Flattening Based Cuckoo Search Optimization Algorithm for Community Detection in Multiplex 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: Community Detection, Flattening, Cuckoo Search Optimization, Multiplex Networks, Modularity.

Abstract: Complex network analysis is a thriving research field, with a particular focus on community detection. This paper addresses the challenge of community detection in multiplex networks, which model multiple types of relationships to reflect reality. Our approach consists of two key steps. First, we employ multiplex network flattening techniques to transform it into a one-dimensional network. Second, we introduce a cuckoo search-based algorithm to maximize the modularity function and identify the best network partitions. Our algorithm strategically combines the continuous aspects of the standard cuckoo search algorithm with the discrete nature of community detection, to achieve better results. Experiments on both synthetic and real-world multiplex networks demonstrate the efficiency and effectiveness of our approach.

1 INTRODUCTION

Real-life networks often involve various types of connections between entities. For instance, human relationships can encompass family ties, business associations, social interactions, and more. To model such complex networks, researchers have introduced the concept of multiplex networks (Boccaletti et al., 2014). A multiplex network comprises several individual layers, with each layer representing a distinct type of connection or relationship (Kivelä et al., 2014). However, the challenge arises as each layer in a multiplex network possesses its unique community structures, potentially missing the comprehensive composite community (Pizzuti, 2017). This complexity contrasts with the relatively straightforward community structure in traditional networks, where subgraphs highlight nodes with stronger internal connections than with the rest of the network (Girvan and Newman, 2002).

Despite the community detection problem being one of the main topics of social network analysis for over a decade, only recently it has been tackled in the context of complex multidimensional systems (Gong et al., 2013; Shen and Cheng, 2010). Several recent surveys have suggested categorizing existing approaches that handle the presence of multiple layers into three main classes (Huang et al., 2021; Roozbahani et al., 2021; Magnani et al., 2021): (1) Flattening techniques that combine all layers into one, followed by the application of traditional network analysis methods. (Loe and Jensen, 2015); (2) Assembly techniques that detect communities in each layer, and then find a consensus between them (Harakawa et al., 2019); and (3) Direct detection techniques that identify multiplex community structures without layer simplification (Mucha et al., 2010).

In this paper, we propose a two-phase community detection approach in multiplex networks. In the first phase, we draw inspiration from the work of Berlingerio et al. (2011) to apply flattening techniques. In the second phase, we apply an optimization algorithm for community detection to the flattened network. To achieve robust results, we introduce a novel algorithm called Cuckoo Search Algorithm (CSA) (Yang and Deb, 2009). CSA is a relatively recent nature-inspired optimization technique known for its effectiveness in global optimization tasks (Yang and Deb, 2010). However, its direct application to community detection is limited. To overcome this limitation, we incorporate a random key representation that operates within a continuous space. This allows for global and local exploration within the search space, striking a balance between exploration and exploitation. Additionally, we introduce a locus-based representation to accommodate the discrete nature of network community detection needs.

The paper is organized as follows: Section 2 de-

Boukabene, R., Benbouzid-Si Tayeb, F. and Dakiche, N.
Flattening Based Cuckoo Search Optimization Algorithm for Community Detection in Multiplex Networks.
DOI: 10.5220/0012363600003636

In *Proceedings of the 16th International Conference on Agents and Artificial Intelligence (ICAART 2024) - Volume 3*, pages 519-526
ISBN: 978-989-758-680-4; ISSN: 2184-433X

scribes the community detection problem in multiplex networks. Section 3 outlines the proposed algorithm. The experimental results on synthetic and real networks are presented in Section 4. Finally, Section 5 concludes the paper and explores potential future research directions.

2 MULTIPLEX NETWORKS AND COMMUNITIES

A multiplex network (Boccaletti et al., 2014) is a series of graphs $G = \{G_\alpha; \alpha \in \{1,...,d\}\}$ where $G_\alpha = (V, E_\alpha)$ represents the α^{th} layer of G, and E_α represents the α^{th} type of connection among the same set of nodes V. G can be represented by an adjacency matrix $A^{[\alpha]} = (a_{ij}^\alpha) \in \mathcal{R}^{N\times N}$ for each layer G_α, where:

$$a_{ij}^\alpha = \begin{cases} 1 & \text{if}(v_i^\alpha, v_j^\alpha) \in E_\alpha \\ 0 & \text{else} \end{cases} \quad (1)$$

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 $i \neq j$, $V_i \cap V_j = \phi$.

