Improved Random Key Cuckoo Search Optimization Algorithm for Community Detection in Social Networks

Randa Boukabene, Fatima Benbouzid-Si Tayeb and Narimene Dakiche
Laboratoire des Methodes de Conception de Systèmes (LMCS),
Ecole Nationale Supérieure d'Informatique (ESI),
BP 68M - 16270 Oued Smar, Alger, Algeria

Keywords: Social Networks, Community Detection, Cuckoo Search Optimization, Modularity, Random Key.

Abstract: Social network analysis is a prominent and thriving research field, with community detection being a particularly active area of study. In this study, we propose a cuckoo search-based approach for identifying the best network partitions by maximizing the modularity function. The proposed algorithm combines wisely the continuous nature of the standard cuckoo search algorithm with the discrete nature of the community detection problem to achieve the best results. Firstly, the algorithm incorporates the random key representation, which operates in a continuous space. This representation enables the algorithm to perform global and local walks, enabling both exploration and exploitation within the search space. Secondly, the algorithm utilizes the locus-based representation to handle the discrete aspect of the community detection problem. Experiments on both synthetic and real-world networks demonstrate the effectiveness and efficiency of our proposed algorithm.

1 INTRODUCTION

Recently, social networks, initially introduced by Wasserman and Faust (1994), have gained considerable popularity in research and development, partly due to more and more social media platforms such as Facebook, Instagram, and Twitter, being built online and the development of Web 2.0 applications. Social networks model social relationships by graph structures using vertices and edges. Vertices model individual social actors in a network, while edges model relationships between social actors.

Social networks have the characteristic property to exhibit a community structure, and highlighting the hidden structures of many real-world networks has drawn a wide range of researchers from many areas. As a result of the huge interest in the topic, a large number of surveys on community detection have been published (Souravlas et al., 2021; Khan and Niazi, 2017; Cai et al., 2016). Communities are formally defined as sub-graphs in which nodes are more strongly connected to each other than to the rest of the network (Girvan and Newman, 2002).

Various approaches have been proposed to address the community detection problem in social networks, as highlighted in the recent survey conducted by Bara'a et al. (2021) wherein researchers formulated the issue as a combinatorial optimization problem. This formulation enables the application of various optimization methods, including heuristics and metaheuristics, and functions for measuring the quality of partitioning in communities of a network have been proposed. Among them, modularity emerged as one of the most well-known and widely employed measures in this field (Newman and Girvan, 2004).

Nowadays, nature-inspired metaheuristic algorithms have gained significant popularity in the optimization field. The literature has been rich in regard to different optimization strategies for detecting communities by maximizing the aforementioned partition quality indicators (Bedi and Sharma, 2016; Su et al., 2022). In particular, metaheuristics, inspired by biological processes, have proven to be effective methods for tackling community detection problems in complex network instances (Osaba et al., 2020), with Evolutionary Algorithms as the most resorted optimization techniques in the last years (Pizzuti, 2017).

Cuckoo search algorithm (CSA) (Yang and Deb, 2009) is one of the recent nature-inspired algorithms used extensively to solve optimization problems in different fields of engineering (Yang and Deb, 2010). It is very effective in solving global optimization because it is able to maintain a balance between local and global random walks. Nonetheless, the appli-

Boukabene, R., Tayeb, F. and Dakiche, N.
Improved Random Key Cuckoo Search Optimization Algorithm for Community Detection in Social Networks.
DOI: 10.5220/0012202000003584
In *Proceedings of the 19th International Conference on Web Information Systems and Technologies (WEBIST 2023)*, pages 113-120
ISBN: 978-989-758-672-9; ISSN: 2184-3252

cation of the CSA to solve the community detection problem is very limited. Zhou et al. (2016) proposed a multi-objective discrete CSA with local search techniques. Their algorithm aimed at minimizing the ratio association (RA) and the ratio cut (RC) to enhance community quality. Shishavan and Gharehchopogh (2022) used a hybrid approach by combining genetic operators with the CSA to overcome local optima and improve its applicability to the community detection problem.

