Solving Biobjective Set Covering Problem Using Binary Cat Swarm Optimization Algorithm

Broderick Crawford[1,2,3], Ricardo Soto[1,4,5], Hugo Caballero[1(✉)], Eduardo Olguín[3], and Sanjay Misra[6]

[1] Pontificia Universidad Católica de Valparaíso, 2362807 Valparaíso, Chile
hcaballec@gmail.com
[2] Universidad Central de Chile, 8370178 Santiago, Chile
[3] Facultad de Ingeniera y Tecnología, Universidad San Sebastián, Bellavista 7, 8420524 Santiago, Chile
[4] Universidad Autónoma de Chile, 7500138 Santiago, Chile
[5] Universidad Científica del Sur, Lima 18, Peru
[6] Covenant University, Ogun 110001, Nigeria

Abstract. The set cover problem is a classical question in combinatorics, computer science and complexity theory. It is one of Karp's 21 NP-complete problems shown to be NP-complete in 1972. Several algorithms have been proposed to solve this problem, based on genetic algorithms (GA), Particle Swarm Optimizer (PSO) and in recent years algorithms based in behavior algorithms based groups or herds of animals, such as frogs, bats, bees and domestic cats. This work presents the basic features of the algorithm based on the behavior of domestic cats and results to solve the SCP bi-objective, experimental results and opportunities to improve results using adaptive techniques applied to Cat Swarm Optimization. For this purpose we will use instances of SCP OR-Library of Beasley by adding an extra function fitness to transform the Beasly instance to Bi-Objective problem.

Keywords: Multiobjective problems · Evolutionary algorithm · Swarm optimization · Cat swarm optimization · Multiobjective cat swarm optimization · Pareto dominance

1 Introduction

Optimization problems require complex and optimal solutions because they relate to distribute limited basic resources. To resolve these problems it means improving the lives of poor people directly and enabling the growth of businesses, for example: resources related to social welfare, reaction by natural disasters, medical distribution capabilities. For these reasons the optimization generates a wide area of research in the sciences of Operations Research and Computer.

In the last decades bio-inspired algorithms have called the attention of researchers, in particular the heuristic Particle Swarm Optimization, which is

O. Gervasi et al. (Eds.): ICCSA 2016, Part I, LNCS 9786, pp. 220–231, 2016.
DOI: 10.1007/978-3-319-42085-1_17

based the behavior of some species: Bugs, fish, felines. These species use the collective intelligence to reach specific objectives guided by some community member. This paper is focused on studying the heuristic based on the behavior of domestic cats to solve a classical problem the Set Covering Problem (SCP).

2 Basic Concepts

2.1 Swarm Intelligent

Swarm intelligence (SI) is the collective behavior of independent individuals, that generate self-organizing, natural or artificial systems. Algorithms based on this principle are generally composed of simple agents that interact directly, locally, with simple rules, without centralized control, with interactions with stochastic components. This interaction between different autonomous agents generates an "intelligent" behavior, which gives rise to a pattern of global functioning that is used for optimization of complex mathematical functions. These techniques are inspired by nature (Bio Inspired), in processes such as ant colonies, schools, flocks or herds [1–3].

2.2 Multi Objective

Decision problems involves multiple evaluation criteria and generally they are in conflict [4]. To resolve a multi objective problem it required to optimize multiple criteria simultaneously. Exists a wide variety of cases in our society, for example: vehicle route optimization, environmental problems, allocation of medical resources [5]. The solution to multi-objective optimization problem it is presented by a set of feasible solutions, and the best of them define a set of non-dominated solutions, this set we will call Front. Formally the multi objective problem is defined as:

$$min \quad z(x) \; = \; [z_1(x), z_2(x), z_3(x), z_4(x), ..., z_M(x)] \tag{1}$$

The goal consists in minimizing a function z with M components with a vector variable x = $(x_1, \ldots, x_n)$ in a universe U, i.e., A solution u dominates v if u performs at least as well as v across all the objectives and performs better than v in at least one objective.

The dimensions of the target area corresponding to the number of functions to optimize. In this single-objective problem is one-dimensional space, since each decision vector corresponds to only a scalar number. In multi-objective problems, this is multi-dimensional space, where each dimension corresponds to each objective function to be optimized [6].