3 PROPOSED SOLVING APPROACH

The proposed process of community detection in multiplex networks using flattening techniques involves two distinct phases (Figure 2). Initially, the multiplex network is consolidated into a mono-dimensional network. Subsequently, we apply an optimization mono-dimensional community detection algorithm.

In what follows, we will delve into a comprehensive exploration of the proposed algorithm, providing a detailed explanation of its two phases.

3.1 Flattening Techniques

Numerous approaches have been developed for community detection in multiplex networks, including the flattening technique, which transforms a multiplex network G_m into a mono-dimensional network (traditional network) G while preserving as much information as possible. Berlingerio et al. (2011) propose three flattening techniques as follows:

- The first technique, referred to as f_1, assigns a weight of 1 to the edge E_{ij} in G if there exists at least one dimension connecting nodes i and j in G_m (Figure 2a) (Eq2).

$$w_{ij} = \begin{cases} 1 & \text{if } \exists d : (i,j,d) \in E \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

- The second technique, denoted as f_2, takes into account the count of dimensions connecting any pair of nodes, i and j, and uses this count as the weight for the newly added mono-dimensional edge (Berlingerio et al., 2011) (Figure 2b) (Eq3).

$$w_{ij} = |d : (i,j,d) \in E| \quad (3)$$

- The third technique, designated as f_3, extends its consideration to the neighbors of nodes i and j, rather than solely focusing on the connection between these nodes. This modification is grounded in the assumption that common neighbors are likely to belong to the same community as nodes i and j (Berlingerio et al., 2011) (Figure 2c)(Eq4).

$$w_{ij} = \frac{|N_i \bigcap N_j|}{|N_i \bigcup N_j| - 2} \quad (4)$$

Where $N_.$ is the set of neighbors in dimension d for a node.

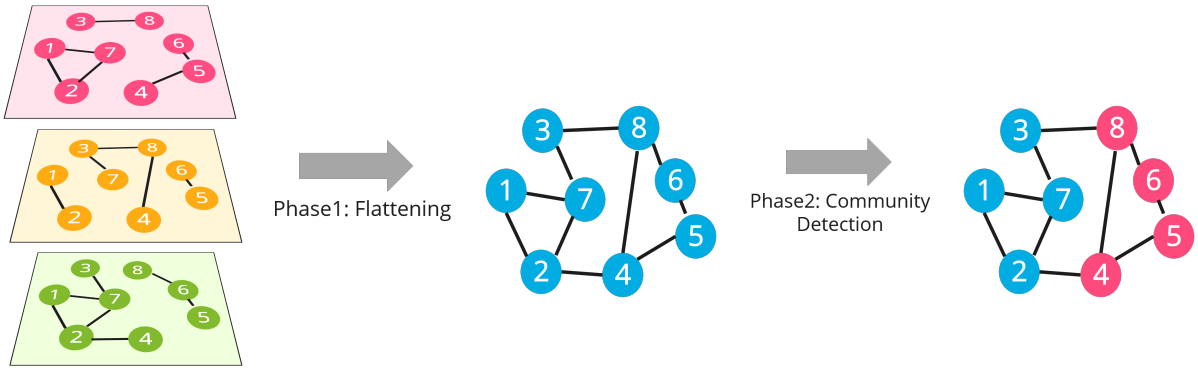
3.2 Cuckoo Search Optimization

The Cuckoo Search Algorithm (CSA), introduced by Yang and Deb (2009), is a nature-inspired metaheuristic optimization algorithm. It draws inspiration from the brood parasitism behavior observed in certain bird species, notably the cuckoo (Yildiz, 2013). In the proposed algorithm, called FCSA for flattening-based CSA, candidate solutions are represented using a novel structure that combines both the locus and random key representations, using nests or eggs. These solutions are created through a combination of global and local walks. Subsequently, the algorithm assesses their quality using the modularity function, leaning on the locus representation, and replaces the existing solutions in the nests with any improved ones. The random key representation serves as a vital guide in steering the search and exploration processes, maintaining the algorithm's continuous nature. However, when it comes to replacing solutions within the nests, the algorithm places a specific emphasis on the locus representation to enhance community detection results.