Using the cuckoo search algorithm directly for community detection can potentially diminish its effectiveness because the algorithm was primarily designed for continuous problems. This presents a challenge because community detection problems inherently possess a discrete nature. In light of this, the aim of this paper is to develop and assess the performance of Cuckoo search algorithm to discover communities in social networks. Firstly, the proposed algorithm incorporates the random key representation, which operates in a continuous space. This representation allows performing global and local walks, enabling both exploration and exploitation within the search space. Secondly, the algorithm utilizes the locus-based representation to meet the requirements of the discrete nature of the network community detection problem.

The structure of the paper is as follows: Section 2 gives the community detection problem description. Section 3 describes the proposed cuckoo search algorithm. The results of experiments conducted on synthetic and real networks are presented in Section 4. Finally, Section 5 concludes the paper and discusses possible future work.

2 PROBLEM FORMULATION

A social network can be modeled by a graph, denoted $G = (V,E)$, which includes $N = |V|$ nodes and $m = |E|$ edges. V represents the set of nodes and E represents the set of edges connecting two nodes in G. A is an $N \times N$ adjacency matrix of the network, where A_{ij} represents the element at the i^{th} row and j^{th} column of A. If there is an edge connecting nodes i and j, $A_{ij} = 1$; otherwise $A_{ij} = 0$.

The problem of community detection requires the partitioning of the network into sub-clusters or communities. This partitioning is denoted as $C = \{C_1, ...C_k\}$, where each element C_l $l = 1, 2, ..., k$ is a proper subset of V, and k is the total number of communities. Moreover, for any two communities C_i and $C_j \in C$ and $i \neq j$, $V_i \cap V_j = \phi$. The quality of the communities obtained is often measured by the so-called modularity function Q (Newman and Girvan, 2004). Q is a value between -1 and 1 that measures the density of the links within the communities compared to the links between the communities; it can also be defined as the difference between the edges of the network connecting vertices of the same type and the expected value of the same quantity in a network with the same community divisions but random links between vertices. It is computed according to equation 1:

$$Q = \frac{1}{2m} \sum_{ij} [A_{ij} - \frac{k_i k_j}{2m}] \delta(c_i, c_j) \quad (1)$$

Where A_{ij} represents the adjacency matrix, $k_i(k_j)$ is the degree of node $i(j)$, $c_i(c_j)$ is the community of node $i(j)$, m is the sum of all the links in the graph. δ is the Kronecker delta function $\delta(x,y) = 1$ if $x = y$, 0 otherwise.

3 PROPOSED CUCKOO SEARCH BASED SOLVING APPROACH

Cuckoo search algorithm (CSA) (Yang and Deb, 2009) is a metaheuristic optimization algorithm, inspired by the brood parasitism behavior of some of a bird family called cuckoo (Yildiz, 2013). Cuckoo birds lay their eggs in another nest, called the host nest where they imitate the color and patterns of the host eggs to prevent them from being discovered. if the host bird discovers them, the cuckoo eggs can be thrown or the host bird leaves its nest. To make the algorithm suitable for solving optimization problems, the eggs of the nest are associated with a set of candidate solutions for a given optimization problem. In particular, CSA is based on three idealized rules:

- Each cuckoo lays one egg at a time, and deposits it in a randomly selected nest.
- The best nests containing high-quality eggs will be passed on to subsequent generations.
- The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with probability *Pa*. In this case, the host bird can either discard the egg or simply abandon the nest and build a brand-new one.

This algorithm follows a simple approach where each nest is associated with a single egg, and each cuckoo can lay only one egg (thus representing one solution). We can safely make no difference between an egg, a nest, or a cuckoo. The aim is to use the new and potentially better solution (cuckoo egg) to replace existing solutions in the nests.

