimal power flow using particle swarm optimization. *Electrical Power and Energy Systems*, 24, 563-571, 2002.
- 2.I. Alaya, C. Solnon and K. Ghédira. Optimisation par colonies de fourmis pour le problème du sac à dos multidimensionnel. *TSI-Technique et Science Informatiques*, 45-60, 2007.
- 3.E. Angelelli, M. G. Speranza and M. W. P. Savelsbergh. Competitive analysis for dynamic multiperiod uncapacitated routing problems. *Networks*, 49, 4, 308–317, 2007.
- 4.B. Brandstatter and U. Baumgartner. Particle Swarm Optimization—Mass-Spring System Analogon. *IEEE Transactions on Magnetics*, 38, 997-1000, 2002.
- 5.E. Chen, J. Li and X. Liuc. In search of the essential binary discrete particle swarm. *Applied Soft Computing*, 11, 3260–3269, 2011.
- 6.P. Chu and J. Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4, 63-86, 1998.
- 7.R. C. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, 39-43, 1995.
- 8.X. Hu and R. Eberhart. Solving Constrained Nonlinear Optimization Problems with Particle Swarm Optimization. *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics*. 2002.
- 9.J. Kennedy and R.C. Eberhart. Particle Swarm Optimisation. *Proceedings of the IEEE International Conference*, 4, 1942- 1948, 1995.
- 10.J. Kennedy and R.C. Eberhart. A discrete binary version of the particle swarm algorithm. *IEEE International Conference on Systems, Man, and Cybernetics*, 5, 4104-4109, 1997.
- 11.J. Kennedy and R. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 2001.
- 12.M. Kong and P. Tian. Apply the Particle Swarm Optimization to the multidimensional knapsack. *L. ICAISC 2006*, LNAI 4029, 1140–1149, 2006.
- 13.A. Salman, I. Ahmad and S. Al-Madani. Particle Swarm Optimization for Task Assignment Problem. *Microprocessors and Microsystems*, 26, 363-371, 2002.
- 14.M. Vasquez and Y. Vimont. Improved results on the 0-1 multidimensional knapsack problem. *European Journal of Operational Research*, 165, 70–81, 2005.