Next, we will delve into a comprehensive exploration of the proposed algorithm.

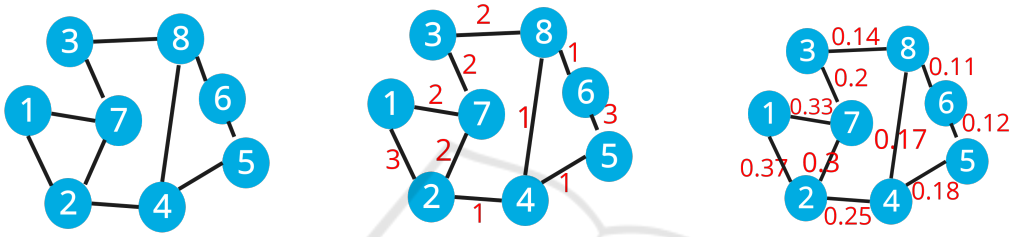
3.2.1 Solution Representation and Objective Function

FCSA starts with solution encoding to create a random initial population. A candidate solution X is en-



(a) Multiplex network. (b) Flattened network. (c) Graph partitions.

Figure 1: Proposed flattening-based process of community detection in multiplex networks.



(a) Flattening function f_1. (b) Flattening function f_2. (c) Flattening function f_3.

Figure 2: Multiplex network (1a) flattening techniques.

coded and initialized using two-field structure as illustrated in Figure 3, where:

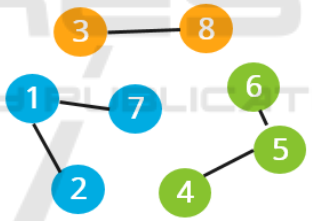
- The first field is the locus vector (Figure 3b) that uses the locus-based representation (Pizzuti, 2008). In this representation, Each node u within an individual is associated with a value v, representing one of its adjacent nodes. The decoding step can be accomplished in linear time (Cormen et al., 2022) (Pizzuti, 2008).
- The second field corresponds to the key vector (Figure 3c), where each node in the network is associated with a random value ranging from 0 to 1.

Modularity Q is by far the most used and best-known quality function for measuring the quality of a partition of a network (Newman and Girvan, 2004)

For each graph G, with n nodes and m edges, modularity is computed as follows:

$$Q = \frac{1}{2m}\sum_{ij}[A_{ij} - \frac{k_i k_j}{2m}]\delta(c_i, c_j) \quad (5)$$

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



(a) Graph partitions.

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

(b) Locus vector.

	1	2	3	4	5	6	7	8
Key	0.45	0.56	0.13	0.72	0.91	0.23	0.06	0.81

(c) Key vector.

Figure 3: Example of a solution representation (Boukabene. et al., 2023).

3.2.2 Global Walk

In the proposed algorithm, the first component, known as Lévy flights or global walk, is responsible for generating new solutions near the best solution found so far (Reda et al., 2022) (Boukabene. et al.,

2023). To create a new cuckoo, both a new key vector and a new locus vector are generated. The new key vector is produced using the Lévy flights' distribution, as defined in equation 6.

$$x^i_{key}(t+1) = x^i_{key}(t) + \alpha s(x^i_{key}(t) - x^{best}_{key}(t)) \quad (6)$$

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 7.

$$s = \frac{u}{|v|^{\frac{1}{\beta}}} \quad (7)$$

With $u \sim N(0, var(u))$ *et* $v \sim N(0, var(v))$

And

$$var(u) = [\frac{\Gamma(1+\beta)}{\beta\Gamma((1+\beta)/2)} \frac{\sin(\pi\beta/2)}{2^{(\beta-1)/2}}]^{1/\beta}, \quad var(v) = 1 \quad (8)$$

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, we calculate the difference between the old and new value of the key vector $|x^i_{key}(t+1) - x^i_{key}(t)|$. If the difference is greater than the acceptance rate δ, 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.

The newly generated solution is evaluated using the modularity measure (Eq. 5) on the locus vector. If the new solution exhibits higher modularity than the current solution, it replaces the current solution within this nest.

3.2.3 Local Walk

The second component of the proposed algorithm is the abandon operator which generates solutions far from the best solution 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).

