Discrete Cuckoo Search with Local Search for Max-cut Problem

Yingying Xu, Zhihua Cui(✉), and Lifang Wang

Complex System and Computational Intelligence Laboratory,
Taiyuan University of Science and Technology, Taiyuan 030024, Shanxi, China
s_yingyingxu@163.com, zhihua.cui@hotmail.com,
wlf1001@163.com

Abstract. Max-cut problem is a well-known NP-hard problem, in this paper, a hybrid meta-heuristic algorithm is designed to solve it. In this algorithm, the discrete cuckoo search is employed to find the approximate satisfied solution, while the local optimal solution is used to further improve the performance. To test the validity, three other algorithms are used to compare, simulation results show our modification is effective.

Keywords: Max-cut problem · Discrete cuckoo search algorithm · Local search strategy

1 Introduction

Max-cut problem is a well-known NP-hard graph problem. For a graph $G = (V, E)$, $V = \{1, 2, \ldots, n\}$ is vertex set and E is the ordered set of undirected edges. Let w_{ij} be the weight associated with edge $\{i, j\} \in E$, then max-cut problem is to find an optimal partition (V_1, V_2) $(V_1 \cap V_2 = \phi,\ V_1 \cup V_2 = V)$, so that the total weights of the edges crossing different subsets is maximized, in other words, the objective function can be represented as:

$$\sum_{i \in V_1, j \in V_2} w_{ij} = \sum_{i<j} w_{ij} \cdot \frac{1 - x_i x_j}{2} \tag{1}$$

where $x_i \in \{1, -1\} (i = 1, 2, \cdots, n)$, $x_i = 1$ represents $x_i \in V_1$, as well as $x_i = -1$ denotes $x_i \in V_2$.

During the past years, many algorithms have been designed to solve it, including exact algorithm, approximate algorithm and heuristic algorithm [1, 2]. Heuristic algorithm [3–5] is an umbrella for all population-based stochastic optimization algorithm inspired by heuristic information [6], such as ant colony optimization [7, 8], fruit fly optimization [9], particle swarm optimization [10–12], artificial bee colony [13–16], social emotional optimisation algorithm [17], firefly algorithm [18–22] and bat algorithm [23–25].

Z. Shi et al. (Eds.): ICIS 2017, IFIP AICT 510, pp. 66–74, 2017.
DOI: 10.1007/978-3-319-68121-4_7

For max-cut problem, Laguna et al. [26] designed a hybrid version of cross entropy method, while Lin [27] proposed a discrete dynamic convexized method. Festa et al. [28] investigated several heuristics derived from greedy randomized adaptive search procedure and variable neighborhood search. In 2007, Wang [29] proposed a hybrid algorithm combining with chaotic discrete Hopfield neural network and genetic particle swarm optimization, while in 2011, Wang [30] designed another combination with tabu Hopfield neural network and estimation of distribution algorithm. Inspired by this work, Lin [31] designed an integrated method combined with particle swarm optimization and estimation of distribution algorithm, and a local search strategy is employed to improve the accuracy. Shylo [32] also employed global equilibrium search algorithms and tabu search to improve the accuracy.

In this paper, we propose a new heuristic algorithm to combining the cuckoo search algorithm and local search strategy, and apply it to solve max-cut problem. The rest of this paper is organized as follows: In Sect. 2, the details of our proposed hybrid algorithm are presented, as well as the simulation results are reported in Sect. 3.

2 Hybrid Algorithm

2.1 Discrete Cuckoo Search Algorithm

Cuckoo search algorithm was proposed in 2009 [33], up to now, many variants are proposed to improve the performance [34–36]. However, max-cut problem is a combination problem, and a discrete cuckoo search algorithm is designed to solve it.

