An Efficient Cooperative Scatter Search for the Large-Scale k-Clustering Minimum Biclique Completion Problem

Mhand Hifi[1,⋆], Yasmine Salmi[2] and Juntao Zhao[3]

***Abstract*— The k-Clustering Minimum Biclique Completion Problem (k-CMBCP) is an NP-hard combinatorial optimization problem with significant implications in bipartite graph clustering applications. The objective is to partition a set of services into k disjoint clusters such that the number of missing edges required to transform each cluster into a complete biclique is minimized. This paper proposes a robust Cooperative Scatter Search (CSS) algorithm designed to exploit the structural characteristics of the problem. The proposed metaheuristic framework integrates a diversification-based population initialization strategy, dynamic management of elite reference set, and a specialized uniform-based recombination operator. Furthermore, the algorithm incorporates a sequential Variable Neighborhood Descent (VND) strategy leveraging three complementary neighborhoods to intensify the local search. The contribution is a problem-tailored integration of known search ingredients rather than a generic new metaheuristic paradigm: its novelty lies in the way diversification, reference-set memory, recombination, and SeqVND are coordinated for the specific structure of the k-CMBCP. Computational experiments on benchmark instances show that the proposed method achieves competitive and superior performance compared with recent heuristics in terms of solution quality and computational stability.**

I. INTRODUCTION

The k-Clustering Minimum Biclique Completion Problem (k-CMBCP) appears in several practical applications, such as multicast communication systems [1], logistics network management [2], and market segmentation [3] in bipartite networks. In these applications, services interact with multiple clients, and overlapping connection patterns must be restructured to reduce structural redundancy. Given a bipartite graph $G = (S, C, E)$, where S denotes the set of services and C denotes the set of clients, the problem aims to partition services into k disjoint clusters such that each cluster forms a complete bipartite subgraph upon the addition of the minimum number of missing edges.

Formally, the objective is to minimize the aggregate number of added edges needed to transform each induced bipartite subgraph into a complete biclique.

The problem is NP-hard and becomes computationally challenging for medium and large-scale instances. Although several heuristic and matheuristic methods have been proposed, achieving a balance between solution quality and scalability remains an open challenge. Existing exact and matheuristic methods obtain strong bounds on small and medium instances, whereas recent population-based and learning-guided heuristics show better scalability. However, gaps still remain in methods combining population diversification, structured recombination, and local search intensification within a unified framework while maintaining feasible service partitions.

In this paper, an efficient Cooperative Scatter Search (CSS) framework is proposed, integrating the following components:

- A *diversified population construction mechanism* that generates structurally diverse initial solutions to strengthen global exploration.
- An *elite reference set management strategy* that maintains high-quality and diverse candidate solutions to balance intensification and diversification.
- A *uniform-based recombination operator* that exchanges cluster assignments between parent solutions to generate promising offspring solutions.
- A *Sequential Variable Neighborhood Descent procedure* (SeqVND) that explores multiple neighborhood structures in a coordinated way to improve local optimality.

Compared with existing k-CMBCP heuristics, the proposed method can be viewed as an effective integration of complementary components: diversification-based construction improves initial exploration, the reference set maintains elite and diverse solutions, the uniform recombination operator exchanges service assignments without destroying the partition structure, and SeqVND improves solution quality through systematic local refinement.

The remainder of the paper is organized as follows. Section II introduces the problem formulation. Section III reviews relevant literature. Section IV discusses the proposed CSS algorithm. Section V reports the computational results. Section VI concludes the paper.

II. PROBLEM DESCRIPTION

The problem can be modeled as an undirected bipartite graph $G = (S, C, E)$, where $S = \{s_1, s_2, \ldots, s_m\}$ denotes the set of services, $C = \{c_1, c_2, \ldots, c_n\}$ denotes the set of clients, and $E \subseteq S \times C$ represents the existing connections between services and clients.

Let $\bar{E} = (S \times C) \setminus E$ be the set of missing edges. The k-CMBCP aims to partition the set of services S into k disjoint clusters such that, for each cluster, the corresponding induced

⋆Corresponding author (M. Hifi). All authors are listed in alphabetical order of their last name.