In the proposed algorithm, hereinafter RKCSA for random key CSA, the nests represent the current solutions or candidate solutions represented with a new structure combining both the locus representation and a random key representation. The eggs or nests correspond to the new solutions generated by both global and local walks. The algorithm evaluates these new solutions using the modularity function based on the locus representation and replaces the existing solutions in the nests with the improved ones. By continuously generating and evaluating new solutions for the locus vector, RKCSA aims to enhance the quality of the solutions over time. The replacement process ensures that only better solutions are retained in the nests, leading to the continual improvement of the overall solution quality. It is worth noting that the random key representation, depicting the continuous nature of the algorithm, also plays a crucial role in guiding the search and exploration process. However, the replacement of solutions in the nests specifically focuses on the locus representation to improve the community detection results. By combining the exploration capabilities of the random key representation and the refinement process of the locus representation, the proposed algorithm aims to achieve better performance in optimizing the modularity function and identifying the best network partitions. The population of nests is subjected to repeated cycles until the stop criterion is satisfied.

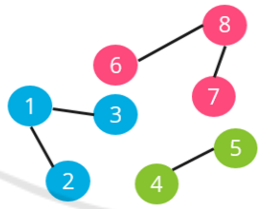
Next, we will delve into a comprehensive exploration of the proposed algorithm, providing a detailed explanation of its step-by-step process in algorithm 1.

3.1 Solution Encoding and Decoding

The implementation of RKCSA at hand begins with egg encoding to generate a random initial population. A candidate solution X, i.e. an egg, (Figure 1a) is coded by a two-field structure where:

- The first field is the locus vector (Figure 1b) that uses the locus-based representation (Pizzuti, 2008). Each individual in the population consists of N genes $\{g_1, ..., g_n\}$ equivalent to the number of nodes in the network. Each gene g_u in an individual is assigned a value v, which represents one of the adjacent nodes of node u. In this representation, it is necessary to perform a decoding step to identify all the components of the graph, so that the nodes participating in the same component are assigned to the same community (Pizzuti, 2008). This decoding step can be performed in linear time using the method described in (Cormen et al., 2022).
- The second field is the key vector (Figure 1c) that takes a random value between 0 and 1 associated with each node in the network. This random key representation enables the algorithm to operate in a continuous space, facilitating smooth transitions between solutions and allowing for efficient exploration of the search landscape. This continuous aspect is crucial in navigating the complex and high-dimensional solution space of community detection problems.

To generate the initial solutions, we need to initialize both the locus-based vector and the random key vector. For the locus vector initialization, we assign to a node one of its neighbors. On the other hand, to initialize the key vector, we assign a random number between 0 and 1 to each node in the network. According to this representation, prior knowledge of the number of communities is not needed.



(a) Graph partitions.

	1	2	3	4	5	6	7	8
Locus	3	1	1	5	4	8	8	7

(b) Locus vector.

	1	2	3	4	5	6	7	8
Key	0.28	0.55	0.83	0.36	0.84	0.62	0.02	0.45

(c) Key vector.

Figure 1: Example of a solution representation.

3.2 Global Walk

The first component of the cuckoo search algorithm known as the Lévy flights or global walk generates new solutions in proximity to the best solution found so far (Reda et al., 2022). In the proposed algorithm, we need to create both a new key vector and a new locus vector to generate a new cuckoo. The generation of a new key vector is done using the Lévy flights' distribution according to equation 2.

$$x^i_{key}(t+1) = x^i_{key}(t) + \alpha s(x^i_{key}(t) - x^{best}_{key}(t)) \quad (2)$$

Where $x^i_{key}(t)$, $x^i_{key}(t+1)$ and x^{best}_{key} are the current, the new, and the best key solution, respectively. s is the step size while α is the scale of the step size.

In Mantegna's algorithm (Mantegna, 1994), the step size s can be calculated using two Gaussian distributions u and v via the transformation of equation 3.

$$s = \frac{u}{|v|^{\frac{1}{\beta}}} \tag{3}$$

With

$$u \sim N(0, var(u)) \;\; et \;\; v \sim N(0, var(v)) \tag{4}$$

And

$$var(u) = [\frac{\Gamma(1+\beta)}{\beta\Gamma((1+\beta)/2)} \frac{\sin(\pi\beta/2)}{2^{(\beta-1)/2}}]^{1/\beta}, \quad var(v) = 1 \tag{5}$$

Where u and v are random numbers from a normal distribution, $N(\Gamma)$ is the normal(gamma) distribution. $var(u(v))$is the variance of $u(v)$ is the gamma distribution, and $\beta = 1.5$.