In this discrete version, the position movement will use the following strategies [37]:
Strategy 1:

$$x_{ijk}^{m+1} = \begin{cases} x_{ijk}^{m}, rand() \le Sig(Step) \\ -1, other \end{cases} \tag{2}$$

where

$$Sig(Step) = 1/(1 + \exp(-Step)) \tag{3}$$

Strategy 2:

$$x_{ijk}^{m+1} = \begin{cases} -1, rand() \le Sig(Step) \\ x_{ijk}^{m}, other \end{cases} \tag{4}$$

where

$$Sig(Step) = 1 - 2/(1 + \exp(-Step)) \tag{5}$$

Strategy 3:

$$x_{ijk}^{m+1} = \begin{cases} 1, rand() \leq Sig(Step) \\ x_{ijk}^{m}, other \end{cases} \tag{6}$$

where

$$Sig(Step) = 2/(1 + \exp(-Step)) - 1 \tag{7}$$

The most difference among three strategy is the sigma function $sig(step)$, and jump path $step$ is a random number with Levy distribution, to provide a deep insight, the Eqs. (3), (5) and (7) are plotted in Figs. 1, 2 and 3.

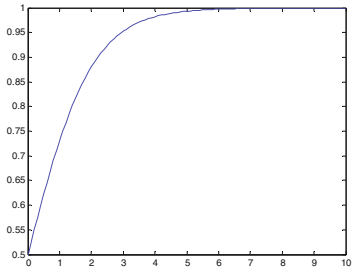

Fig. 1. Illustration for Eq. (3)

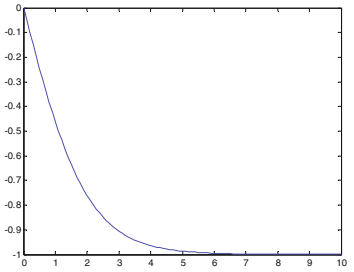

Fig. 2. Illustration for Eq. (5)

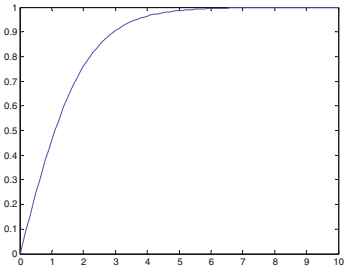

Fig. 3. Illustration for Eq. (7)

Algorithm 1 is the pseudocode of our discrete cuckoo search algorithm, where p_r is the predefined factor, and P_a is the probability of being discovered by the host bird.

Algorithm 1. Discrete cuckoo search algorithm

Begin
For each cuckoo, randomly initialize the position, the control factor p_r, the probability p_a of being discovered, Levy distribution $Step$;
Calculate the objective function values with Eq. (1) and Record the best position from swarm;
While (stop criterion is met)
If $rand() \le p_r$
Update the nest position for each cuckoo by strategy 1;
Else
If $Step \le 0$
Update each cuckoo with strategy 2;
Else
Update each cuckoo with strategy 3;
End
End
Evaluate the objective fitness for each cuckoo;
If $rand() > p_r$
Re-update the position of corresponding cuckoo with formulas:

$$x_{jk}^{t+1} = x_{jk}^{t'} + R \cdot (p_{gk}^{t'} - x_{uk}^{t'})$$

Evaluate the new cuckoos' finesses;
End
Record the best position
End
Output the best position
End

2.2 Local Search Strategy

For any partition (V_1, V_2), if the vertex j is moved from the current partition to another subset, the gain index g_j is defined as follows:

$$g_j = \begin{cases} \sum\limits_{\{j,k\} \in E, k \in V_1} w_{jk} - \sum\limits_{\{j,k\} \in E, k \in V_2} w_{jk}, j \in V_1 \\ \sum\limits_{\{j,k\} \in E, k \in V_2} w_{jk} - \sum\limits_{\{j,k\} \in E, k \in V_1} w_{jk}, j \in V_2 \end{cases} \tag{8}$$

Gain $g_j > 0$ means the vertex j should be moved with a lower objective function. With this manner, the following local search strategy is introduced:

Algorithm 2 Local Search Strategy

Begin

Use formula (8) to calculate the gain of each vertex

Descending all gains

Count the number (sel) of vertices with $g_j > 0$ is satisfied

If ($sel > 0$)

If sel >30

$$sel1 = \frac{1}{3} \cdot sel ;$$

Else

$sel1 = sel$;

Move the vertices of $sel1$, which are selected sequentially from sorted result.