2.3 Pareto Dominance

If we have two candidate solutions u and v from U, vector z(u) is said to dominate vector $z(v)$ denoted by: z(u)$\prec$ z(v), if and only if:

$$z_i(u) \; \leq \; z_i(v), \qquad \forall \, i \; \in \; \{1, ..., M\} \tag{2}$$

$$z_i(u) \leq z_i(v), \qquad \exists\, i \in \{1, ..., M\} \tag{3}$$

If solution u is not dominated by any other solution, then u is declared as a Non Dominated or Pareto Optimal Solution. There are no superior solutions to the problem than u, although there may be other equally good solutions [7,8].

2.4 Hypervolume

When measuring the quality of multi-objective algorithms we consider two aspects: minimizing the distance of Pareto Front obtained by the algorithm to Front exact Pareto problem and maximize the spread of solutions on the front so that the distribution is as uniform possible [6]. Hypervolume is designed to measure both aspects: convergence and diversity - in a given front. This metric calculates the volume (in the objective space) covered by members of a given set, Q, non-dominated solutions to problems where all the objectives are to be minimized. Mathematically, for each $i \in$ Q a hypercube v_i is built with a reference point W and the solution i that define the diagonal thereof. The point W can be obtained simply with the worst values of the objective functions. Then the union of all hypercubes is what defines the hypervolume (HV):

$$HV = \bigcup_{i=1}^{|Q|} v_i \tag{4}$$

3 Set Covering Problem

SCP is defined as a fundamental problem in Operations Research and often described as a problem of coverage of *m-rows n-columns* of a binary matrix by a subset of columns to a minimum cost [9]. It is one of Karp's 21 NP-complete problems. This is the problem of covering the rows of an m-row, n column, zero-one $m\ x\ n$ matrix a_{ij} by a subset of the columns at minimal cost. Formally, the problem can be defined as:
Defining $x_j = 1$ if column j with cost c_j is in the solution and $x_j = 0$ otherwise

$$Minimize\ Z = \sum_{j=1}^{n} c_j x_j \qquad j \in \{1, 2, 3, ..., n\} \tag{5}$$

Subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1 \qquad i \in \{1, 2, 3, ..., m\} \tag{6}$$

$$x_j = \{0, 1\} \tag{7}$$

This definition contains a one fitness function, there is just one objective to be optimized. We study the case for two objective functions, using meta heuristic Cat Swarm Optimization (CSO) and using position vector of ones and zeros. A complete case study of SCP using CSO was done **Pontificia Universidad**

Cátolica de Valparaíso [10]. This work focuses on solving the SCP with two fitness functions, i.e., textit M = 2. To ensure the fitness functions have opposed criteria the second cost vector will be transposed the first, therefore the definition will be:

$$c_2 = (c_1)^t \tag{8}$$

$$min \quad z(x) = [z_1(x), z_2(x)] \tag{9}$$

$$Minimize\ Z_1 = \sum_{j=1}^{n} c_j^1 x_j \qquad j \in \{1, 2, 3, ..., n\} \tag{10}$$

$$Minimize\ Z_2 = \sum_{j=1}^{n} c_j^2 x_j \qquad j \in \{1, 2, 3, ..., n\} \tag{11}$$

Subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1 \qquad i \in \{1, 2, 3, ..., m\} \tag{12}$$

4 Cat Swarm Optimization CSO

4.1 Basic Concepts

A detailed description of the behavior of cats and especially domestic cats may be revised in [10,11]. This work indicates the specific concepts that control the behavior of the algorithm. There are importants features in their behavior and they employ to achieve their goals:

- Seeking mode. Resting but being alert - looking around its environment for its next move.
- Tracing mode. The cat is moving after its prey.
- The presentation of solution sets. How many cats we would like to use in the iteration and we must to define a mixture ratio (MR) which dictates the joining of seeking mode with tracing mode. According to observations, the cats spend a lot more time resting therefore MR should take a low value to ensure this feature.