The purpose of the abandon operator is to discover foreign eggs from those of the host bird within a probability *Pa*. If foreign eggs are found, the host bird will abandon them and replace them with new eggs. In FCSA, this process is applied to the key vector, where foreign solutions are replaced with new solutions generated using equation 9.

$$x^i_{key}(t+1) = x^i_{key}(t) + S * K \quad (9)$$

With

$$K = \begin{cases} 1 & if \ \ rand > Pa \\ 0 & Otherwise \end{cases} \quad (10)$$

And

$$S = (x^{randperm(n)}_{key}(t) - x^{randperm(n)}_{key}(t)) \quad (11)$$

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 comparing the old and new value of the key vector $|x^i_{key}(t+1) - x^i_{key}(t+1)|$. If the difference exceeds the acceptance rate, a specific node in the locus vector undergoes re-initialization through a roulette selection process. The probability of halting the roulette for this modification is denoted as p and is determined by the formula:

$$p = \frac{w_{ij}}{\sum_{k \in N(i)} w_{ik}}$$

Here, w_{ij} represents the weight of the edge E_{ij} between nodes i and j and $N(i)$ denotes the set of neighbor nodes connected to node i. If the calculated probability p exceeds a random number, a replacement is made, and the current node is substituted with node i. Otherwise, the node's value remains the same as in the previous step. If the newly generated solution has greater modularity than the current solution, it replaces the current solution for this nest.

4 EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we present the results of a series of computational experiments conducted to assess the effectiveness of our three proposed flattening-based CSA algorithms. We denote these proposed algorithms as FCSA1, FCSA2, and FCSA3, each of which combines our CS algorithm with one of the flattening functions, namely, $f1$ (Eq. 2), $f2$ (Eq. 3), and $f3$ (Eq. 4), respectively. We implemented our algorithms in

Phyton and ran experiments on a personal computer running Windows 10 Enterprise, equipped with 8 GB of RAM and powered by an Intel(R) Core(TM) i7-3537u CPU.

All experiments are conducted on both synthetic and real-world networks. We generated synthetic networks using the mLFR benchmark (Brodka and Grecki, 2014), which extends the LFR benchmark (Lancichinetti et al., 2008), with varied structures and layers by adjusting both the mixing parameter (μ) and the degree change chance (Dc). Increasing the values of μ and Dc made the community detection task more challenging by blurring the boundaries between communities and increasing the differences in node degrees across network layers (Table 1). For real-world networks, we employed a diverse set of 6 real-world networks from various domains with diverse sizes and characteristics (Table 2).

Table 1: The m-LFR128 Parameter Settings.

Parameter	Value
Number of nodes	128
Node average degree	8
Number of layers	[2, 3, 4]
Mixing parameter	[0.1, 0.2, 0.3, 0.4, 0.5]
Degree change chance	[0.2, 0.4, 0.6, 0.8]
Maximal community size	32
Minimal community size	8
Node maximal degree	16

Table 2: Real-world networks.

Network	Nodes	Edges	Layers
florentine	16	35	2
Tailorshop	39	552	4
London Transport	369	441	3
CKM Physicians Innovation	246	1551	3
EU-Air Transportation	450	3588	37
FAO Trade	183	16048	339
Plasmodium GPI	1203	2521	3
Celegans GPI	3879	8181	6

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. For real-world networks, where the ground truth community structure is not available, we evaluated the quality of the detected communities using Q_m (Eq. 12) the average modularity function (Eq. 5) across all layers.

$$Q_m = \frac{\sum_{i=1}^{d} Q_i}{d} \tag{12}$$

Where d is the number of layers and Q_i is the modularity of layer i.

Furthermore, we conducted a sensitive analysis of the FCSA algorithm to assess the impact of varying its five parameters. Due to space constraints, we provide a summary of these parameters, Population size n takes a value of 150, Maximum number of iterations $maxGen$ equal to 1000, Abandonment probability Pa is 0.5, Step size scale α takes a value of 0.1, Acceptance rate δ is 0.1.

The proposed algorithms are compared against the Louvain algorithm, which will be combined with three flattening techniques as proposed in (Berlingerio et al., 2011), denoted Louvain1, Louvain2, and Louvain3. Additionally, we will evaluate our algorithms against five other community detection algorithms in multiplex networks: Cinfomap (Rosvall and Bergstrom, 2008), CBGLL (Lancichinetti and Fortunato, 2012), GenLouvin (Mucha et al., 2010), CLPAm (Barber and Clark, 2009), and MMCD (Ma et al., 2018)). However, due to the unavailability of the source code for these algorithms, the comparison will only take place in real-world networks where result values are shared.