Evaluate the fitness

End

End

2.3 Proposed Hybrid Algorithm

Our modification is a hybrid meta-heuristic method combining a discrete cuckoo search algorithm and local search strategy. The discrete cuckoo search is employed to find the approximate satisfied solution, while the local optimal solution is used to further improve the performance for the obtained approximate satisfied solution. Furthermore, to avoid the premature convergence, a mutation strategy is also employed to avoid the premature convergence. The pseudocode of our hybrid algorithm is listed in Algorithm 3. For each cuckoo, the mutation operation will randomly take 0.1% vertices to flip. We find that the algorithm will be improved after adding the mutation operation.

Algorithm 3 Cuckoo search with Local search

Begin

For the graph, initialise population pop , and record the best solution $gbest$;

Find the approximate satisfied solution by the discrete cuckoo search (refer to Alg.1);

Where (Stop condition is not verified)

the local strategy (refer to Alg.2) is employed to improve the quality of the solution;

If a particle x in a group is not improved in two consecutive searches, a mutation strategy is employed to avoid the premature convergence;

End

End

3 Performance Evolution

To test the performance of our proposed hybrid algorithm, G-set graph benchmarks are employed, and compared with the following algorithms:

- Hybridizing the cross-entropy method (HCE, in briefly) [26];
- A new lagrangian net algorithm (LNA, in briefly) [38];
- Discrete Hopfield network with estimation of distribution algorithm (DHNN-EDA, in briefly) [14]
- Discrete cuckoo search with local search (DCSLS, in briefly)

The program is implemented with MATLAB. In this set of instances, the number of vertex range from 800 to 3000, the control factor p_r is 0.3, the probability p_a are both set to 0.5, the total generation is 500.

For chosen benchmarks, each instance will run 50 times, Table 1 provides the optimal value achieved by the three algorithms and our proposed algorithm. The last line noted as "*w/t/l*" is the comparison results between our proposed DCSLS and its competitors. "*w/t/l*" represents our algorithm wins in *w* functions, ties in *t* functions, and loses in *l* functions. It means DCSLS is better than HCE for six functions, while only worse than HCE for three functions. DCSLS are superior than DHNN-EDA and LNA for six functions too, while DHNN-EDA and LNA only better than DCSLS with two and three functions, respectively. In one word, DCSLS achieves the best performance when compared with HCE, DHNN-EDA and LNA.

Table 1. Comparison of the results

Instances	Best	HCE	DHNN-EDA	LNA	DCSLS
G1	11624	11584	11614	11490	11607
G2	11620	11595	11599	11505	11599
G3	11622	11574	11617	11511	11605
G11	564	552	494	560	530
G12	556	542	476	546	528
G13	582	564	520	572	552
G14	3064	3030	3027	3023	3035
G15	3050	3012	2988	2996	3016
G16	3052	3015	3001	2994	3018
w/t/l		**6/0/3**	**6/1/2**	**6/0/3**	

To provide a deep comparison, two non-parametric statistics tests: Friedman test and Wilcoxon test, are employed to show the differences among these four algorithms. In Table 2, the ranking value is: DCSLS < HCE < DHNN-EDA < LNA, DCSLS maintains the lowest ranking, it means the performance of DCSLS is more better. Table 3 implies there is significantly difference between DCSLS and DHNN-EDA.

Table 2. Friedman test

	Rank-value
HCE	2.33
DCSLS	1.94
LNA	2.89
DHNN-EDA	2.83

Table 3. Wilcoxon test

DCSLS vs	P-value
HCE	0.594
LNA	0.192
DHNN-EDA	0.050

4 Conclusion

In this paper, a new hybrid algorithm combining with discrete cuckoo search and local search strategy is designed. The cuckoo update manner of discrete cuckoo search is the same as [37], while the local search strategy is designed. Simulation results show our modification achieves the best performance when compared with other three algorithms.