To generate a new locus vector as shown in figure 2, we perform the following steps:

1. Calculate the difference between the old and new key vectors $|x^i_{key}(t+1) - x^i_{key}(t)|$.
2. If the difference is greater than the acceptance rate $|x^i_{key}(t+1) - x^i_{key}(t)| > \delta$, we reinitialize the corresponding node with the neighbor of the best solution so far $x^{best}_{locus}(t)$. Otherwise, the node's value remains the same as the previous iteration.
3. The newly generated solution is evaluated using the modularity measure (Eq.1) on the locus vector, and if it has greater modularity than the current solution, then it replaces the current solution for this nest.

3.3 Local Walk

The second component of the cuckoo search algorithm is the abandon operator. Its purpose is to generate solutions that are far from the best solution obtained. This is done to prevent the search from getting trapped in local optima and to encourage exploration of different regions of the search space. By introducing diversity into the search process, the algorithm increases the chances of finding the global optimum or better solutions (Reda et al., 2022).

In the abandon operator, the host bird has a probability *Pa* of discovering foreign eggs in its nest. If foreign eggs are found, the host bird will abandon them and replace them with new eggs. In RKCSA, this process is applied to the key vector, where foreign solutions are replaced with new solutions generated using equation 6.

$$x^i_{key}(t+1) = x^i_{key}(t) + S * K \tag{6}$$

With

$$K = \begin{cases} 1 & if \;\; rand > Pa \\ 0 & Otherwise \end{cases} \tag{7}$$

And

$$S = (x^{randperm(n)}_{key}(t) - x^{randperm(n)}_{key}(t)) \tag{8}$$

Where *rand* is a random number within [0,1], *Pa* is the discovery probability and $randperm(n)$ is the permutation function that chooses a random number in the range [0,n].

After generating a new key vector, the locus vector needs to be updated by following these steps:

1. Calculate the difference between the old and new key vectors $|x^i_{key}(t+1) - x^i_{key}(t+1)|$;
2. If the difference is greater than the acceptance rate (i.e, $|x^i_{key}(t+1) - x^i_{key}(t+1)| > \delta$), then the corresponding node in the locus vector is reinitialized by one of its neighbors, unlike the global walk, whose value is reinitialized by the neighbor of the best solution. Otherwise, the node's value remains the same as in the previous step;
3. The newly generated solution is evaluated using the modularity measure (Eq.1) on the locus vector, and if it has greater modularity than the current solution, then it replaces the current solution for this nest.

4 EXPERIMENTAL RESULTS AND DISCUSSION

To assess the efficiency and performance of our proposed algorithm, RKCSA, we conducted a comprehensive set of experiments on both synthetic and real-world networks using a personal computer running Windows 10 Enterprise, equipped with 8 GB of RAM and powered by an Intel(R) Core(TM) i7-3537u CPU with a clock speed of 2.50 GHz.

For the synthetic networks, we used the well-known LFR benchmark (Lancichinetti et al., 2008), which allows us to generate networks with various parameters. In our experiments, we focused on varying the mixing parameter μ from 0 to 0.8, with an interval of 0.05. This deliberate variation was intended to create a scenario where the boundaries between communities became increasingly blurred, thereby making the task of community detection significantly more