In what follows, we will first present the performance analysis of our newly proposed flattening-based CSAs on the synthetic multiplex networks, and then on the real ones.

4.1 FCSA Performance Analysis on Synthetic Networks

Figure 4 shows the results of our experiments on the mLFR benchmarks. It serves as a visual representation of the performance of various algorithms, including FCSA1, FCSA2 and FCSA3, compared with Louvain1, Louvain2 and Louvain3. These algorithms are evaluated in terms of their effectiveness in identifying real communities, considering different values of the mixing parameter (μ) and the degree change hazard (Dc). Each column in this figure is devoted to networks with a different number of layers, while the rows represent networks with different degree change chance (Dc). The graphs in the figure illustrate the results in terms of NMI versus mixing parameter (μ).

When we consider a μ of 0.1, most of the algorithms successfully identify the real communities. However, as we increase μ to values beyond 0.1, the algorithms encounter challenges in accurately detect-

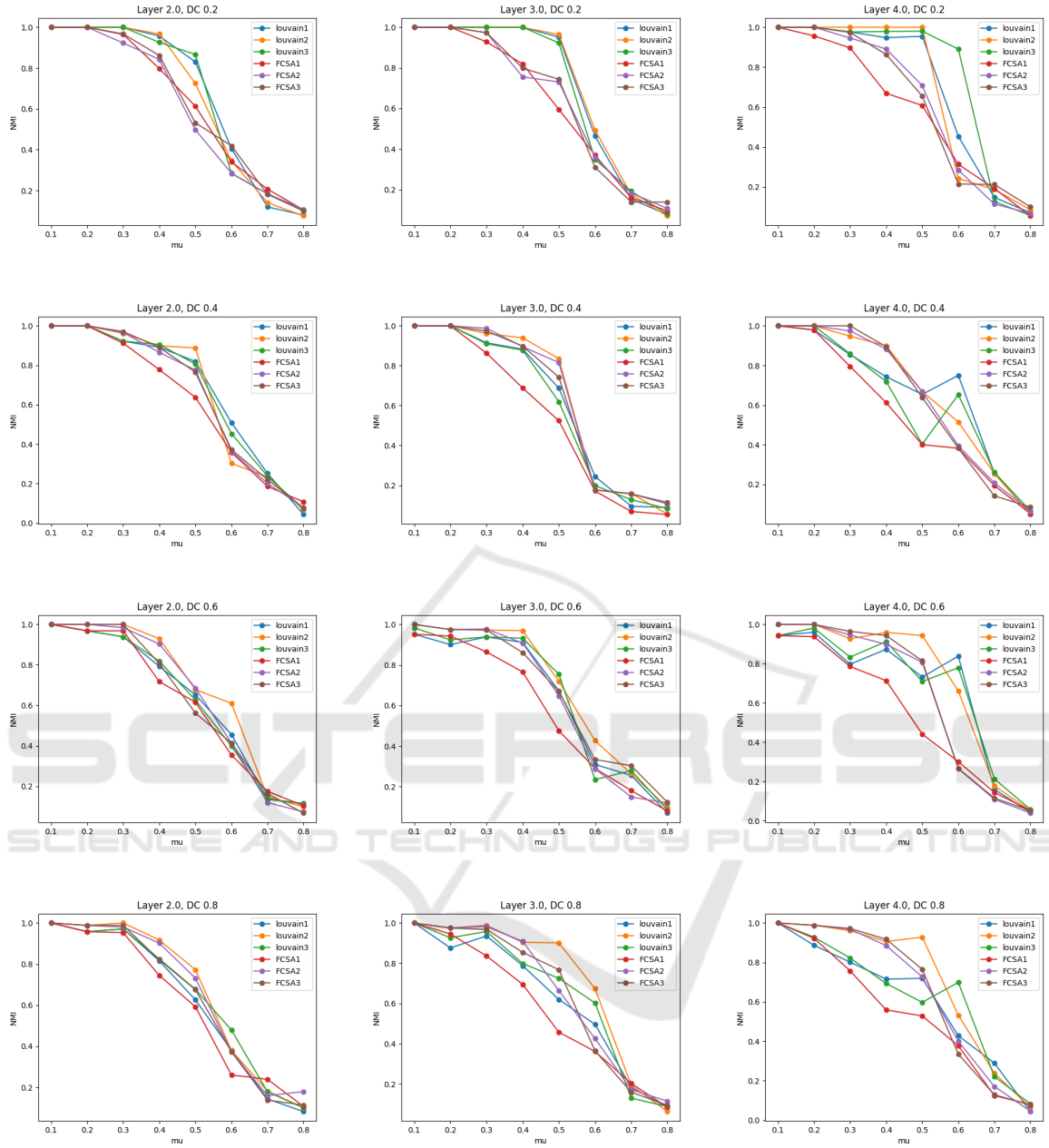


Figure 4: The NMI results for mLFR-128 networks with the increase of network layers (i.e., L = 2, 3, 4) and the change of network structure (mixing parameter *mu* and degree change chance *Dc*).

ing the true communities. Notably, for μ values between 0.3 and 0.6, the Louvain algorithms consistently outperform the other methods in community detection. In more complex networks with higher values of the mixing parameter, we observe that FCSA algorithms deliver superior results. A similar trend can be observed regarding the degree change chance (*Dc*). At a *Dc* value of 0.2, the algorithms perform at their best, after which their performance starts to decline as *Dc* increases. Furthermore, the number of layers in the network significantly influences community detection, particularly for μ values between 0.3 and 0.6. It's evident that the task is more manageable for networks with fewer layers, and the complexity increases when we move from two to three or four layers.