Acknowledgments. This research is supported by the Natural Science Foundation of Shanxi Province under No. 201601D011045.

References

1. Xia, Y., Xu, Z.: An efficient lagrangian smoothing heuristic for max-cut. Indian J. Pure Appl. Math. **41**(5), 683–700 (2010)
2. Marti, R., Duarte, A., Laguna, M.: Advanced scatter search for the max-cut problem. Inf. J. Comput. **21**(1), 26–38 (2009)
3. Wang, G.G., Deb, S., Gao, X.Z., Coelho, L.: A new metaheuristic optimization algorithm motivated by elephant herding behavior. Int. J. Bio-Inspired Comput. **8**(6), 394–409 (2016)
4. Bilbao, M.N., Ser, J.D., Salcedo-Sanz, S., Casanova-Mateo, C.: On the application of multi-objective harmony search heuristics to the predictive deployment of firefighting aircrafts: a realistic case study. Int. J. Bio-Inspired Comput. **7**(5), 270–284 (2015)
5. Rajakumar, R., Dhavachelvan, P., Vengattaraman, T.: A survey on nature inspired meta-heuristic algorithms with its domain specifications. In: International Conference on Communication and Electronics Systems, pp. 550–555 (2016)
6. Xiao, R., Zhang, Y., Huang, Z.: Emergent computation of complex systems: a comprehensive review. Int. J. Bio-Inspired Comput. **7**(2), 75–97 (2015)
7. Dorigo, M., Gambardella, L.M., Middendorf, M., Stutzle, T.: Special section on ant colony optimization. IEEE Trans. Evol. Comput. **6**(4), 317–320 (2002)
8. Stodola, P., Mazal, J.: Applying the ant colony optimisation algorithm to the capacitated multi-depot vehicle routing problem. Int. J. Bio-Inspired Comput. **8**(4), 228–233 (2016)
9. Zhang, Y.W., Wu, J.T., Guo, X., Li, G.N.: Optimising web service composition based on differential fruit fly optimisation algorithm. Int. J. Comput. Sci. Math. **7**(1), 87–101 (2016)
10. Eberhart, R.C., Shi, Y.H.: Special issue on particle swarm optimization. IEEE Trans. Evol. Comput. **8**(3), 201–203 (2004)

