Stochastic Diffusion Binary Differential Evolution to Solve Multidimensional Knapsack Problem

Ayed A. Salman, Imtiaz Ahmad, and Mahmad G. H. Omran

***Abstract*—Multi Knapsack Problem (MKP) is NP-hard combinational optimization problem, also known as the multi-constraint knapsack problem. MKP is one of the most studied problems in combinatorial optimization, with variety of real-life applications. In this paper a Stochastic Diffusion Binary differential evolution (SD-BDE) algorithm is applied for optimizing the Multidimensional Knapsack Problem (MKP). SD-BDE, is a Binary version of Differential Evolution hybridized with ideas extracted from Stochastic Diffusion search. SD-BDE algorithm, in this paper, is compared against state-of-the-art existing algorithms in solving MKP. Experimental results show that the SD-BDE algorithm outperformed the existing algorithms by finding either better or at least similar solutions for all tested benchmarks**

***Index Terms*—Differential evolution, stochastic diffusion search, np-complete problem, multidimensional knapsack problem.**

I. Introduction

Differential Evolution (DE) is an evolutionary algorithm that has been proposed by Storn and Price [1], [2]. DE is similar to other evolutionary algorithms in that a population of individuals is used to search for an optimal solution. However, the main difference between traditional evolutionary algorithms and DE is that, in traditional evolutionary algorithms, mutation results in small random perturbations to the genes of an individual while in standard DE the mutation is an arithmetic combination of individuals [3]. DE was proposed to work on real-valued domain problems, however several researchers extended it to discrete and binary domains such as those proposed by Gong *et al*. [4], Tasgeiren *et al*. [5] and Wang *et al*. [6].

On other hand, Stochastic Diffusion Search (SDS) [7] is a population-based, naturally inspired search and optimization algorithm. It belongs to a family of swarm intelligence (SI) methods. SDS is based on direct (one-to-one) communication between agents. SDS has been successfully applied to a wide range of optimization problems. Omran and Salman [8] proposed a probabilistic Stochastic Diffusion search to tackle continuous optimization problems

The meta-heuristic Stochastic Diffusion Binary Differential Evolution (i.e. SD-BDE) was proposed by Salman [9] with an idea of hybridizing binary differential evolution with stochastic diffusion. Authors used ideas from all Gong *et al*. [4], Yuan *et al.* in [10], and [8] to build the core algorithm of SD-BDE: The concept of the diffusion for the characteristics of an “active agent” was used to enhance the overall quality of the population by increasing the chance of exposing individuals to DE operators only if these individuals are not “active” (aka was declared in-competent compared to a randomly selected competitor). Updating the non-active individual will push the whole population toward areas of better quality. Moreover, only those individuals that were randomly-selected in the mutation step as the target vector, and happen to be an active individual, have the chance to influence the mutation step, otherwise, the mutation is random.

Salman [9] shows the superiority of such algorithm onto different well-crafted benchmark binary-valued problems as well as satellite broadcasting scheduling problem which is of a real-world importance. As a continuation, this work we further studied the performance of SD-BDE to solve MKP. Results presented here show that SD-BDE algorithm is a very promising optimization method for solving binary problems. The remainder of the paper is organized in the following manner: Section II presents a detailed overview about SD-BDE. Experimental results comparing to similar, state-of-the-art methods onto different MKP benchmark problems and are reported in Section III. Finally, Section IV concludes the paper.

II. Stochastic Diffusion Binary DE

DE uses the difference between randomly selected vectors (individuals) as the source of variation for a third vector, called the target vector. Trial solutions are generated by adding a weighted difference vector to the target vector. This process is referred to as the mutation operator where the target vector is mutated. A recombination, or crossover step, is then applied to produce an offspring, which is only accepted if it improves the fitness of the parent individual.

The binary DE algorithms are described in more details below, in terms of the three evolution operators: mutation, crossover, and selection:

A. Mutation

For each parent $x_i(t)$ of generation t, a trial vector $v_i(t)$ is created by mutating a target vector. Randomly select the target vector $x_{i3}(t)$, with $i \neq i_3$. Then, the two individuals $x_{i1}(t)$ and $x_{i2}(t)$ are randomly selected with $i_1 \neq i_2 \neq i_3 \neq i$, and the difference vector $x_{i1}(t)$ - $x_{i2}(t)$, is calculated. The trial vector is then calculated as

Manuscript received December 22, 2015; revised March 15, 2016. This work was supported by Kuwait University Research Grant No. [EO 03/09].

Ayed A. Salman and Imtiaz Ahmad are with the Computer Engineering Department, Kuwait University, Kuwait (e-mail: ayed.salman@ku.edu.kw, imtiaz.ahmad@ku.edu.kw).