[1]EPROAD UR 4669, UPJV, Amiens, France. Corresponding author: hifi@u-picardie.fr

[2]LaROMaD, USTHB, BP 32 El Alia, 16111 Alger, Algérie salmi68yasmine@gmail.com

[3]School of Electrical Engineering, Hebei University of Technology, Tianjin, China. juntao.zhao@hebut.edu.cn

bipartite subgraph becomes a complete bipartite graph, i.e., a biclique, after adding the minimum number of missing edges.

A. Mathematical Formulation

Binary decision variables are defined as follows:

$$x_{ip} = \begin{cases} 1 & \text{if service } s_i \text{ is assigned to cluster } p, \\ 0 & \text{otherwise,} \end{cases}$$

$$y_{jp} = \begin{cases} 1 & \text{if client } c_j \text{ is associated with cluster } p, \\ 0 & \text{otherwise.} \end{cases}$$

The quadratic formulation of the problem is given by [1]:

$$\min f(x,y) = \sum_{p=1}^{k} \sum_{(i,j)\in\bar{E}} x_{ip} y_{jp} \tag{1}$$

subject to

$$\sum_{p=1}^{k} x_{ip} = 1, \quad \forall i \in S, \tag{2}$$

$$x_{ip} \le y_{jp}, \quad \forall (i,j) \in E, \ \forall p, \tag{3}$$

$$x_{ip}, y_{jp} \in \{0,1\}. \tag{4}$$

The objective function quantifies the total number of missing edges that must be added to achieve biclique completion for each cluster. For each cluster, the objective counts all missing service–client pairs that need to be inserted once clients adjacent to at least one service are assigned to the cluster. Therefore, the product $x_{ip}y_{jp}$ contributes to the completion cost only when service s_i and client c_j are assigned to the same biclique and the corresponding edge does not exist in the original graph.

Alternatively, the quadratic formulation is linearized by introducing auxiliary binary variables z_{ijp} to represent the product $x_{ip}y_{jp}$, defined as:

$$z_{ijp} = \begin{cases} 1 & \text{if both } x_{ip} = 1 \text{ and } y_{jp} = 1, \\ 0 & \text{otherwise.} \end{cases}$$

The resulting linear formulation [1], [4] is:

$$\min f(z) = \sum_{p=1}^{k} \sum_{(i,j)\in\bar{E}} z_{ijp} \tag{5}$$

subject to the original assignment constraints and the following linking constraints:

$$z_{ijp} \le x_{ip}, \quad \forall (i,j) \in \bar{E}, \ \forall p, \tag{6}$$

$$z_{ijp} \le y_{jp}, \quad \forall (i,j) \in \bar{E}, \ \forall p, \tag{7}$$

$$x_{ip} + y_{jp} \le 1 + z_{ijp}, \quad \forall (i,j) \in \bar{E}, \ \forall p, \tag{8}$$

$$x_{ip}, y_{jp}, z_{ijp} \in \{0,1\}. \tag{9}$$

The auxiliary variables guarantee that z_{ijp} takes a value of 1 if and only if service s_i and client c_j are both assigned to cluster p. This linearization allows the model to be solved by standard exact solvers while preserving the original structure of the problem. This linearized model is the one used for CPLEX comparisons. It yields the same objective value as the quadratic formulation, but replaces each bilinear term with three standard linking inequalities allowing the model to be solved by a mixed-integer linear solver.

B. Solution Representation and Evaluation

Each solution is represented as:

$$x = (x_1, \ldots, x_m),$$

where x_i denotes the cluster assigned to service s_i.

Cluster costs are updated incrementally to accelerate evaluation during neighborhood exploration. Each service is assigned to exactly one of the k clusters.

For cluster p, define:

- S_p: the set of services assigned to cluster p;
- C_p: the union of clients connected to services in S_p.

The cost associated with cluster p is computed as:

$$\text{cost}_p = |S_p|\,|C_p| - \sum_{s_i \in S_p} d(s_i) \tag{10}$$

where $d(s_i)$ denotes the degree of service s_i in the original bipartite graph. Equivalently, this expression evaluates the same completion cost as the mathematical model: $|S_p||C_p|$ represents the number of edges required to form a complete biclique induced by (S_p, C_p), whereas the second term removes the edges already present. The neighborhood search therefore evaluates the original biclique-completion objective, using an efficient cluster-wise incremental evaluation scheme.