The CSO was originally developed for continuous valued spaces. But there exist a number of optimization problems, as the SCP, in which the values are not continuous numbers but rather discrete binary integers. Sharafi et al. introduced a discrete binary version of CSO for discrete optimization problems: Binary Cat Swarm Optimization (BCSO) [14]. BCSO is based on CSO algorithm proposed by Chu, Tsai ann Pan in 2006 [11]. The difference is that in BCSO the vector position consists of ones and zeros, instead of the real numbers of CSO.

4.2 Parameters Important

In this paper, we considered the following BCSO parameters as relevant for our experimentation. We have considered the impact on the results in previous experiments

- NC: Number of population or pack cats
- MR: Mixture Rate that defines number of cats mode, this parameter must be chosen between 0 and 1. Define what percentage of cats are in seeking mode and tracing mode
- Termination condition. Normally is used a number of iterations.
- Adaptative criteria. To get better results usually we choose a parameter modified to perform the process again. In our work we choose the MR parameter especially to calculate the front not dominated ranging from 0.1 to 0.9 on increasing 0.1

4.3 Description of Cat Swarm Optimization - Main Algorithm

The main mechanism used in our experiments was consider as criteria of adaptive change the mixing ratio. In our experiment It was modified from 0.6 until 0.9 and determined non dominated solution

(a) Create N cats
(b) Initiate MR_p in min value (= 0.5)
(c) Create the cat swam, N cats working to solve the problem
(d) Define, randomly the position and velocity for each cat
(e) Distribute the swarm in tracing and seeking mode based on MR_p
(f) Check the cat mode, if cat is in Seeking Mode, applay Seekin Mode, else apply Tracing Mode
(g) Check if the cat is feasible solution (Ec. 11). if the cat satisfies the restriction compute the fitness (Ec. 9 and Ec.10) and compare with the non dominated solutions in the archive
(h) Update de solution file
(i) If number iteration less than the max iteration continue work, goto step c
(j) If MR_p les than max value MR, increment MR_p and go to step b
(k) Calculate the pareto front from non domination file

Below the two main modes are described

4.4 Seeking Mode

The seeking mode corresponds to a global search technique in the search space of the optimization problem. A term used in this mode is seeking memory pool (SMP). It is the number of copies of a cat produced in seeking mode.

There are four essential factors in this mode: seeking memory pool (SMP), seeking range of the selected dimension (SRD), counts of dimension to change (CDC), and self-position considering (SPC).

- SMP is used to define the size of seeking memory for each cat. SMP indicates the points explored by the cat. This parameter can be different for different cats.
- SRD declares the mutation ratio for the selected dimensions.
- CDC indicates how many dimensions will be varied.
- SPC is a Boolean flag, which decides whether current position of cat

The steps involved in this mode are:

(a) Create T (=SMP) copies of jth cat i.e. Y_{kd} where $(1 \leq k \leq T)$ and $(1 \leq d \leq D)$. D is the total number of dimensions.
(b) Apply a mutation operator to Y_k.
(c) Evaluate the fitness of all mutated copies.
(d) Update the contents of the archive with the position of those mutated copies which represent non dominated solutions.
(e) Pick a candidate randomly from T copies and place it at the position of *jth* cat.

4.5 Tracing Mode

The tracing mode corresponds to a local search technique for the optimization problem. In this mode, the cat traces the target while spending high energy. The rapid chase of the cat is mathematically modeled as a large change in its position. Define position and velocity of ith cat in the *D-dimensional* space as $X_i = (X_{i1}, X_{i2}, X_{i3} \ldots X_{iD})$ and $V_i = (V_{i1}, V_{i2}, V_{i3} \ldots V_{iD})$ where $(1 \leq d \leq D)$ represents the dimension. The global best position of the cat swarm is represented as $X_g = (X_{g1}, X_{g2}, X_{g3} \ldots X_{gD})$. The steps involved in tracing mode are:

(a) Compute the new velocity of *ith* cat using (13)

$$V_{id} = w * V_{id} + c * r * (X_{gd} - X_{id}) \tag{13}$$

where
w = is the inertia weight
c = is the acceleration constant
r = is a random number uniformly distributed in the range [0, 1]

(b) Compute the new position of *ith* cat using

$$V_{id} = X_{gd} - X_{id} \tag{14}$$

(c) If the new position of *ith* cat corresponding to any dimension goes beyond the search space, then the corresponding boundary value is assigned to that dimension and the velocity corresponding to that dimension is multiplied by -1 to continue the search in the opposite direction.
(d) Evaluate the fitness of the cats.
(e) Update the contents of the archive with the position of those cats which represent no dominated vectors.

