A Black Hole Algorithm for Solving the Set Covering Problem

Ricardo Soto[1(✉)], Broderick Crawford[1], Ignacio Figueroa[1], Stefanie Niklander[2,3,4], and Eduardo Olguín[5]

[1] Pontificia Universidad Católica de Valparaíso, Valparaíso, Chile
{ricardo.soto,broderick.crawford}@ucv.cl, ignacio.figueroa.b@mail.pucv.cl
[2] Universidad Adolfo Ibañez, Viña del Mar, Chile
stefanie.niklander@uai.cl
[3] Universidad Autónoma de Chile, Santiago, Chile
[4] Universidad Cientifica del Sur, Lima, Peru
[5] Universidad San Sebastián, Santiago, Chile
eduardo.olguin@uss.cl

Abstract. The set covering problem is a classical optimization benchmark with many industrial applications such as production planning, assembly line balancing, and crew scheduling among several others. In this work, we solve such a problem by employing a recent nature-inspired metaheuristic based on the black hole phenomena. The core of such a metaheuristic is enhanced with the incorporation of transfer functions and discretization methods to handle the binary nature of the problem. We illustrate encouraging experimental results, where the proposed approach is capable to reach various global optimums for a well-known instance set from the Beasley's OR-Library.

Keywords: Meta-heuristics · Soft computing · Black Hole Algorithm · Set covering problem

1 Introduction

The Set Covering Problem (SCP) is a classic benchmark in the subject of combinatorial optimization that belongs to the NP-complete class of problems [19]. The purpose of the SCP is to find a set of solutions that cover a range of needs at the lowest possible cost. The SCP can be applied to many real-world problems, such are airline crew scheduling [14], network discovery [10], plant location [12], and service allocation [6] among others. Different algorithms have been developed to solve the classic SCP, ranging from classic exact methods to more recent bio-inspired metaheuristics. Exact methods can be applied to solve SCPs [1,2], the main problem is when the instance size increases the algorithm is commonly unable to reach a solution in a reasonable amount of time. Approximate methods such as the well-known metaheuristics tackle this concern, being capable to generally provide good enough local optimums in a limited time interval.

H. Fujita et al. (Eds.): IEA/AIE 2016, LNAI 9799, pp. 855–861, 2016.
DOI: 10.1007/978-3-319-42007-3_72

In this context a large list of metaheuristics have been proposed to solve the SCP [4,5,8,9,17,20].

In this paper, a new approach for SCPs based on the black hole algorithm is presented. The Black Hole Algorithm (BHA) is a population-based metaheuristic based on the gravitational force that has a black hole to attract everything that is around it. The core of the BHA is enhanced with the incorporation of binarization through transfer and discretization functions in order to handle the binary nature of the SCP. Repairing operators are also employed to rapidly discard the unfeasible solutions and as a consequence to alleviate the search. We present promising results on 40 well-known pre-processed instances from the Beasley's OR-Library, where a considerable amount of global optimums are reached.

This paper is organized as follows: In Sect. 2, we describe the SCP. Next section presents the BHA including binarization and repairing. Section 4 provides the experimental results, followed by conclusions and future work.

2 The Set Covering Problem

The Set Covering Problem consists in finding a set of solutions at the lowest possible cost to cover a set of needs. Formally, we define the problem as follows: Let $A = (a_{ij})$ be a binary matrix with $m\text{–}rows \times n\text{–}columns$, and let $C = (c_j)$ be a vector representing the cost of each column j, assuming that $c_j > 0$ for $(j \in N)$. So we can say that column $(j \in N)$ cover a row i that exists in M if $a_{ij} = 1$. The mathematical model is as follows:

$$min\,(z) = \sum_{j=1}^{n} c_j X_j$$

Subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1 \qquad \forall i \in M \qquad\qquad x_j = \begin{cases} 1\, j \in S \\ 0 \;\; if \;\; not \end{cases} \forall\, j \in N$$

3 The Black Hole Algorithm

A black hole is a region of space that has so much mass concentrated in it that there is no way for a nearby object to escape its gravitational pull [15]. Anything falling into a black hole, including light, cannot escape. The BHA is inspired on this phenomena [11].

Similar to other population-based metaheuristics, The BHA begins by randomly generating a population of candidate solutions, called stars, which are placed in the search space of some problem or function. After initialization, the fitness values of the population are evaluated and the best candidate, which has the best fitness values is introduced as black hole and the other solutions are selected as normal stars. Then, all the stars commence moving towards the black hole due of the power absorbing of the black hole.