The objective function is:

$$\min f(x) = \sum_{p=1}^{k} \text{cost}_p. \tag{11}$$

A solution is feasible when every cluster contains at least one service.

Although the mathematical formulation is compact and intuitive, solving it exactly becomes computationally intractable for medium and large-scale instances because of the combinatorial complexity of the problem. Therefore, efficient heuristic and metaheuristic approaches are therefore needed to obtain high-quality solutions within reasonable computation time.

III. RELATED WORKS

The k-CMBCP was first introduced in [1], where its NP-hardness and both quadratic and linear formulations were established in the context of multicast communication. Exact approaches were later strengthened through constraint-programming and semidefinite relaxations [5], as well as branch-and-price combined with infeasible-search ideas [4]. These methods provide valuable bounds, but their scalability remains limited when instance size and density increase.

Heuristic and matheuristic approaches were proposed to address this limitation. Adaptive neighborhood search [6], iterated variable neighborhood search [7], and fast approximation-oriented frameworks [8] improved the trade-off between solution quality and computational time. Later, rounding-and-improvement schemes [9], [10] exploited mathematical relaxations followed by local refinement, obtaining strong results at the cost of increased algorithmic complexity.

More recent population-based and learning-guided methods, including discrete particle swarm and augmented swarm strategies [11], [12] and reinforcement-learning-guided VNS [13], highlight the importance of adaptive diversification strategies and operator selection. However, maintaining feasible clusters while combining diverse high-quality partitions remains challenging. CSS follows this line of research, it does not aim to outperform all metaheuristic frameworks conceptually, but demonstrates that a coordinated scatter-search framework can exploit the specific cost structure of the k-CMBCP effectively on the benchmark instances considered.

IV. PROPOSED COOPERATIVE SCATTER SEARCH

A. Overall Framework

The proposed CSS framework combines diversification-based population construction, elite reference set management, structured recombination mechanism, and sequential local search improvement. This framework exploits the properties of bipartite graphs by balancing global exploration and intensified exploitation. The improvement mainly comes from the interaction among these components, including the diversified population generates structurally different service partitions, the reference set maintains high-quality and diverse solutions, the recombination operator exchanges promising cluster assignments, and SeqVND further refines the recombined solutions into locally optimized partitions.

Algorithm 1 Cooperative Scatter Search Framework

```
Generate diversified initial population P
Construct reference set R from P
while termination criterion not met do
    Select elite pairs from R
    Generate offspring via uniform recombination
    Improve offspring using SeqVND
    Update reference set R based on quality and diversity
end while
return best solution found
```

In each iteration, elite solutions are combined to generate offspring solutions, which are subsequently refined using a Sequential Variable Neighborhood Descent (SeqVND) procedure. Improved solutions can replace lower-quality members in the reference set, thereby maintaining convergence toward high-quality regions while preserving diversity.

B. Diversified Population Generation

The initial population is generated through randomized greedy assignment procedures to increase structural diversity among solutions. Each service is assigned to one of the k clusters while ensuring that no cluster remains empty.

Controlled randomization is applied during cluster selection to enhance exploration and avoid premature convergence toward structurally similar configurations. The resulting population establishes a diverse foundation for the subsequent reference set construction.

Algorithm 2 Diversified Population Generation

```
for i = 1 to population size do
    Initialize empty clusters
    Randomly assign at least one service to each cluster to ensure feasibility
    for remaining services do
        Assign service to a randomly selected cluster
    end for
    Add solution to population P
end for
return P
```

C. Reference Set Construction

The reference set R contains high-quality and diverse solutions selected from the population. A subset of solutions is selected based on their objective values, whereas additional solutions are added to maximize diversity among solutions, measured by the Hamming distance between service-to-cluster assignment vectors.