5 The Execution of the Algorithm

1 Load Instance SCP
2 Initiate phase
  (a) For to obtein the best Pareto Front: $MR = 0, 1...0.99$
  (b) For Analysis CSO, $MR = 0.5$
  (c) Initiate: SRD, CDC, SPC, w, $c1$
  (d) Define the size file Pareto
  (e) Define Iteration number
  (f) Define, randomly the position and velocity for each cat
3 Distribute the swarm in tracing and seeking mode based on MR
4 Repeat until iteration number reached
  (a) If *cat* is seeking mode, apply seeking process and return a solution candidate $\overrightarrow{X}$
  (b) If *cat* is tracing mode, apply seeking process and return a solution candidate $\overrightarrow{X}$
  (c) Check if the $\overrightarrow{X}$ satisfies the restriction problem: $\sum_{j=1}^{n} a_{ij} X_j \geq 1$
  (d) Compute
    $f1 = \sum_{j=1}^{n} c_{1j} X_j$
    $f2 = \sum_{j=1}^{n} c_{2j} X_j$
  (e) Store the position of the cats representing non-dominated solutions in the archive
5 Calculate the pareto front from non domination file

This algorithm was executed 30 times for each SCP Instance, and the pareto front was obtained $ParetoFront = \bigcup_{i=1}^{30} (pf)_i$

6 Experimental Results

The CSO was evaluated using the next features:

Table 1. Parameter values CSO

Name	Parameter	Value	Obs
Number of cats	C	30	
Mixture ratio	MR	0.5	
Seeking memory pool	SMP	20	-
Probability of Mutation	PMO	1	-
Counts of dimensions to change	CDC	0,001	-
Inertia weight	w	1	-
Factor c_1	c_1	1	-

Table 2. Experimentals results

INST	HYPER	MAX	MIN	PROM	DESV
scp41	0,6223	132,441	109,345	118,84	7,48
scp42	0,6845	156,556	121,211	143,1	10,03
scp43	0,7261	135,3	115,309	125,22	6,53
scp44	0,5804	154,609	129,51	140,53	6,38
scp45	0,7426	134,763	105,963	119,49	9,28
scp46	0,5435	140,833	114,68	134,02	7,41
scp47	0,5172	147,812	126,058	136,42	7,26
scp48	0,7319	135,586	114,344	120,57	7,09
scp49	0,6029	159,194	135,4	148,2	7,13
scp51	0,6156	270,516	247,489	256,56	7,63
scp52	0,6378	282,612	259,742	270,77	7,24
scp53	0,6613	257,966	203,538	229,88	17,42
scp54	0,8511	259,181	212,809	241,01	15,83
scp55	0,5872	234,38	205,496	225,25	9,034
scp56	0,7223	265,601	218,673	238,11	14,736
scp57	0,6036	259,252	234,426	245,85	8,5
scp58	0,6242	270,754	242,436	254,9	9,502
scp59	0,5338	243,13	209,511	227,58	11,928
scp61	0,5992	103,339	81,946	94,31	7,73
scp62	0,6673	100,748	79,064	91,99	8,472
scp63	0,6873	96,555	77,817	86,94	6,077
scp64	0,6363	103,206	78,183	90,4	9,285
scp65	0,6696	101,83	78,088	90,66	7,244
scpa1	0,7834	506,95	463,377	482,7	14,943
scpa2	0,5462	618,465	513,501	559,59	34,738
scpa3	0,5631	517,718	474,45	496,63	13,896
scpa4	0,6269	526,053	469,132	502,76	17,687
scpa5	0,7679	529,614	445,58	488,48	27,499