The absorption of stars by the black hole is formulated as follows: $x_i(t+1) = x_i + rand * (x_{bh} - x_i(t)) \geq 1 \ for \ i = \{1, 2, 3, \ldots, N\}$. Where $x_i(t+1)$ is the location of the i_{th} star at the iteration t+1, *Rand* is a random number between zero and one, x_{bh} is the location of the black hole in the search space, and N is the number of solutions (stars). In addition, there is a distance between stars and black hole, the stars that crosses the event horizon of the black hole will be absorbed by the black hole, in carrying out this event another candidate solution (star) is born and distributed randomly in the search space and starts a new iteration, this is known as probability of crossing the event horizon. This is done to keep the number of candidate solutions constant.

The radius of the event horizon in the black hole algorithm is calculated by using the following equation: $E = f_{BH} / \sum_{i=1}^{N} f_i$. Where f_{bh} is the fitness value of the black hole, f_i is the fitness value of the i_{th} star, and N is the number of candidate solutions (stars). When the distance from the black hole with the star is less than the radio, or in other words when the difference in fitness between the black hole and the star is less than the radio, that star is swallowed by the black hole.

3.1 Binarization

When the star moves toward the black hole, the algorithm generates a real number which must be transformed to a binary domain due to the nature of the problem treated. To this end, we firstly employ a transfer function, which map a real value to a $[0, 1]$ real interval. As transfer function we employ the V-shaped-V4 (Eq. 1), which is was the best-performing one among the 8 tested transfer functions (4 S-shaped and 4 V-shaped) [8,13]. Then, the resulting value from the transfer function is discretized via the half method depicted in Eq. 2 in order to obtain a binary value.

$$T(x) = \left| \frac{2}{\pi} arctan \left(\frac{\pi}{2} x \right) \right| \tag{1}$$

$$x_i(t+1) = \begin{cases} 1 \text{ if } rand > 0.5 \\ \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

3.2 Repairing

The BHA, as most metaheuristics do, generates a random population with solutions that violate the constraints, i.e., solutions holding uncovered rows. Repairing operators are responsible for turning unfeasible solutions on feasible ones. To this end, we incorporate a heuristic operator that achieves the generation of feasible solutions, and additionally eliminates column redundancy [3].

To make all feasible solutions we compute a ratio based on the sum of all the constraint matrix rows covered by a column c_j/N_{uc}, where N_{uc} is the amount of uncovered columns. The unfeasible solution are repaired by covering the columns of the solution that had the lower ratio. After this, a local optimization step is applied, where column redundancy is eliminated. A column is redundant when it can be deleted and the feasibility of the solution is not affected.

4 Experiment Results

The performance of the proposed black hole algorithm was experimentally evaluated by using 40 preprocessed instances of the SCP from the Beasley's OR-Library[1]. This algorithm has been implemented in Java and the experiments have been launched on a 2.3 Ghz Intel Core i3 with 4 GB RAM machine running Windows 7. We employ an initial population of 20 stars, 4000 iterations and 20 executions per instance. The results are given in Table 1 where column 1

Table 1. Results obtained by BHA for the tested SCP instances.

Instance	Opt	Best	Avg	RPD	Instance	Opt	Best	Avg	RPD
4.1	429	430	430.25	0.23	6.1	138	140	142.95	1.45
4.2	512	512	512	0.0	6.2	146	147	149.1	0.68
4.3	516	516	517.2	0.0	6.3	145	145	147.7	0.0
4.4	494	495	495.25	0.20	6.4	131	131	131	0.0
4.5	512	514	514.9	0.39	6.5	161	161	163.5	0.0
4.6	560	560	560.9	0.0	A.1	253	253	255.5	0.0
4.7	430	430	431.1	0.0	A.2	252	253	257.35	0.39
4.8	492	493	496.3	0.20	A.3	232	233	235.65	0.43
4.9	641	644	648.05	0.46	A.4	234	234	234.95	0.0
4.10	514	514	515.05	0.0	A.5	236	236	236.7	0.0
5.1	253	253	255.6	0.0	B.1	69	69	70.3	0.0
5.2	302	305	306.2	0.99	B.2	76	76	77.6	0.0
5.3	226	228	228	0.88	B.3	80	80	80.9	0.0
5.4	242	242	242.25	0.0	B.4	79	79	80.1	0.0
5.5	211	211	211.4	0.0	B.5	72	72	72.3	0.0
5.6	213	213	213.15	0.0	C.1	227	229	231.25	0.88
5.7	293	293	295.05	0.0	C.2	219	219	221.4	0.0
5.8	288	288	289	0.0	C.3	243	245	250.7	0.82
5.9	279	279	282.35	0.0	C.4	219	219	222.7	0.0
5.10	265	265	265.1	0.0	C.5	215	215	216.6	0.0