In summary, comparing various flattening techniques proves to be challenging due to performance variations across diverse network structures and parameter settings. It's worth noting that FCSA consis-

Table 3: Comparison of modularity results on real-world networks.

Network	Florentine	Tailorshop	London Transport	CKM	EU-Air	FAO Trade	Plasmo-dium	Celegans
FCSA1	0.2781	0.2370	0.7259	0.6973	0.0107	0.0301	0.1563	0.4498
FCSA2	**0.3250**	**0.2377**	0.7239	**0.7030**	**0.0131**	**0.1740**	**0.1601**	**0.5320**
FCSA3	0.2725	0.2332	**0.7325**	0.6960	0.0110	0.0215	0.1559	0.4324
Louvain1	0.2780	0.2335	**0.7227**	**0.6938**	-0.0399	0.0555	0.1765	0.5480
Louvain2	**0.3250**	**0.2374**	0.6529	0.6918	**-0.0173**	**0.1938**	**0.1766**	**0.5480**
Louvain3	0.2385	0.1696	0.6224	0.6715	-0.0230	0.0449	0	0.4250
Cinfomap	-	-	0.7456	0.6579	0.0120	0.3113	0.4601	0.4905
CBGLL	-	-	0.7352	0.6833	0.0479	0.2378	0.4734	0.4732
Gen-Louvain	0.2936	**0.2238**	0.7892	0.6944	0.1154	**0.3174**	0.5183	0.5445
CLPAm	-	-	0.5352	0.5702	0.1033	0.3124	0.3406	0.3998
MMCD	**0.3060**	0.2107	**0.7932**	**0.7056**	**0.1219**	**0.3174**	**0.5237**	**0.5524**

tently delivers optimal results for complex networks characterized by high values of *mu* and *DC*.

4.2 FCSA Performance Analysis on Real Networks

Table 3 presents the experimental modularity results for real-world networks, with values highlighted in bold indicating the highest modularity values in their respective columns based on problem resolution type. As the modularity values for real-world networks are gathered from the literature, the use of dashes indicates that the specific result values for the corresponding algorithm and dataset were not available. Our analysis reveals noteworthy trends, particularly the superior performance of the second flattening function, outperforming other techniques in 6 out of 8 real-world networks when both FCSA and Louvain are employed. Compared to Louvain, FCSA demonstrates superiority in 4 out of 7 real-world networks and achieves comparable results for the Florentine network.