MAX, MIN, PROM, DESV in sec

1 Using 65 Instances for set covering from OR-Library of Beasley [12]
2 MacBook Pro (13-inch, Mid 2012), CPU MacBook Pro (13-inch, Mid 2012), 16 GB 1333 MHz DDR3, OS X Yosemite, version 10.10.5
3 IDE: BlueJ version 3.1.5 (Java version 1.8.0_31)

The Optimal Pareto Front was estimated individually for each instance varying the value of MR from 0.1 until 0.9 using an increment 0.1, and we choose use the hypervolume because is the only one indicator of performance that is

compatible unary with Pareto dominance and it has been able to demonstrate that its maximization is equivalent to achieve convergence to the true Pareto front for a specific value of MR (fig. a) and collectively by varying the value of MR from 0.5 until 0.9 using an increment 0.1 (fig b). According to the results, the best result is obtained collectively. The working conditions of the process were:

1 1.500 iterations for each MR value
2 30 times each Beasly instance
3 Varying MR from 0.5 until 0.9 using an increment 0.1
4 The parameters BCO was obteined from [10,13] and show in Table 1.

Table 2 shows the experimental results for each Beasly Instance. The Optimal Pareto Front was obtained varying MR from 0.1 until 0.99 and determined by the union of fronts obtained MR.

In our experiments the best HV was with SCP54 instance, however the worst HV was with SCP47. If we observe the charts there are zones of the solution space that they need a better exploration strategy by changing the calculation (Fig. 1).

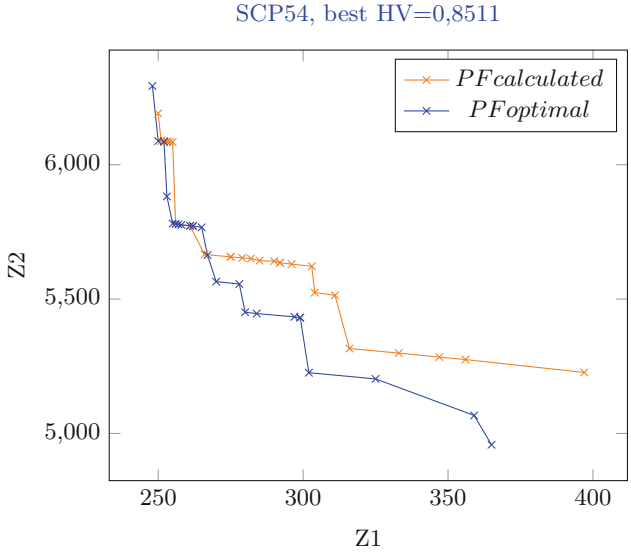

Fig. 1. Best HV was with SCP54 instance (Color figure online)

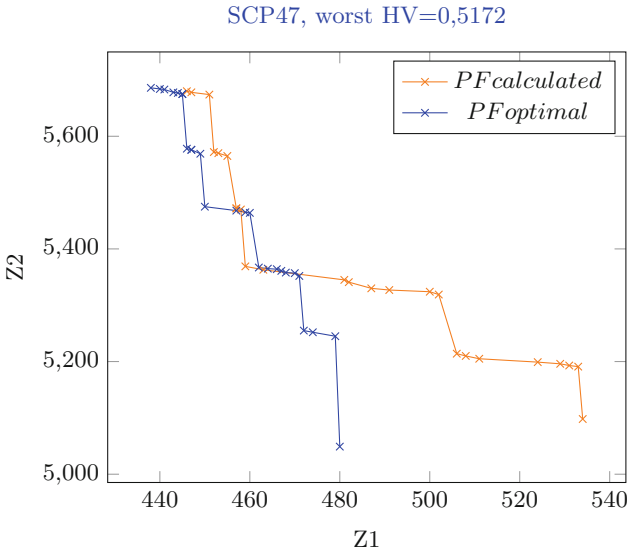

Fig. 2. Worst HV was with SCP54 instance (Color figure online)