[1] Available at http://www.brunel.ac.uk/~mastjjb/jeb/info.html.

shows the SCP instance, column 2 depicts the best known optimum for the instance, column 3 provides the best optimal value found by the algorithm, while columns 4 and 5 show average of results and the relative percentage deviation, respectively. The relative percentage deviation (RPD) is computed as follows: $RDP = (Z - Z_{opt})/Z_{opt} \times 100$, where Z is the best optimum value found by the metaheuristic and Z_{opt} depicts the best known optimum value for the instance.

In Table 2, the proposed approach is compared with three recently reported metaheuristics for the SCP, namely, shuffled frog leaping algorithm (SFLA) [7], XOR-based artificial bee colony (xABC) [16], and a binary firefly algorithm (BFF) [8]. Table 3 depicts the amount of global optimums reached by each algorithm. BHA is able to reach 27 global optimums, while the results for the remaining 13 instances stay very near to the global optimum (RPDs around 1 %). The

Table 2. Results obtained using BHA for instances SCP

Instance	Opt	BHA	SFLA	xABC	BFF	Instance	Opt	BHA	SFLA	xABC	BFF
4.1	429	430	430	430	429	6.1	138	140	140	142	138
4.2	512	512	513	512	517	6.2	146	147	147	147	147
4.3	516	516	519	519	519	6.3	145	145	147	148	147
4.4	494	495	501	495	495	6.4	131	131	131	131	131
4.5	512	514	514	514	514	6.5	161	161	166	165	164
4.6	560	560	563	561	563	A.1	253	253	255	254	255
4.7	430	430	431	431	430	A.2	252	253	160	257	259
4.8	492	493	497	493	497	A.3	232	233	237	235	238
4.9	641	644	656	649	655	A.4	234	234	235	236	235
4.10	514	514	518	517	519	A.5	236	236	236	236	236
5.1	253	253	254	254	257	B.1	69	69	70	70	71
5.2	302	305	307	309	309	B.2	76	76	76	78	78
5.3	226	228	228	229	229	B.3	80	80	80	80	80
5.4	242	242	242	242	242	B.4	79	79	79	80	79
5.5	211	211	211	211	211	B.5	72	72	72	72	72
5.6	213	213	213	214	213	C.1	227	229	229	231	230
5.7	293	293	297	298	298	C.2	219	219	223	222	223
5.8	288	288	291	289	291	C.3	243	245	253	254	253
5.9	279	279	281	280	284	C.4	219	219	227	231	225
5.10	265	265	265	267	265	C.5	215	215	217	216	217

Table 3. Optimums reached for the 40 instances

	BHA	SFLA	xABC	MBFF
Opt. reached	**27/40**	10/40	7/40	11/40

results also illustrate that BHA greatly outperforms its competitors, which were unable to reach more than 12 optimum values from the 40 tested instances. Let us also note the robustness of the proposed BHA, whose averages for 20 executions remain very close to the best optimum value found.

5 Conclusions

In this paper we have presented a new approach for solving SCPs based on the black hole algorithm. We have incorporated a transfer function and a discretization method in order to handle the binary nature of the problem. Repairing operators are also employed to avoid unfeasible solutions and column redundancy. We have tested 40 non-unicost instances from the Beasley's OR-Library where the quality of results clearly outperform very recent reported metaheuristics for the SCP. The proposed approach is also robust able to provide averages very near to global optimums. As future work, we plan to test larger instances of the SCP as well as to incorporate adaptive capabilities to the BHA for performing parameter tuning during solving as exhibited in [18].

