The Minimum Vertex Sum Sitting Arrangement on the cycle: Problem Definition and Empirical Analysis

Marcos Robles*[0000−0002−8376−6209], Sergio Cavero[0000−0002−5258−5915], and Eduardo G. Pardo[0000−0002−6247−5269]

Universidad Rey Juan Carlos, Madrid, Spain
{marcos.robles,sergio.cavero,eduardo.pardo}@urjc.es

Abstract. This paper addresses the Cyclic Minimum Vertex Sum Sitting Arrangement (CMinVSSA), a graph layout optimization problem where the vertices of an input signed graph, containing positive and negative edges, must be assigned to those of a cyclic host graph in a one-to-one correspondence. The objective is to minimize the number of vertices that experience any conflict, defined as a situation in which a vertex connected by a negative edge lies on the shortest path to a vertex connected by a positive edge. We analyze the similarities and differences between the CMinVSSA and other variants belonging to the Cyclic Sitting Arrangement family. A comparison of optimal solutions on small instances reveals its unique structural properties, confirming that the CMinVSSA requires a dedicated optimization process. We then analyze different heuristic algorithms for this problem by adapting two state-of-the-art methods from the related literature: a Multi-Armed Bandit (MAB) hyper-heuristic and a Basic Variable Neighborhood Search (BVNS). Experimental results on 30 representative instances of varying size show that the BVNS outperforms the MAB, a result attributed to its flat-landscape navigation mechanisms, which prove effective in the count-based objective landscape of the CMinVSSA, while also highlighting the potential for developing fully problem-specific approaches.

Keywords: Graph Layout Problems · Cyclic Minimum Vertex Sum Sitting Arrangement · Combinatorial Optimization · Metaheuristics

1 Introduction

Graph Layout Problems (GLPs) are a family of problems in combinatorial optimization [4,13]. In these problems, the objective is to map the vertices of an input graph onto the vertices of a host graph such that a given objective function is optimized. GLPs are applicable to a wide range of real-world scenarios, including VLSI design, the assignment of virtual nodes to physical machines, and mathematical contexts such as numerical analysis and graph theory [13].

* Corresponding author

Among the GLPs, a well-known subfamily is the Sitting Arrangement (SA) family of problems, where input graphs contain vertices connected by positive or negative relationships (i.e., edges). Originally inspired by social networks, its first formal appearance in the literature is "Can everybody sit closer to their friends than their enemies?" [8]. This original formulation was proposed as a decision problem, asking whether the vertices of a signed graph can be mapped onto a host graph such that every vertex is placed closer to all its positive neighbors than to any of its negative neighbors. Subsequent research extended this problem to other host graph structures and shifted the focus from the decision question to the optimization variant, in which the goal is not to determine whether a conflict-free arrangement exists, but to find the arrangement that satisfies that condition for as many vertices as possible. Among the host graph structures studied in this context, the cycle is of particular interest due to its natural correspondence with real-world circular arrangements and its distinctive structural properties, and it has given rise to a dedicated line of research.

A conflict occurs for a given vertex whenever a negative neighbor (i.e., an adjacent vertex connected by a negative edge) lies on the shortest path to a positive neighbor (i.e., an adjacent vertex connected by a positive edge) in the cycle. Different aggregation strategies over vertex conflicts give rise to distinct problems. The Cyclic Minimum Sitting Arrangement problem (CMinSA) [16] sums the conflicts across all vertices, minimizing the total conflict load. The Cyclic Min-Max Sitting Arrangement (CMinMaxSA) [17] instead minimizes the maximum number of conflicts over any single vertex, penalizing solutions in which the conflict load is concentrated on a small subset of vertices.

This paper addresses the Cyclic Minimum Vertex Sum Sitting Arrangement (CMinVSSA), which adopts a different aggregation criterion: it minimizes the number of vertices that experience any conflict, regardless of how many conflicts each affected vertex accumulates. This formulation is motivated by scenarios in which a vertex is equally penalized by one or by many conflicts, making it preferable to concentrate all conflicts on as few vertices as possible.

The contributions of this paper are twofold. First, we analyze the similarities and differences between the CMinVSSA and the previously mentioned variants (CMinSA and CMinMaxSA). To do so, we conduct a systematic comparison of their optimal solutions on small instances to investigate the unique structural properties of the new formulation and determine whether it requires a dedicated optimization approach. Second, we solve the CMinVSSA by adapting the two state-of-the-art heuristic methods previously developed for the CMinSA and CMinMaxSA problems, providing an empirical evaluation of their performance on this new objective landscape.

The remainder of this paper is organized as follows. Section 2 formally defines the problem. Section 3 reviews the related literature, with emphasis on the most recent contributions and the state-of-the-art algorithms for other SA variants. Section 4 details the algorithmic strategies adapted for the CMinVSSA. The experimental results are presented in Section 5, first comparing the optimal

solutions across problem variants, and then evaluating the algorithms. Section 6 presents the conclusions and directions for future work.