7 Conclusion and Future Work

There are not published results on a SCP Bi Objective using BCSO and we think the Pareto Front is quite promising considering just we varied only MR. We also think that applying adaptive mechanisms in other parameters of the CSO metaheuristic we can improve results. We also believe that this particular CSO should be compared with genetic and evolutionary algorithms. Within the same field with metaheuristics highlighted with ants, bees and frogs. The proposed BCSO is implemented and tested using 65 SCP test instances from the OR-Library of Beasley. This is first phase of our research. We only work with MR parameter, however. We think that using adaptive techniques for parameter we wil improve our results (Fig. 2). The next step:

1 To use adaptive techniques for BCSO parameters to improve HV
2 To obtain a Pareto optimal front using genetic, evolutionary algorithms. Within the same field with metaheuristics highlighted with ants, bees and frogs and

Acknowledgements. The author Broderick Crawford is supported by grant CONICYT/FONDECYT/REGULAR/1140897 and Ricardo Soto is supported by Grant CONICYT/FONDECYT/REGULAR/1160455.

References

1. Crawford, B., Soto, R., Cuesta, R., Olivares-Suárez, M., Johnson, F., Olguin, E.: Two swarm intelligence algorithms for the set covering problem. In: 2014 9th International Conference on Software Engineering and Applications (ICSOFT-EA), pp. 60–69. IEEE (2014)
2. Crawford, B., Soto, R., Peña, C., Riquelme-Leiva, M., Torres-Rojas, C., Misra, S., Johnson, F., Paredes, F.: A comparison of three recent nature-inspired metaheuristics for the set covering problem. In: Gervasi, O., Murgante, B., Misra, S., Gavrilova, M.L., Rocha, A.M.A.C., Torre, C., Taniar, D., Apduhan, B.O. (eds.) ICCSA 2015. LNCS, vol. 9158, pp. 431–443. Springer, Heidelberg (2015)
3. Gervasi, O., Murgante, B., Misra, S., Gavrilova, M.L., Rocha, A.M.A.C., Torre, C., Taniar, D., Apduhan, B.O. (eds.): ICCSA 2015. LNCS, vol. 9158. Springer, Heidelberg (2015)
4. Lopez, J., Lanzarini, L.C., Leguizamón, G.: Optimización multiobjetivo: aplicaciones a problemas del mundo real. Buenos Aires, Argentina, Universidad Nacional de la Plata, pp. 66–90 (2013)
5. Wikipedia, Problema del conjunto de cobertura – wikipedia, la enciclopedia libre (2014). [Internet; descargado 29-octubre-2015]
6. Zitzler, E., Thiele, L.: Multiobjective optimization using evolutionary algorithms–a comparative case study. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) Parallel Problem Solving from Nature—PPSN V. LNCS, vol. 1498, pp. 292–301. Springer, Heidelberg (1998)
7. Wikipedia, Pareto efficiency – wikipedia, the free encyclopedia (2015). Accessed 29 Oct 2015
8. Knowles, J., Corne, D.: The pareto archived evolution strategy: a new baseline algorithm for pareto multiobjective optimisation. In: Proceedings of the 1999 Congress on Evolutionary Computation, 1999, CEC 99, vol. 1, IEEE (1999)
9. Caprara, A., Toth, P., Fischetti, M.: Algorithms for the set covering problem. Ann. Oper. Res. **98**(1–4), 353–371 (2000)
10. Crawford, B., Soto, R., Berrios, N., Johnson, F., Paredes, F.: Binary cat swarm optimization for the set covering problem pp. 1–4 (2015)
11. Chu, S.-C., Tsai, P.-W.: Computational intelligence based on the behavior of cats. Int. J. Innovative Comput. Inf. Control **3**(1), 163–173 (2007)
12. Beasley, J.E.: A lagrangian heuristic for set-covering problems. Nav. Res. Logistics (NRL) **37**(1), 151–164 (1990)
13. Pradhan, P.M., Panda, G.: Solving multiobjective problems using cat swarm optimization. Expert Syst. Appl. **39**(3), 2956–2964 (2012)
14. Chu, S.-C., Tsai, P., Pan, J.-S.: Cat swarm optimization. In: Yang, Q., Webb, G. (eds.) PRICAI 2006. LNCS (LNAI), vol. 4099, pp. 854–858. Springer, Heidelberg (2006)
15. Panda, G., Pradhan, P.M., Majhi, B.: IIR system identification using cat swarm optimization. Expert Syst. Appl. **38**(10), 12671–12683 (2011)
16. Lust, T., Tuyttens, D.: Two-phase pareto local search to solve the biobjective set covering problem. In: 2013 Conference on Technologies and Applications of Artificial Intelligence (TAAI), pp. 397–402. IEEE (2013)
17. Durillo, J.J., Nebro, A.J.: jmetal: A java framework for multi-objective optimization. Ad. Eng. Softw. **42**(10), 760–771 (2011)
18. Wikipedia, Optimización multiobjetivo – wikipedia, la enciclopedia libre (2013). [Internet; descargado 29-octubre-2015]