11. Adewumi, A.O., Arasomwan, M.A.: On the performance of particle swarm optimisation with(out) some control parameters for global optimisation. Int. J. Bio-Inspired Comput. **8**(1), 14–32 (2016)
12. Grillo, H., Peidro, D., Alemany, M., Mula, J.: Application of particle swarm optimisation with backward calculation to solve a fuzzy multi-objective supply chain master planning model. Int. J. Bio-Inspired Comput. **7**(3), 157–169 (2015)
13. Lv, L., Wu, L.Y., Zhao, J., Wang, H., Wu, R.X., Fan, T.H., Hu, M., Xie, Z.F.: Improved multi-strategy artificial bee colony algorithm. Int. J. Comput. Sci. Math. **7**(5), 467–475 (2016)
14. Sun, H., Wang, K., Zhao, J., Yu, X.: Artificial bee colony algorithm with improved special centre. Int. J. Comput. Sci. Math. **7**(6), 548–553 (2016)
15. Lu, Y., Li, R.X., Li, S.M.: Artificial bee colony with bidirectional search. Int. J. Comput. Sci. Math. **7**(6), 586–593 (2016)
16. Yu, G.: A new multi-population-based artificial bee colony for numerical optimization. Int. J. Comput. Sci. Math. **7**(6), 509–515 (2016)
17. Guo, Z.L., Wang, S.W., Yue, X.Z., Yin, B.Y., Deng, C.S., Wu, Z.J.: Enhanced social emotional optimisation algorithm with elite multi-parent crossover. Int. J. Comput. Sci. Math. **7**(6), 568–574 (2016)
18. Wang, H., Wang, W.J., Zhou, X.Y.: Firefly algorithm with neighborhood attraction. Inf. Sci. **382**, 374–387 (2017)
19. Wang, H., Wang, W.J., Sun, H.: Firefly algorithm with random attraction. Int. J. Bio-Inspired Comput. **8**(1), 33–41 (2016)
20. Yu, G.: An improved firefly algorithm based on probabilistic attraction. Int. J. Comput. Sci. Math. **7**(6), 530–536 (2016)
21. Nasiri, B., Meybodi, M.R.: History-driven firefly algorithm for optimisation in dynamic and uncertain environments. Int. J. Bio-Inspired Comput. **8**(5), 326–339 (2016)
22. Fister, I., Fister, I., Yang, X.S., Brest, J.: A comprehensive review of firefly algorithms. Swarm Evol. Comput. **13**, 34–46 (2013)
23. Yang, X.S., Gandomi, A.H.: Bat algorithm: a novel approach for global engineering optimization. Eng. Comput. **29**(5–6), 464–483 (2012)
24. Cai, X., Gao, X.Z., Xue, Y.: Improved bat algorithm with optimal forage strategy and random disturbance strategy. Int. J. Bio-Inspired Comput. **8**(4), 205–214 (2016)
25. Xue, F., Cai, Y., Cao, Y., Cui, Z., Li, F.: Optimal parameter settings for bat algorithm. Int. J. Bio-Inspired Comput. **7**(2), 125–128 (2015)
26. Laguna, M., Duarte, A., Marti, R.: Hybridizing the cross-entropy method: an application to the max-cut problem. Comput. Oper. Res. **36**(2), 487–498 (2009)
27. Lin, G., Zhu, W.: A discrete dynamic convexized method for the max-cut problem. Ann. Oper. Res. **196**(1), 371–390 (2012)
28. Festa, P., Pardalos, P.M., Resende, M.G.C., Ribeiro, C.C.: Randomized heuristics for the max-cut problem. Optim. Method Softw. **17**(6), 1033–1058 (2002)
29. Wang, J.: A memetic algorithm with genetic particle swarm optimization and neural network for maximum cut problems. In: Li, K., Fei, M., Irwin, G.W., Ma, S. (eds.) LSMS 2007. LNCS, vol. 4688, pp. 297–306. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74769-7_33
30. Wang, J., Zhou, Y., Yin, J.: Combining tabu hopfield network and estimation of distribution for unconstrained binary quadratic programming problem. Expert Syst. Appl. **38**(12), 14870–14881 (2011)
31. Lin, G., Guan, J.: An integrated method based on PSO and EDA for the max-cut problem. Comput. Intell. Neurosci. (2016). doi:10.1155/2016/3420671

32. Shylo, V.P., Shylo, O.V.: Solving the maxcut problem by the global equilibrium search. Cybern. Syst. Anal. **46**(5), 744–754 (2010)
33. Yang, X.S., Deb, S.: Cuckoo search via levy flights. In: World Congress on Nature and Biologically Inspired Computing, pp. 210–214 (2009)
34. Cui, Z.H., Sun, B., Wang, G.G., Xue, Y.: A novel oriented cuckoo search algorithm to improve DV-hop performance for cyber-physical systems. J. Parallel Distrib. Comput. **103**, 42–52 (2017)
35. Zhang, M.Q., Wang, H., Cui, Z.H., Chen, J.J.: Hybrid multi-objective cuckoo search with dynamical local search. Memetic Comp. (2017). doi:10.1007/s12293-017-0237-2
36. Li, F.X., Cui, Z.H., Sun, B.: DV-hop localisation algorithm with DDICS. Int. J. Comput. Sci. Math. **7**(3), 254–262 (2016)
37. Feng, D.K., Ruan, Q., Du, L.M.: Binary cuckoo search algorithm. J. Comput. Appl. **33**(6), 1566–1570 (2013). (in Chinese)
38. Xu, F.M., Ma, X.S., Chen, B.L.: A new lagrangian net algorithm for solving max-bisection problems. J. Comput. Appl. Math. **235**(13), 3718–3723 (2011)