Algorithm 3 Reference Set Construction

```
Sort population P by objective value
Select top r_1 solutions into R
while |R| < r do
    Select solution s ∈ P \ R maximizing the minimum distance to all solutions currently in R
    Add selected solution to R
end while
return R
```

D. Uniform-Based Recombination

Given two elite parent solutions from the reference set, an offspring solution is generated by independently inheriting the cluster assignment of each service from one of the parents with equal probability. This recombination mechanism allows for the preservation of high-quality assignment patterns while generating new structural combinations.

A repair procedure is subsequently applied to ensure that no cluster becomes empty after recombination, thereby maintaining the requirement of exactly k clusters.

Algorithm 4 Uniform-Based Recombination

```
for each service s_i do
    With probability 0.5, inherit assignment from parent 1
    Otherwise inherit from parent 2
end for
if any cluster is empty then
    Reassign randomly selected services to empty clusters to restore feasibility
end if
return offspring solution
```

E. Sequential Variable Neighborhood Descent

To intensify the search, three neighborhood structures are explored in sequence:

- **Shift (1–0):** relocates a single service to a different cluster;
- **Swap (1–1):** exchanges two services between different clusters;
- **Shift (2–0):** transfers two services simultaneously to a target cluster.

These operators are illustrated using the instance in Fig. 1(a), where the service set is $S = \{1,2,3,4\}$ and the client set is $C = \{5,6,7,8,9\}$. Edges correspond to existing service–client connections in the bipartite graph.

The initial solution divides the services into two clusters: $S_1 = \{1,2,3\}$ and $S_2 = \{4\}$. The clients associated with each cluster are determined by the union of neighboring clients of the assigned services. The corresponding objective value measures the missing edges required to transform each induced subgraph into a biclique. In this configuration, S_1 has a cost of 9 while S_2 already forms a complete biclique, resulting in a total cost of 9.

Fig. 1(b)–(d) shows the effect of the three local operators:

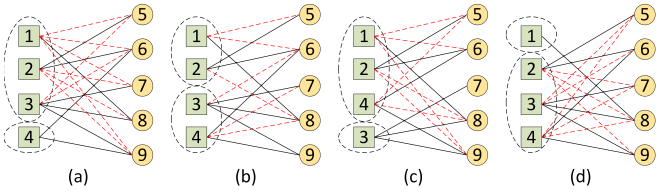

Fig. 1. Illustration of the three local operators on the k-CMBCP instance.

- *Shift (1–0)*: Moving service 3 from S_1 to S_2 produces clusters $\{1,2\}$ and $\{3,4\}$ (Fig. 1(b)). This redistribution improves structural density, reducing the objective value to 6.
- *Swap (1–1)*: Exchanging services 3 and 4 yields $S_1 = \{1,2,4\}$ and $S_2 = \{3\}$ (Fig. 1(c)), leading to an objective value of 7. Although the improvement is limited, the move changes cluster composition while preserving cardinalities.
- *Shift (2–0)*: Relocating services 2 and 3 to S_2 results in $S_1 = \{1\}$ and $S_2 = \{2,3,4\}$ (Fig. 1(d)), producing an objective value of 8. This operator introduces larger structural perturbations to escape local minima.

Whenever an improving move is identified, the search restarts from the first neighborhood. Computational efficiency is achieved through incremental cost evaluation during move assessment. The example in Fig. 1 is provided only to illustrate the operators. In the implementation, each move is evaluated by updating only the affected clusters instead of recomputing the entire objective value, which contributes to the scalability observed on large instances.

Algorithm 5 Sequential Variable Neighborhood Descent

```
k ← 1
while k ≤ 3 do
    Explore neighborhood k
    if improving move found then
        Apply move and update objective value
        k ← 1
    else
        k ← k + 1
    end if
end while
return improved solution
```

F. Reference Set Update

Following local improvement, the offspring solution is compared with the current members of the reference set. The solution is added to the set if it improves solution quality or increases diversity, potentially replacing the worst or most similar solution to maintain a high-quality and diverse population.