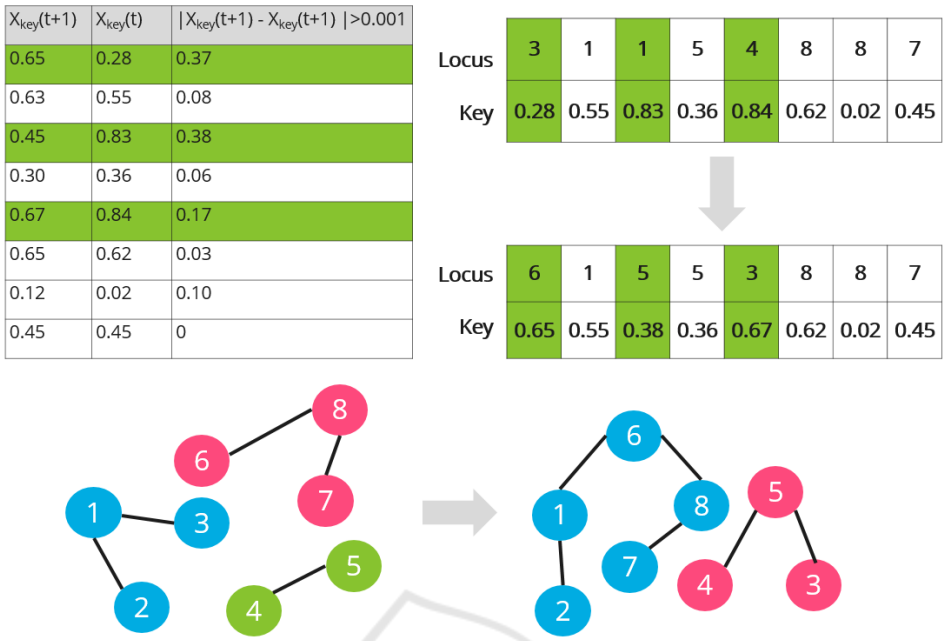


Figure 2: An example of the locus vector generation.

challenging. Then, we created two groups of networks with different community sizes by specifying the minimum and maximum number of communities.

For real-world networks, we also employed a diverse set of 11 real-world networks from various domains with diverse sizes and characteristics. Details are presented in Table 1.

Table 1: Real-world networks.

Network	Nodes	Edges
Karate	34	78
Dolphin	62	159
Polbooks	105	441
Football	115	613
Lesmis	77	254
Jazz	198	2742
Metabolic	453	2040
08blocks	300	584
Neural	297	2148
Polblogs	1490	16718
Netscience	1589	2742

In order to evaluate the performance of RKCSA algorithm, we compared it with six other algorithms namely: GN (Girvan and Newman, 2002), CNM (Blondel et al., 2008), Louvain (Blondel et al., 2008), BCD (Binary Swarm Optimization) (Beldi and Bessedik, 2019), ABC (Artificial Bee Colony) (Dakiche et al., 2022), and CSA (The standard Cuckoo Search Algorithm).

4.1 Evaluation Metrics

Given the fact that we have two types of networks, with and without the ground truth community structure, we adopt two widely used criteria to evaluate the accuracy of community detection algorithms. For the synthetic networks where the ground truth community structure is known, we used the Normalized Mutual Information (NMI) (Danon et al., 2005) metric to assess the similarity between the detected communities and the ground truth. The NMI is calculated according to equation 9.

$$NMI(A,B) = \frac{-2\sum_{i=1}^{c_A}\sum_{j=1}^{c_B} C_{ij}log(C_{ij}N/C_{i.}C_{.j})}{\sum_{i=1}^{c_A} C_{i.}log(C_{i.}/N) + \sum_{j=1}^{c_B} C_{.j}log(C_{.j}/N)} \quad (9)$$

Where A and B are two communities in the network. C_{ij} is the number of nodes in community $A_i \in A$ that are also in community $B_j \in B$. $c_A(c_B)$ is the number of groups in $A(B)$. $C_{i.}$ and $C_{.j}$ are the sum of the elements of C in row i and column j respectively. N is the number of nodes.

For real-world networks, where the ground truth community structure is not available, we used the Modularity function (Eq 1) to measure the quality of the detected communities.

4.2 RKCSA's Parameters Tuning

The most important cuckoo search algorithm includes three parameters: Pa (abandonment probability for

worse nests), step size scale (α), and acceptance rate (δ). Among these parameters, *Pa* and α play a crucial role in achieving improved solutions. Therefore, in this paper, we have undertaken a sensitivity analysis of performance for RKCSA to examine the effects of varying the values of five parameters: population size (n), the maximum number of iterations (*maxGen*), and the CSA parameters *Pa*, α, and δ. After a deep analysis of the results, we observed that the performance of the algorithm was sensitive to the choice of parameter values. For the population size (n), we found that values between 150 and 200 led to the best outcomes. Regarding the maximum number of iterations (*maxGen*), we observed that values between 800 and 1000 were optimal. The step size scale (α) was found to be most effective when set to 0.1. For the acceptance rate (δ), we obtained the best results when it was set to 0.1. The complete details are not reported for the sake of a concise presentation.