In a comparison between FCSA and direct methods, FCSA exhibits superiority in multiple instances, surpassing CLPAm and Cinfomap in 3 out of 6 cases, CBGLL in 2 out of 6 cases, GenLouvain in 3 out of 8 cases, and MMCD in 2 out of 8 cases. Upon analysis, it becomes apparent that a significant amount of information is lost when flattening multiplex networks with a high number of layers and high density. This information loss could explain why direct methods outperform our proposed algorithm. However, for small networks with a limited number of layers, such as Florentine and Tailorshop, our approach yields better results due to the minimal loss of information during flattening.

5 CONCLUSIONS

In this paper, we introduced FCSA, a novel community detection algorithm utilizing three distinct strategies to flatten multiplex networks. The incorporation of locus and random key representations in FCSA enhances network encoding and search capabilities. Experimental results on synthetic networks demonstrate the algorithm's superior performance for complex networks. However, when applied to real-world networks, its effectiveness is more pronounced for small networks. It is crucial to note that the transformation of the multiplex network into a monodimensional network introduces some information loss, which stands as a current limitation of our algorithm. Our future objective is to extend the algorithm to operate directly on the multiplex network, aiming to overcome this specific limitation.

REFERENCES

Barber, M. J. and Clark, J. W. (2009). Detecting network communities by propagating labels under constraints. *Physical Review E*, 80(2):026129.

Berlingerio, M., Coscia, M., and Giannotti, F. (2011). Finding and characterizing communities in multidimensional networks. In *2011 international conference on advances in social networks analysis and mining*, pages 490–494. IEEE.

Boccaletti, S., Bianconi, G., Criado, R., Del Genio, C. I., Gómez-Gardenes, J., Romance, M., Sendina-Nadal, I., Wang, Z., and Zanin, M. (2014). The structure and dynamics of multilayer networks. *Physics reports*, 544(1):1–122.

Boukabene., R., Tayeb., F., and Dakiche., N. (2023). Improved random key cuckoo search optimization algorithm for community detection in social networks. In

Proceedings of the 19th International Conference on Web Information Systems and Technologies - WEBIST, pages 113–120. INSTICC, SciTePress.

Brodka, P. and Grecki, T. (2014). mlfr benchamark: Testing community detection algorithms in multi-layered. *Multiplex and Multiple Social Networks*.

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

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.

Gong, M., Cai, Q., Chen, X., and Ma, L. (2013). Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition. *IEEE Transactions on evolutionary computation*, 18(1):82–97.

Harakawa, R., Takimura, S., Ogawa, T., Haseyama, M., and Iwahashi, M. (2019). Consensus clustering of tweet networks via semantic and sentiment similarity estimation. *IEEE Access*, 7:116207–116217.

Huang, X., Chen, D., Ren, T., and Wang, D. (2021). A survey of community detection methods in multilayer networks. *Data Mining and Knowledge Discovery*, 35:1–45.

Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y., and Porter, M. A. (2014). Multilayer networks. *Journal of complex networks*, 2(3):203–271.

Lancichinetti, A. and Fortunato, S. (2012). Consensus clustering in complex networks. *Scientific reports*, 2(1):336.

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

Loe, C. W. and Jensen, H. J. (2015). Comparison of communities detection algorithms for multiplex. *Physica A: Statistical Mechanics and its Applications*, 431:29–45.

Ma, L., Gong, M., Yan, J., Liu, W., and Wang, S. (2018). Detecting composite communities in multiplex networks: A multilevel memetic algorithm. *Swarm and evolutionary computation*, 39:177–191.

Magnani, M., Hanteer, O., Interdonato, R., Rossi, L., and Tagarelli, A. (2021). Community detection in multiplex networks. *ACM Computing Surveys (CSUR)*, 54(3):1–35.

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

Mucha, P. J., Richardson, T., Macon, K., Porter, M. A., and Onnela, J.-P. (2010). Community structure in time-dependent, multiscale, and multiplex networks. *science*, 328(5980):876–878.

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

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.

Roozbahani, Z., Emamgholizadeh, H., Rezaeenour, J., and Hajialikhani, M. (2021). A systematic survey on multi-relational community detection. *arXiv preprint arXiv:2103.15698*.

Rosvall, M. and Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the national academy of sciences*, 105(4):1118–1123.

Shen, H.-W. and Cheng, X.-Q. (2010). Spectral methods for the detection of network community structure: a comparative analysis. *Journal of Statistical Mechanics: Theory and Experiment*, 2010(10):P10020.

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.