Mahmad G. H. Omran is with the Department of Computer Science, Gulf University for Science and Technology, Kuwait (e-mail: omran.m@gust.edu.kw).

doi: 10.18178/ijmlc.2016.6.2.586

$$v_i(t) = x_{i3}(t) + F(x_{i1}(t)\ \ x_{i2}(t)) \quad (1)$$

where the term $F(x_{i1}(t) - x_{i2}(t))$ represents the mutation step size, and F is a scale factor used to control the amplification of the differential variation. Note that $F \in (0; \infty)$. For a binary DE, Gong *et al*. [4] considered each binary decision variable as a single dimension. Thus, the distance between two individuals for each dimension, $D_j(x_{i1,j}; x_{i2,j})$, is either 0 or 1. Eq. 1 is replaced with the following equation:

$$v_{ij} = \begin{cases} 1 - x_{i3,j^{(t)}} & \text{if } rand(0,1) < F \text{ and} \\ & D_j(x_{i1,j^{(t)}}, x_{i2,j^{(t)}}) = 1 \\ x_{i3,j^{(t)}} & \text{otherwise.} \end{cases} \quad (2)$$

where, rand (0; 1) is a uniform distribution random number between [0, 1]. In Eq. 2, for each dimension j, F X Dj ($x_{i1;\,j}(t)$; $x_{i2;\,j}(t)$) is used to decide the probability that a change is applied to x_{i1} in the corresponding dimension to produce v_i.

B. Crossover

DE follow a discrete recombination approach where elements from the parent vector $x_i(t)$ are combined with elements from the trial vector $v_i(t)$ to produce the offspring, $\mu_i(t)$. Using the binomial crossover,

$$m_i(t) = \begin{cases} v_i(t) & \text{if } rand(0;\,1) < CR \\ x_i(t) & \text{otherwise:} \end{cases} \quad (3)$$

where CR is the probability of reproduction (with $CR \in [0; 1]$). Thus, each offspring is a stochastic linear combination of three randomly chosen individuals when rand (0; 1) < CR; otherwise the offspring is inherited directly from the parent.

C. Selection

The offspring $m_i(t)$ replaces the parent $x_i(t)$ if and only if the fitness of the offspring is better than that of the parent.

The concept of the diffusion for the characteristics of an "active agent" is used in SD-BDE to enhance the over-all quality of the population by increasing the chance of exposing individuals to DE operators only if these individuals are not "active". Updating the non-active individual is pushing the whole population toward areas of better quality. Moreover, only those individuals that were randomly-selected in the mutation step as the target vector, and happen to be an active individual, have the chance to influence the mutation step, otherwise, the mutation is random. Finally, this algorithm pushes toward updating only non-active individuals in the DE, which leaves active members untouched. This will create a problem when the population converges onto similar individuals; those individuals will have similar status (all active). If left the same, the algorithm will have no way to evolve further. To avoid such situation, the concept of probability of changes for active individuals (P_c) is introduced to force some very limited randomized changes even if individuals are active. The theory behind Stochastic Diffusion Search is well explained and discussed by Nasuto *et al*. [11] and Nasuto and Bishop [7].

```
Repeat
  For each individual xi in the population do
      Randomly choose an agent, (xr) from the
      swarm;
        if f (xi) <= f (xr) then activei = true
        else
          activei = false;
        end
  end
  for each individual xi in the population do if activei
     = false or rand(0; 1) < Pc then
        Let i1, i2, and i3 be three random integers in
         [1, s] ;
         for j=1 to Nd do
              if activei3 = true; then
                 if rand(0; 1) < F and

                 D j(xi1; j; xi2; j)= 1 then

                 vi; j = 1- xi3;j;
                 else
                   vi; j = xi3; j;
                 end else
               vi; j = randomly selected value ∈
                 {0; 1} ;
              end
              if rand(0,1)< CR then
                μi; j = vi; j;
              else
                μi; j = xi; j;
              end else
          if f (μi) < f (xi) then
              xi = μi ;
          end
   end
until the terminating condition is achieved;
```

Algorithm 1: Pseudo-code for the proposed stochastic diffusion binary DE algorithm (SD-BDE).