Algorithm 6 Reference Set Update

```
if offspring dominates worst in R then
    Replace worst solution
else if offspring contributes to diversity then
    Replace most similar solution
end if
return updated R
```

V. COMPUTATIONAL RESULTS

A. Benchmark Instances and Experimental Setting

The proposed CSS algorithm is evaluated on two publicly available benchmark sets [1] introduced in [10]. Specifically, `Set II` contains 18 medium- and large-scale instances with $|S| = |C| \in \{50,80,100\}$ and $k \in \{5,10\}$, while `Set III` consists of 36 large-scale instances with $|S| = |C| \in \{120,150,200,300\}$ and $k \in \{5,10,15\}$. Both sets incorporate bipartite densities $d \in \{0.3,0.5,0.7\}$ to cover sparse to dense graph structures. Small instances from `Set I` are excluded as recent state-of-the-art approaches solve them to optimality within negligible computation time [10], [12], [13].

The CSS algorithm is implemented in `C++` and compiled using the `-O3` optimization flag. Computational experiments were conducted on an Intel Core i7-4790 processor (3.6 GHz) with 32 GB RAM. Each instance is executed 10 times with independent random seeds to account for stochastic variability, and the best, average, and standard deviation of the objective values are reported. The method is compared against CPLEX (v22.1.1.0, RINS=100, one thread, 60 min limit), RSBA [10], ASOA [12], and RLE-VNS [13]. To ensure fairness, all metaheuristics are limited to 5 minutes for `Set II` and 30 minutes for `Set III`.

[1] `https://www.u-picardie.fr/eproad/?page_id=485`

To obtain exact baselines, the linear formulation of the k-CMBCP was solved using CPLEX (version 22.1.1.0) in a single-thread setting on the MatriCS platform [2]. The RINS heuristic frequency was set to 100, and a maximum time limit of 60 minutes was imposed for each instance. Since RSBA, ASOA, and RLE-VNS are taken from previously published results rather than being reimplemented on the same platform, the comparison mainly focuses on solution quality; runtime comparisons are limited to the reported hardware and time settings. For the stochastic runs of CSS, the average and standard deviation reported in the experiments are used to evaluate robustness, so the analysis is not based solely on best-case results.

B. Parameter Tuning

The CSS framework involves two main parameters: the reference set size nb_R and population size $nb_P = 2 \times nb_R$. Configurations for $nb_R \in \{5, 10, 15, 20, 25\}$ were evaluated over 20 independent runs on 36 tuning instances. As reported in Table I, the configuration $(nb_R, nb_P) = (15, 30)$ achieves the evaluated average objective value (7332.92) and the highest number of best solutions (18/36), statistically supported by the Wilcoxon signed-rank test ($p = 1.96 \times 10^{-4}$). Smaller configurations ($nb_R \leq 10$) result in insufficient population diversity, while larger ones ($nb_R \geq 20$) provide limited additional improvement because of the additional computational overhead.

TABLE I

SUMMARY OF PARAMETER TUNING RESULTS ON 36 INSTANCES.

$nb_{\mathscr{R}}$	5	10	**15**	20	25
$nb_{\mathscr{P}}$	10	20	**30**	40	50
f_{avg}	7380.50	7377.63	**7372.39**	7370.74	7366.99
$f_{\text{best\_avg}}$	7337.08	7341.58	**7332.92**	7339.14	7336.39
Nb_best	10	9	**18**	14	16
p-value	8.24E-06	5.59E-06	1.96E-04	3.99E-05	8.82E-05

C. Ablation Study: Effect of Neighborhood Structures

To assess the contribution of each neighborhood operator, three ablation variants were tested: $\text{CSS}_{\texttt{NoN1}}$ (without shift 1-0), $\text{CSS}_{\texttt{NoN2}}$ (without swap 1-1), and $\text{CSS}_{\texttt{NoN3}}$ (without shift 2-0). As reported in Table II, the full CSS variant obtains the lowest average objective value (7331.33) and the largest number of best solutions (21/36). Removing $\mathscr{N}_1$ causes the largest performance degradation, highlighting its critical role. The Wilcoxon signed-rank tests show that $\text{CSS}_{\texttt{Full}}$ performs significantly better than all variants, supporting the combined use of all three neighborhoods.

TABLE II

ABLATION STUDY: PERFORMANCE OF CSS WITH AND WITHOUT EACH NEIGHBORHOOD.