References

1. Balas, E., Carrera, M.C.: A dynamic subgradient-based branch-and-bound procedure for set covering. Locat. Sci. **5**(3), 203–203 (1997)
2. Beasley, J.E.: An algorithm for set covering problem. Eur. J. Oper. Res. **31**(1), 85–93 (1987)
3. Beasley, J.E., Chu, P.C.: A genetic algorithm for the set covering problem. Eur. J. Oper. Res. **94**(2), 392–404 (1996)
4. Brusco, M.J., Jacobs, L.W., Thompson, G.M.: A morphing procedure to supplement a simulated annealing heuristic for cost and coveragecorrelated set covering problems. Ann. Oper. Res. **86**, 611–627 (1999)
5. Caserta, M.: Tabu search-based metaheuristic algorithm for large-scale set covering problems. In: Doerner, K.F., Gendreau, M., Greistorfer, P., Gutjahr, W., Hartl, R.F., Reimann, M. (eds.) Operations Research/Computer Science Interfaces Series, vol. 39, pp. 43–63. Springer, New York (2007)
6. Ceria, S., Nobili, P., Sassano, A.: Annotated Bibliographies in Combinatorial Optimization. Wiley, Chichester (1997)
7. Crawford, B., Soto, R., Peña, C., Palma, W., Johnson, F., Paredes, F.: Solving the set covering problem with a shuffled frog leaping algorithm. In: Nguyen, N.T., Trawiński, B., Kosala, R. (eds.) ACIIDS 2015. LNCS, vol. 9012, pp. 41–50. Springer, Heidelberg (2015)
8. Crawford, B., Soto, R., Riquelme-Leiva, M., Peña, C., Torres-Rojas, C., Johnson, F., Paredes, F.: Modified binary firefly algorithms with different transfer functions for solving 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. 307–315. Springer, Heidelberg (2015)
9. 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, Switzerland (2014)

10. Grossman, T., Wool, A.: Computational experience with approximation algorithms for the set covering problem. Eur. J. Oper. Res. **101**(1), 81–92 (1997)
11. Hatamlou, A.: Black hole: a new heuristic optimization approach for data clustering. Inf. Sci. **222**, 175–184 (2013)
12. Krarup, J., Bilde, O.: Plant location, set covering and economic lot size: an 0 (mn)-algorithm for structured problems. In: Collatz, L., Meinardus, G., Wetterling, W. (eds.) Numerische Methoden bei Optimierungsaufgaben. International Series of Numerical Mathematics, vol. 36, pp. 155–180. Birkhuser, Basel (1977)
13. Mirjalili, S., Hashim, S., Taherzadeh, G., Mirjalili, S., Salehi, S.: A Study of Different Transfer Functions for Binary Version of Particle Swarm Optimization. CSREA Press, Las Vegas (2011)
14. Mirjalili, S., Lewis, A.: S-shaped versus V-shaped transfer functions for binary particle swarm optimization. Swarm Evol. Comput. **9**, 1–14 (2013)
15. Ruffini, R., Wheeler, J.A.: Introducing the black hole. Phys. Today **24**(1), 30 (1971)
16. Soto, R., Crawford, B., Lizama, S., Johnson, F., Paredes, F.: A XOR-based ABC algorithm for solving set covering problems. In: Gaber, T., Hassanien, A.E., El-Bendary, N., Dey, N. (eds.) Proceedings of the 1st International Conference on Advanced Intelligent System and Informatics (AISI). AISC, vol. 407, pp. 208–218. Springer, Switzerland (2016)
17. Soto, R., Crawford, B., Muñoz, A., Johnson, F., Paredes, F.: Pre-processing, repairing and transfer functions can help binary electromagnetism-like algorithms. In: Silhavy, R., Senkerik, R., Oplatkova, Z.K., Prokopova, Z., Silhavy, P. (eds.) Artificial Intelligence Perspectives and Applications. AISC, vol. 347, pp. 89–97. Springer, Heidelberg (2015)
18. Soto, R., Crawford, B., Palma, W., Galleguillos, K., Castro, C., Monfroy, E., Johnson, F., Paredes, F.: Boosting autonomous search for CSPs via skylines. Inf. Sci. **308**, 38–48 (2015)
19. Ullman, J.D.: Np-complete scheduling problems. J. Comput. Syst. Sci. **10**(3), 384–393 (1975)
20. Valenzuela, C., Crawford, B., Soto, R., Monfroy, E., Paredes, F.: A 2-level metaheuristic for the set covering problem. Int. J. Comput. Commun. Control **7**(2), 377 (2014)