Algorithm 1: Random key cuckoo search algorithm for community detection.

Data: A network $G = (V, E)$
Result: Community structure of the network $C = \{C_1, C_2, ..., C_k\}$
Initialize the algorithm parameters n (population size), *maxGen* (maximum number of iterations), *Pa* (discovery probability), α (step size scale), δ (acceptance rate).
Generate an initial population of n hosts with both key vector and locus vector x_{key}, x_{locus}, respectively.
while $t < maxGen$ **do**
  Generate cuckoos by global walk.
  Evaluate the quality of the cuckoo solution $Q(x^i_{locus})$.
  Choose a nest j among n nests at random.
  if $(Q(x^j_{locus}) > Q(x^i_{locus}))$ **then**
    Replace cuckoo i with the new cuckoo j.
  end
  A fraction *Pa* of the nests are abandoned.
  New nests are generated by local walk.
  Keep the best solutions (nests with quality solutions).
  Rank the solutions and find the best current solutions.
  Update $t \leftarrow t + 1$.
end

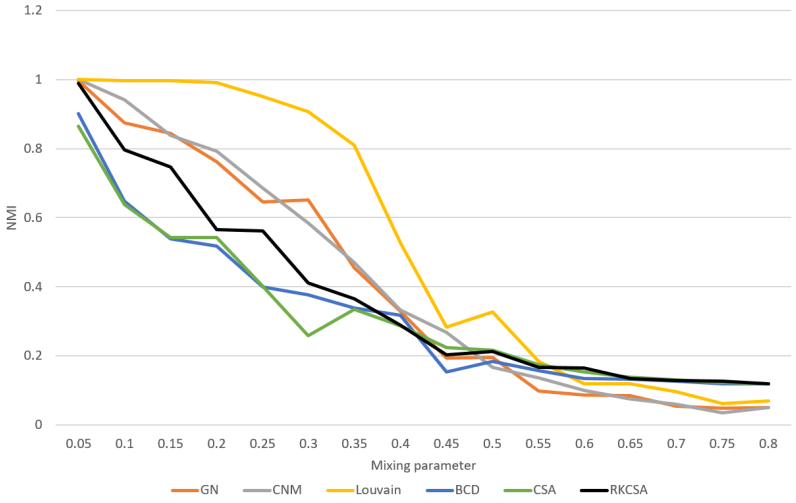
4.3 RKCSA Performance Analysis on Synthetic Networks

Figure 3 presents the experimental results on LFR benchmarks. It shows the performance of different algorithms in detecting the real communities for various values of the mixing parameter, μ. Plots (a) and (b) in Figure 3 correspond to the same network size with two different community sizes (small 20-50 and big 30-100). For $\mu = 0.05$, most of the algorithms are able to accurately detect the real communities. However, as the value of μ increases beyond 0.05, the algorithms start to struggle in identifying the real communities, except for the Louvain algorithm which maintains its effectiveness until $\mu = 0.2$. From $\mu = 0.25$ to $\mu = 0.5$, the Louvain algorithm consistently outperforms the other algorithms in detecting real communities. In terms of the metaheuristic algorithms, RKCSA demonstrates better performance than the other metaheuristics for μ values less than 0.5. It is able to maintain its ability to detect real communities effectively. However, for μ values greater than 0.5, we observe that the CSA, BCD, and RKCSA algorithms converge towards similar results and outperform the other algorithms (CNM, GN, and Louvain) in terms of community detection accuracy.