Variant	$\text{CSS}_{\texttt{NoN1}}$	$\text{CSS}_{\texttt{NoN2}}$	$\text{CSS}_{\texttt{NoN3}}$	$\text{CSS}_{\texttt{Full}}$
$f_{\text{best\_avg}}$	7361.97	7337.75	7354.78	**7331.33**
f_{avg}	7391.73	7374.52	7391.17	**7371.18**
Nb_best	8	14	10	**21**

[2] https://www.matrics.u-picardie.fr/

D. Comparison with State-of-the-Art Methods

1) Results on Set II: Table III reports the best objective values obtained by the competing methods on `Set II`. Overall, CSS performs better than the other tested methods in terms of solution quality and robustness, obtaining the best-known solutions for 16 out of 18 instances and establishing 9 new upper bounds (†). The CPLEX results remain below the metaheuristic best values on this set, with no best-known solution reached within the 60-minute time limit. This underscores the significant combinatorial challenge posed by the k-CMBCP for exact solvers.

TABLE III

RESULTS ON SET II: BEST OBJECTIVE VALUES.

#Inst.	Best	CPLEX	RSBA	ASOA	RLE-VNS	CSS
50-50-5-0.3	1309	1354	1315	1313	1312	**1309**†
50-50-5-0.5	1060	1085	1068	1062	1061	**1060**†
50-50-5-0.7	671	672	671	671	671	671
50-50-10-0.3	930	1112	938	934	932	**930**†
50-50-10-0.5	871	944	871	871	871	871
50-50-10-0.7	574	584	575	574	574	574
80-80-5-0.3	3800	3977	3805	3805	3802	**3800**†
80-80-5-0.5	2860	2925	2862	2861	2860	2860
80-80-5-0.7	1768	1773	1768	1768	1768	1768
80-80-10-0.3	3160	3599	3188	3178	3175	**3160**†
80-80-10-0.5	2551	2673	2570	2568	2567	**2551**†
80-80-10-0.7	1614	1659	1617	1616	1615	**1614**†
100-100-5-0.3	6243	6406	6247	6247	6247	**6243**†
100-100-5-0.5	4650	4722	4650	4650	4650	4655
100-100-5-0.7	2832	2863	2832	2832	2832	2832
100-100-10-0.3	5407	5948	5426	5419	5419	**5407**†
100-100-10-0.5	4274	9067	4298	4274	4274	4293
100-100-10-0.7	2644	2660	2644	2644	2644	2644
Avg		3001.28	2630.28	2627.06	2626.33	**2624.56**
Nb_best		0	6	8	9	**16**

†: newly discovered upper bound; underline: matches best known solution.

For $|S| = |C| = 100$, CSS remains effective when $k = 10$, suggesting that recombination and SeqVND can better handle complex cluster interactions. Its average value (2624.56) is the best among the compared heuristics, outperforming RSBA (2630.28), ASOA (2627.06), and RLE-VNS (2626.33). The CPLEX average is 3001.28, indicating that the exact model mainly serves as a reference baseline but scales poorly under the imposed 60-minute limit.

2) Results on Set III: Table IV reports the results on the more challenging `Set III`, which includes larger and more complex instances. Overall, CSS outperforms the competing heuristics, obtaining the best-known solution for 30 out of 36 instances and establishing 26 newly improved upper bounds (†). In contrast, RSBA, ASOA, and RLE-VNS obtain only 5, 4, and 5 best results, respectively, while CPLEX reaches the best value on one instance but remains less competitive on average under the 60-minute limit. These results confirm that exact approaches and traditional swarm methods become less effective as problem size and complexity increase.

On `Set III`, CSS achieves the best average objective value (23162.58), outperforming RLE-VNS (23305.53), ASOA (23559.39), and RSBA (23569.17). CPLEX improves the result for one 120-service instance but still yield an average of 24387.89, highlighting the scalability limitations of exact optimization on large instances. The advantages of CSS are most apparent for $|S| = |C| = 300$ and larger k, where reference-set diversity, recombination, and SeqVND jointly help avoid premature convergence while maintaining effective intensification.

TABLE IV

RESULTS ON SET III: BEST OBJECTIVE VALUES.