TABLE I: RESULTS FOR COMPARING SD-BDE TO ALGORITHMS REPORTED IN [12] ON EASY-SET OF MKP PROBLEM; RESULTS IN BOLD ARE STATISTICALLY SIGNIFICANT OVER OTHERS USING MANN-WHITNEY TEST

Benchmark	Best Known	Algorithm	Best Fitness	Success Rate	Average Fitness
Pet1 m=10 n=6	3800	KBPSO	3800	100%	3800
		MBPSO	3800	100%	3800
		PBPSO	3800	100%	3800
		DE	3800	100%	3800
		SD-BDE	3800	100%	3800
Pet2 m=10 n=10	8706.1	KBPSO	8706.1	100%	8706.1
		MBPSO	8706.1	100%	8706.1
		PBPSO	8706.1	100%	8706.1
		DE	8706.1	95%	8700.51
		SD-BDE	8706.1	100%	8706.1
Pet3 m=10 n=15	4015	KBPSO	4015	100%	4015
		MBPSO	4015	100%	4015
		PBPSO	4015	100%	4015
		DE	4015	70%	4006.00
		SD-BDE	**4015**	**100%**	**4015**

Pb4 *m*=02 *n*=29	95168	KBPSO	95168	20%	91879.15
		MBPSO	95168	15%	92419
		PBPSO	95168	40%	93114.1
		DE	**95168**	**90%**	**95089.65**
		SD-BDE	**95168**	**90%**	**95147.70**
Pb5 *m*=10 *n*=20	2139	KBPSO	2139	65%	2131.1
		MBPSO	2139	5%	2110.9
		PBPSO	2139	75%	2134.45
		DE	2139	50%	2119.55
		SD-BDE	2139	60%	2130.35
Pb6 *m*=30 *n*=40	776	KBPSO	776	10%	746.95
		MBPSO	776	10%	708.60
		PBPSO	776	15%	752.85
		DE	765	0%	734.60
		SD-BDE	**776**	**30%**	**764.55**

TABLE II: RESULTS FOR COMPARING SD-BDE TO ALGORITHMS REPORTED IN [12] ON COMPLEX-SET OF MKP PROBLEM; RESULTS IN BOLD ARE STATISTICAL SIGNIFICANT OVER OTHERS USING MANN-WHITNEY TEST

Benchmark	Best Known	Algorithm	Best Fitness	Success Rate	Average Fitness
Sent1 *m*=30 *n*=60	7772	KBPSO	7676	0%	7562.4
		MBPSO	7762	0%	7683.55
		PBPSO	7772	5%	7695.9
		DE	7772	10%	7716.85
		SD-BDE	**7772**	**40%**	**7765.25**
Sent2 *m*=30 *n*=60	8722	KBPSO	8655	0%	8603.5
		MBPSO	8711	0%	8651
		PBPSO	8722	5%	8671.1
		DE	8721	0%	8709.95
		SD-BDE	**8722**	**55%**	**8721.05**
Weish12 *m*=5 *n*=50	6339	KBPSO	6339	15%	6295.1
		MBPSO	6339	35%	6317.05
		PBPSO	6339	45%	6331.75
		DE	6339	80%	6329.35
		SD-BDE	**6339**	**100%**	**6339.00**
Weish20 *m*=5 *n*=70	9450	KBPSO	9146	0%	9092.05
		MBPSO	9445	0%	9352.95
		PBPSO	9450	5%	9362.05
		DE	9450	70%	9442.75
		SD-BDE	**9450**	**100%**	**9450.00**

TABLE III: AVERAGE EXECUTION TIME (WITH STANDARD DEVIATION) FOR SD-BDE AND STANDARD BINARY DE FOR MKP PROBLEM

Benchmark	SD-BDE	DE
Pet1	2.769(0.08)	1.51(0.02)
Pet2	2.43(1.05)	1.61(0.01)
Pet3	2.88(0.10)	1.69(0.02)
Pb4	1.61(0.05)	1.57(0.09)
Pb5	2.96(0.073)	2.86(0.27)
Pb6	7.66 (0.20)	7.09(0.27)
Sent1	32.66 (6.62)	34.25(9.82)
Sent2	29.49 (8.99)	30.19(3.71)
Weish12	7.47 (1.42)	7.74(1.74)
Weish20	10.86(1.92)	9.69(2.26)

III. EXPERIMENTAL RESULTS

A. Experiments Setup

All proposed methods are implemented using MATLAB. All tests are run on a PC with Intel Core Due 2 processor running at 2.20 GHz with 3GB of RAM. For all bench-marks, (unless stated otherwise in the following subsections), SD-BDE algorithm parameters were set as follows: The population size s = 20, F = 0:05, P_c = 0:05, and CR = 0:7. The meta-heuristic Stochastic Diffusion Binary Differential Evolution (i.e. SD-BDE) were set as follow, for easy set shown in Table I, all algorithms were run for a maximum of 9000 FE, while for the complex set shown in Table II, they were run for a maximum of 24000 FE. Table I shows results for easy-set MKP. Results show that SD-BDE managed to either beat or produce similar optimal results like others techniques. In particular, SD-BDE had always the same or better success rate than others. Moreover, Table II shows results for the complex-set MKP. SD-BDE was clearly better than others by far on all aspects: best-fitness, success rate, and average fitness. Those results indicates the superiority of SD-BDE over other binary-coded similar algorithms. On the other hand, comparing SD-BDE with original BDE shows that using SD-BDE has an advantage in this problem as well as the previous problem (i.e. SBS).