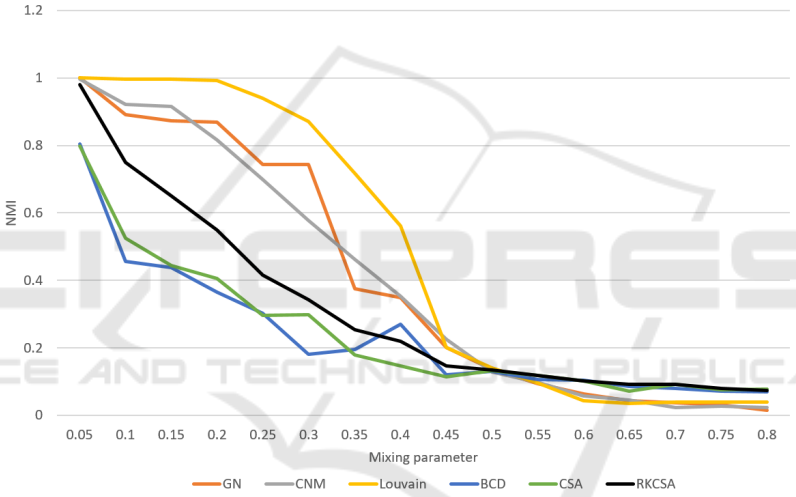
These results highlight the varying performance of the algorithms based on the value of the mixing parameter, with some algorithms performing better in specific ranges of μ. It also indicates the effectiveness of RKCSA and other metaheuristic algorithms in community detection tasks, particularly for certain parameter settings.

4.4 RKCSA Performance Analysis on Real Networks

Table 2 presents the experimental results of modularity on real-world networks. The numbers in bold indicate the highest modularity values in each corresponding row. It is observed that the RKCSA algorithm achieves optimal modularity results for 7 out of 11 real-world networks, accounting for 63.64% of the networks. On the other hand, the Louvain algorithm obtains the best results for 5 networks, which corresponds to 45.45% of the networks. Furthermore, RKCSA outperforms GN, CNM, ABC, BCD, and CSA algorithms on 10 out of the 11 networks, making it a competitive method for the community detection problem. These results highlight the effectiveness of RKCSA in achieving high modularity values and its capability to accurately identify community structures in various real-world networks.



(a) 20-50



(b) 30-100

Figure 3: NMI results on LFR benchmarks with $n = 500$. (a) Small size communities. (b) Big-size communities.

Table 2: Comparison of modularity results on real-world networks.

Network	GN	CNM	Louvain	BCD	ABC	CSA	RKCSA
Karate	0.4013	0.3807	**0.4198**	**0.4198**	**0.4198**	0.4188	**0.4198**
Dolphin	0.5194	0.4955	0.5185	0.5253	**0.5285**	0.4572	**0.5285**
Polbooks	0.5168	0.5020	0.5270	0.5189	0.5116	0.4915	**0.5285**
Football	0.5996	0.5497	0.6043	0.5146	0.6009	0.3671	**0.6044**
Lesmis	0.5381	0.5006	0.5548	0.5430	0.5594	0.4879	**0.5600**
Jazz	0.4051	0.4389	**0.4438**	0.3120	0.4384	0.2392	0.4426
Metabolic	0.4048	0.4172	**0.4405**	0.3128	0.3664	0.2569	0.4055
08blocks	0.8599	**0.8750**	**0.8750**	0.8104	**0.8750**	0.8749	**0.8750**
Neural	0.3010	0.3728	0.3926	0.2319	0.3137	0.1901	**0.4254**
Polblogs	0.4180	**0.4270**	0.4269	0.3642	0.3702	0.2234	0.4234
Netscience	0.9579	0.9551	**0.9592**	0.9006	0.9086	0.8909	0.9500

5 CONCLUSIONS

In this paper, we presented a novel cuckoo search algorithm, RKCSA, for community detection in social networks. Where we proposed a new solution representation that combines the locus and random key representations of the network to enhance its search ability. Experiments on both synthetic and real-world networks show that RKCSA can accurately and effectively uncover the community structure. We also demonstrated the superior performance of RKCSA compared to the standard CSA algorithm. However, in real-life networks, we can find multiple relationships between a couple of nodes. Therefore, we aim to extend our algorithm to handle multilayer networks.

REFERENCES

Bara'a, A. A., Abbood, A. D., Hasan, A. A., Pizzuti, C., Al-Ani, M., Özdemir, S., and Al-Dabbagh, R. D. (2021). A review of heuristics and metaheuristics for community detection in complex networks: Current usage, emerging development and future directions. *Swarm and Evolutionary Computation*, 63:100885.