#Inst.	Best	CPLEX	RSBA	ASOA	RLE-VNS	CSS
120-120-5-0.3	8192	8381	8285	8280	8276	**8192**†
120-120-5-0.5	6066	6107	6071	6071	6071	**6066**†
120-120-5-0.7	4307	**4307**†	4394	4394	4391	4337
120-120-10-0.3	7590	7878	7670	7663	7659	**7590**†
120-120-10-0.5	5764	5835	5862	5841	5837	**5764**†
120-120-10-0.7	4094	4139	4132	4126	4122	**4094**†
120-120-15-0.3	6958	7483	7065	7054	7048	**6958**†
120-120-15-0.5	5406	5646	5478	5471	5468	**5406**†
120-120-15-0.7	3873	3942	3968	3943	3926	**3873**†
150-150-5-0.3	14304	14491	14385	14385	14384	**14304**†
150-150-5-0.5	1960	1970	2004	1989	1970	**1960**†
150-150-5-0.7	8584	8610	8584	8584	8584	8649
150-150-10-0.3	13359	13941	13398	13398	13398	**13359**†
150-150-10-0.5	1510	1568	1576	1569	1548	**1510**†
150-150-10-0.7	8232	8387	8254	8256	8256	**8232**†
150-150-15-0.3	12346	14651	12554	12533	12522	**12346**†
150-150-15-0.5	1077	1205	1772	1749	1451	**1077**†
150-150-15-0.7	7859	8099	7954	7954	7954	**7859**†
200-200-5-0.3	28331	28927	28476	28476	28466	**28331**†
200-200-5-0.5	19430	19579	19430	19430	19430	19627
200-200-5-0.7	19422	19522	19422	19422	19422	19563
200-200-10-0.3	26561	28995	26949	26951	26931	**26561**†
200-200-10-0.5	18818	19877	18914	18914	18907	**18818**†
200-200-10-0.7	18799	19197	18814	18814	18814	**18799**†
200-200-15-0.3	24720	28822	25505	25500	25304	**24720**†
200-200-15-0.5	18212	19931	18364	18366	18366	**18212**†
200-200-15-0.7	18193	19843	18298	18298	18298	**18193**†
300-300-5-0.3	69068	70479	70444	70421	70273	**69068**†
300-300-5-0.5	55598	56123	55598	55598	55598	56694
300-300-5-0.7	53725	54463	53725	53728	53725	56938
300-300-10-0.3	65355	71145	66684	66545	66528	**65355**†
300-300-10-0.5	54171	56501	54246	54246	54246	**54171**†
300-300-10-0.7	52392	54334	53913	53913	53389	**52392**†
300-300-15-0.3	61002	72354	65956	65945	62504	**61002**†
300-300-15-0.5	52876	56694	55322	55300	53897	**52876**†
300-300-15-0.7	50957	54538	55024	55011	52036	**50957**†
Avg		24387.89	23569.17	23559.39	23305.53	**23162.58**
Nb_best		1	5	4	5	**30**

†: newly discovered upper bound; underline: matches best known solution.

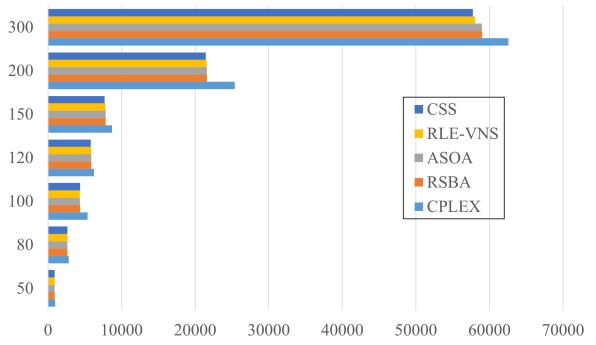

Fig. 2. Average objective values of different algorithms across varying instance sizes.

Figure 2 illustrates the variation in average objective values with instance size. As the problem scale increases from 50 to 300, all algorithms show an increase in objective value; however, CSS consistently achieves the lowest values across all size categories. The performance gap becomes more evident for large instances (200 and 300), where competing methods, including the CPLEX run, show significant performance deterioration. This trend confirms the scalability and robustness of CSS when handling large and complex k-CMBCP instances.