Looking at the success rate shown in Tables I and II one can find that SD-BDE by far is able to find optimal solutions much frequent than original BDE. Table III shows that the time overhead needed by SD-BDE is very accept-able and sometimes negligible.

IV. Conclusion

This paper present a solution for the Multidimensional Knapsack Problem using a newly developed algorithm (aka, SD-BDE). The algorithm is very effective compared to previous well-known state of the art algorithms pro-posed in the literature. Experimental results show that SD-BDE is robust and able to find optimal solution with reasonable computational time. Furthermore, the SD-BDE algorithm outperformed or at least obtained similar solutions found by previously proposed algorithms.

References

[1] R. Storn and K. Price, "Differential Evolution — A simple and efficient adaptive scheme for global optimization over continuous spaces," Technical Report Tr 95-012, International Computer Science Institute, Berkeley, Ca, 1995.

[2] R. M. Storn and K. V. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341-359, 1997.

[3] V. Feoktistov and S. Janaqi, "Generalization of the strategies in differential evolution," in *Proc. The 18th International Parallel and Distributed Processing Symposium*, 2004, pp. 165-170.

[4] T. Gong and A. Tuson, "Differential evolution for binary encoding," *Adv Soft Computing*, vol. 39, pp. 251-262, 2007.

[5] M. F. Tasgeiren, P. N. Suganth, and Q. K. Pan, "An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem," *Applied Mathematics and Computation*, vol. 215, issue 9, pp. 3356-3368, 2010.

[6] L. Wang, Q. Pan, P. Suganthan, W. Wang, and Y. Wang, "A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems," *Computers & Operations Research*, vol. 37, issue 3, pp. 509-520, 2010.

[7] S. J. Nasuto and J. M. Bishop. "Convergence analysis of stochastic diffusion search," *Journal of Parallel Algorithms and Applications*, vol. 14, pp. 89-107, 1999.

[8] M. G. Omran and A. A. Salman, "Probabilistic stochastic diffusion search," *Swarm Intelligence, Lecture Notes in Computer Science*, vol. 7461, pp. 300-307, 2012.

[9] A. Salman, I. Ahmad, and M. G. H. Omran, "A metaheuristic algorithm to solve satellite broadcast scheduling problem," *Information Sciences Journal*, vol. 322, pp. 72-91, 2015.

[10] X. Yuan, A. Su, H. Nie, Y. Yuan, and L. Wang, "Application of enhanced discrete differential evolution approach to unit commitment problem," *Energy Conversion and Management*, vol. 50, pp. 2449-2456, 2009.

[11] S. J. Nasuto, J. M. Bishop, and L. Lauria, "Time complexity of stochastic diffusion search," *Neural Computation*, vol. 98, Vienna, Austria, 1998.

[12] L. Wang, X. Wang, J. Fu, and L. Zhen, "A novel probability binary particle swarm optimization algorithm and its application," *Journal of Software*, vol. 3, issue 9, p. 28, 2008.

Ayed A. Salman was born in September 30, 1970. He got his bachelor in computer engineering from Kuwait University in January 1994, and his master and PhD in computer science from Syracuse University, NY USA, in December 1996 and September 1999 respectively. He specialized in application and theory of Meta-Heuristic algorithms' research. He has 70+ publication in different internationally refereed journals and conferences. He is currently an associate professor in the Department of Computer Engineering at Kuwait University, Kuwait and occupying the vice-dean for planning and development office since March 2013.

Imtiaz Ahmad has a B.Eng in electrical engineering, from University of Engineering & Technology, Lahore, Pakistan in 1984, M.Eng in electrical engineering from King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia in 1988, and Ph.D. in computer engineering from Syracuse University, Syracuse, New York, USA, in 1992. His research interests is in design automation of digital systems, parallel and distributed computing. He is currently a professor of the Computer Engineering Department at Kuwait University, Kuwait since 1994.

Mahmed Omran received his B.Sc. and M.Sc. degrees with honors in computer engineering from Kuwait University, State of Kuwait, in 1998 and 2000, respectively. He completed his Ph.D. in the Department of Computer Science at University of Pretoria, South Africa, in April 2005. His current research interest is in the area of meta-heuristics and operational research. In addition, he is interested in the area of data clustering, unsupervised image classification and color image quantization. Dr. Omran has more than 40 publications in these fields. Omran is currently an associate professor in GUST University, Kuwait.