Bedi, P. and Sharma, C. (2016). Community detection in social networks. *Wiley interdisciplinary reviews: Data mining and knowledge discovery*, 6(3):115–135.

Beldi, Z. and Bessedik, M. (2019). A new brainstorming based algorithm for the community detection problem. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 2958–2965. IEEE.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Cai, Q., Ma, L., Gong, M., and Tian, D. (2016). A survey on network community detection based on evolutionary computation. *International Journal of Bio-Inspired Computation*, 8(2):84–98.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2022). *Introduction to algorithms*. MIT press.

Dakiche, N., Benatchba, K., Tayeb, F. B.-S., Slimani, Y., and Brahmi, M. A. (2022). A hybrid artificial bee colony algorithm with simulated annealing for enhanced community detection in social networks. In *2022 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 104–108. IEEE.

Danon, L., Diaz-Guilera, A., Duch, J., and Arenas, A. (2005). Comparing community structure identification. *Journal of statistical mechanics: Theory and experiment*, 2005(09):P09008.

Girvan, M. and Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826.

Khan, B. S. and Niazi, M. A. (2017). Network community detection: A review and visual survey. *arXiv preprint arXiv:1708.00977*.

Lancichinetti, A., Fortunato, S., and Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110.

Mantegna, R. N. (1994). Fast, accurate algorithm for numerical simulation of levy stable stochastic processes. *Physical Review E*, 49(5):4677.

Newman, M. E. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113.

Osaba, E., Del Ser, J., Camacho, D., Bilbao, M. N., and Yang, X.-S. (2020). Community detection in networks using bio-inspired optimization: Latest developments, new results and perspectives with a selection of recent meta-heuristics. *Applied Soft Computing*, 87:106010.

Pizzuti, C. (2008). Ga-net: A genetic algorithm for community detection in social networks. In *Parallel Problem Solving from Nature–PPSN X: 10th International Conference, Dortmund, Germany, September 13-17, 2008. Proceedings 10*, pages 1081–1090. Springer.

Pizzuti, C. (2017). Evolutionary computation for community detection in networks: A review. *IEEE Transactions on Evolutionary Computation*, 22(3):464–483.

Reda, M., Onsy, A., Elhosseini, M. A., Haikal, A. Y., and Badawy, M. (2022). A discrete variant of cuckoo search algorithm to solve the travelling salesman problem and path planning for autonomous trolley inside warehouse. *Knowledge-Based Systems*, 252:109290.

Shishavan, S. T. and Gharehchopogh, F. S. (2022). An improved cuckoo search optimization algorithm with genetic algorithm for community detection in complex networks. *Multimedia Tools and Applications*, 81(18):25205–25231.

Souravlas, S., Sifaleras, A., Tsintogianni, M., and Katsavounis, S. (2021). A classification of community detection methods in social networks: a survey. *International Journal of General Systems*, 50(1):63–91.

Su, X., Xue, S., Liu, F., Wu, J., Yang, J., Zhou, C., Hu, W., Paris, C., Nepal, S., Jin, D., et al. (2022). A comprehensive survey on community detection with deep learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Wasserman, S. and Faust, K. (1994). Social network analysis: Methods and applications.

Yang, X.-S. and Deb, S. (2009). Cuckoo search via lévy flights. In *2009 World congress on nature & biologically inspired computing (NaBIC)*, pages 210–214. IEEE.

Yang, X.-S. and Deb, S. (2010). Engineering optimisation by cuckoo search. *International Journal of Mathematical Modelling and Numerical Optimisation*, 1(4):330–343.

Yildiz, A. R. (2013). Cuckoo search algorithm for the selection of optimal machining parameters in milling operations. *The International Journal of Advanced Manufacturing Technology*, 64:55–61.

Zhou, X., Liu, Y., and Li, B. (2016). A multi-objective discrete cuckoo search algorithm with local search for community detection in complex networks. *Modern Physics Letters B*, 30(07):1650080.