VI. CONCLUSION

This paper presented an efficient cooperative scatter search (CSS) framework tailored for the large-scale k-CMBCP. The approach integrates diversification-based population generation, elite reference set management, uniform recombination, and sequential variable neighborhood descent. This cooperative framework supports global exploration, while the structured local search provides effective intensification in promising regions of the search space.

Computational experiments show that CSS is highly competitive with and frequently outperforms state-of-the-art methods and improves several best-known upper bounds, especially on large-scale instances. The results validate that the combined search strategy is effective for handling the increasing complexity of large bipartite instances. The results suggest that a tailored combination of established metaheuristic components performs effectively on the k-CMBCP, rather than as a claim of universal dominance for all graph clustering problems.

Future research includes extending CSS to dynamic or stochastic variants, incorporating adaptive parameter control mechanisms, and developing hybrid exact–heuristic strategies to improve solution quality for very large instances.

REFERENCES

[1] N. Faure, P. Chrétienne, E. Gourdin, and F. Sourd, "Biclique completion problems for multicast network design," *Discrete Optimization*, vol. 4, no. 3-4, pp. 360–377, 2007.

[2] A. Aboelfotoh, M. Singh, and G. Suer, "Order batching optimization for warehouses with cluster-picking," *Procedia Manufacturing*, vol. 39, pp. 1464–1473, 2019.

[3] M. Subramaniyan, A. Skoogh, A. S. Muhammad, J. Bokrantz, B. Johansson, and C. Roser, "A generic hierarchical clustering approach for detecting bottlenecks in manufacturing," *Journal of Manufacturing Systems*, vol. 55, pp. 143–158, 2020.

[4] S. Gualandi, F. Maffioli, and C. Magni, "A branch-and-price approach to k-clustering minimum biclique completion problem," *International Transactions in Operational Research*, vol. 20, no. 1, pp. 101–117, 2013.

[5] S. Gualandi, "k-clustering minimum biclique completion via a hybrid cp and sdp approach," in *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems: 6th International Conference, CPAIOR 2009 Pittsburgh, PA, USA, May 27-31, 2009 Proceedings 6*. Springer, 2009, pp. 87–101.

[6] M. Hifi, I. Moussa, T. Saadi, and S. Saleh, "An adaptive neighborhood search for k-clustering minimum bi-clique completion problems," in *Modelling, Computation and Optimization in Information Systems and Management Sciences: Proceedings of the 3rd International Conference on Modelling, Computation and Optimization in Information Systems and Management Sciences-MCO 2015-Part I*. Springer, 2015, pp. 15–25.

[7] N. Al-Iedani, M. Hifi, and T. Saadi, "Neighborhood search-based heuristic for the k-clustering minimum biclique completion problem," in *2016 International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, 2016, pp. 639–643.

[8] S. A. Saleh and M. Hifi, "A fast method for optimizing the k-clustering bi-clique completion problem in telecommunication," *Journal of Duhok University*, pp. 175–183, 2017.

[9] M. Hifi and S. Sadeghsa, "Effect of local branching on the iterative rounding-based method: The case of k-clustering minimum completion problems," *Cybernetics and Systems*, vol. 53, no. 1, pp. 58–79, 2022.

[10] M. Hifi and S. Sadeghsa, "A rounding strategy-based algorithm for the k-clustering minimum biclique completion problem," *Journal of the Operational Research Society*, vol. 74, no. 1, pp. 258–271, 2023.

[11] G.-M. Cochard, M. Hifi, E. Samod, and L. Yousef, "A population-based algorithm for the k-clustering minimum biclique completion problem," in *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, vol. 1. IEEE, 2022, pp. 1461–1466.

[12] G.-M. Cochard, S. Elmi Samod, M. Hifi, and L. Yousef, "An augmented swarm optimization algorithm for k-clustering minimum biclique completion problems," *Soft Computing*, pp. 1–18, 2024.

[13] J. Zhao and M. Hifi, "Reinforcement learning-enhanced variable neighborhood search strategies for the k-clustering minimum biclique completion problem," *Computers & Operations Research*, vol. 178, p. 107008, 2025.