19. Friedrich, T., He, J., Hebbinghaus, N., Neumann, F., Witt, C.: Approximating covering problems by randomized search heuristics using multi-objective models*. Evol. Comput. **18**(4), 617–633 (2010)
20. Vazirani, V.V.: Approximation Algorithms. Springer Science & Business Media, New York (2013)
21. Bouzidi, A., Riffi, M.E.: Cat swarm optimization to solve flow shop scheduling problem. J. Theor. Appl. Inf. Technol. 72(2) (2015)
22. Hadi, I., Sabah, M.: Improvement cat swarm optimization for efficient motion estimation. Int. J. Hybrid Inf. Technol. **8**(1), 279–294 (2015)
23. Musliu, N.: Local search algorithm for unicost set covering problem. In: Ali, M., Dapoigny, R. (eds.) IEA/AIE 2006. LNCS (LNAI), vol. 4031, pp. 302–311. Springer, Heidelberg (2006)
24. Zhang, L.-B., Zhou, C.-G., Liu, X., Ma, Z., Ma, M., Liang, Y.: Solving multi objective optimization problems using particle swarm optimization. In: Proceedings of IEEE congress on evolutionary computation, pp. 2400–2405 (2003)
25. Crawford, B., Soto, R., Aballay Leiva, F., Johnson, F., Paredes, F.: The set covering problem solved by the binary teaching-learning-based optimization algorithm. In: 2015 10th Iberian Conference on Information Systems and Technologies (CISTI), pp. 1–4. IEEE (2015)
26. Crawford, B., Soto, R., Peña, C., Riquelme-Leiva, M., Torres-Rojas, C., Johnson, F., Paredes, F.: Binarization methods for shuffled frog leaping algorithms that solve set covering problems. In: Silhavy, R., Senkerik, R., Oplatkova, Z.K., Prokopova, Z., Silhavy, P. (eds.) Software Engineering in Intelligent Systems. AISC, vol. 349, pp. 317–326. Springer, Heidelberg (2013)
27. Crawford, B., Soto, R., Aballay, F., Misra, S., Johnson, F., Paredes, F.: A teaching-learning-based optimization algorithm for solving set covering problems. In: Gervasi, O., Murgante, B., Misra, S., Gavrilova, M.L., Rocha, A.M.A.C., Torre, C., Taniar, D., Apduhan, B.O. (eds.) ICCSA 2015. LNCS, vol. 9158, pp. 421–430. Springer, Heidelberg (2015)
28. Crawford, B., Soto, R., Torres-Rojas, C., Peña, C., Riquelme-Leiva, M., Misra, S., Johnson, F., Paredes, F.: A binary fruit fly optimization algorithm to solve the set covering problem. In: Gervasi, O., Murgante, B., Misra, S., Gavrilova, M.L., Rocha, A.M.A.C., Torre, C., Taniar, D., Apduhan, B.O. (eds.) ICCSA 2015. LNCS, vol. 9158, pp. 411–420. Springer, Heidelberg (2015)
29. Cuesta, R., Crawford, B., Soto, R., Paredes, F.: An artificial bee colony algorithm for the set covering problem. In: Silhavy, R., Senkerik, R., Oplatkova, Z.K., Silhavy, P., Prokopova, Z. (eds.) Modern Trends and Techniques in Computer Science. AISC, vol. 285, pp. 53–63. Springer, Heidelberg (2014)