2 Problem definition

In line with the standard framework for GLPs, the mathematical formulation of the CMinVSSA is articulated through four elements: the input graph, the host graph, the embedding function, and the objective function.

The input graph is a signed graph in which edges are labeled as either $+1$ or -1, representing positive and negative relationships between vertices, respectively. Formally, a signed graph G is defined as $G = (V_G, E_G^+, E_G^-)$, where V_G is the vertex set, and E_G^+ and E_G^- are the sets of edges labeled $+1$ and -1, respectively. These sets are disjoint, i.e., $E_G^+ \cap E_G^- = \varnothing$, and the complete edge set is $E = E_G^+ \cup E_G^-$. The neighborhood functions $\Gamma^+(u)$ and $\Gamma^-(u)$ return the sets of vertices connected to u by a positive or negative edge, respectively. To illustrate these concepts, consider the signed graph G_1 depicted in Fig. 1(a), with vertex set $V_{G_1} = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, \mathrm{E}\}$. The positive edges, $E_{G_1}^+ = \{\{\mathrm{A}, \mathrm{D}\}, \{\mathrm{A}, \mathrm{C}\}\}$, are represented by dotted lines labeled $+1$, and the negative edges, $E_{G_1}^- = \{\{\mathrm{A}, \mathrm{B}\}, \{\mathrm{A}, \mathrm{E}\}, \{\mathrm{C}, \mathrm{D}\}, \{\mathrm{D}, \mathrm{E}\}\}$, are represented by dashed lines labeled -1. For vertex A, the positive neighborhood set is $\Gamma^+(\mathrm{A}) = \{\mathrm{C}, \mathrm{D}\}$ and the negative neighborhood set is $\Gamma^-(\mathrm{A}) = \{\mathrm{B}, \mathrm{E}\}$.

The host graph is a cycle, i.e., a connected, undirected, finite, and simple graph $H = (V_H, E_H)$ in which every vertex has a degree of exactly two. Given two vertices $i, j \in V_H$, there exist exactly two paths between them in H. We denote the vertex set with fewer vertices as $\psi_H(i, j)$. In cases where both paths have equal size, the one that further optimizes the objective function is selected. Let us consider the cycle host graph H_1 depicted in Fig. 1(b), with vertex set $V_{H_1} = \{1, 2, 3, 4, 5\}$ and edge set $E_{H_1} = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{5, 1\}\}$. For vertices 1 and 4 in H_1, the two possible paths are $p_1 = (1, 2, 3, 4)$ and $p_2 = (1, 5, 4)$. Since p_2 has fewer vertices, the shortest path is $\psi_{H_1}(1, 4) = (1, 5, 4)$.

Given the signed input graph and the cycle host graph, an embedding (i.e., a solution) is defined by a bijective function $\varphi : V_G \to V_H$ that assigns each input vertex to a distinct host vertex.

The objective function of the CMinVSSA, is defined as the number of vertices with one or more conflicts. Mathematically,

$$\mathrm{CMinVSSA}(\varphi) = \big|\{u \in V_G \mid \epsilon(u, \varphi) > 0\}\big|, \tag{1}$$

where $\epsilon(u, \varphi)$ denotes the total number of conflicts for vertex u under embedding φ, and is defined as follows:

$$\epsilon(u, \varphi) = \sum_{v \in \Gamma_G^+(u)} \big|\{w \in \Gamma_G^-(u) \mid \varphi(w) \in \psi_H(\varphi(u), \varphi(v))\}\big|. \tag{2}$$

Notice that a conflict for a vertex u in φ occurs whenever a negative neighbor $w \in \Gamma^-(u)$ is embedded on the shortest path to a positive neighbor $v \in \Gamma^+(u)$.

Finally, the CMinVSSA problem is defined as the minimization of the Eq. 1 for all possible embeddings $\varphi \in \Phi$.

To illustrate the evaluation of the objective function, consider the embedding φ_1 of G_1 into H_1 depicted in Fig. 1(c), defined as $\varphi_1(\text{A}) = 1$, $\varphi_1(\text{B}) = 2$, $\varphi_1(\text{C}) = 3$, $\varphi_1(\text{D}) = 4$, and $\varphi_1(\text{E}) = 5$. For vertex A, the shortest path to its positive neighbor D is $\psi_{H_1}(1, 4) = (1, 5, 4)$, which contains vertex E, a negative neighbor of A. Similarly, the shortest path to its positive neighbor C is $\psi_{H_1}(1, 3) = (1, 2, 3)$, which contains its negative neighbor B. Therefore, under this embedding, vertex A experiences two conflicts. Vertices B and E cannot experience any conflicts, as they only possess negative incident edges, whereas a conflict requires the presence of both positive and negative neighbors. Vertex D has one conflict since the shortest path to A contains a negative neighbor. Finally, vertex C does not have any conflicts because its shortest path to A, $\psi_{H_1}(3, 1) = (3, 2, 1)$, does not include vertex D, which is its only negative neighbor. Consequently, the objective function value is $\text{CMinVSSA}(\varphi_1) = 2$, since exactly two vertices (A and D) experience one or more conflicts.

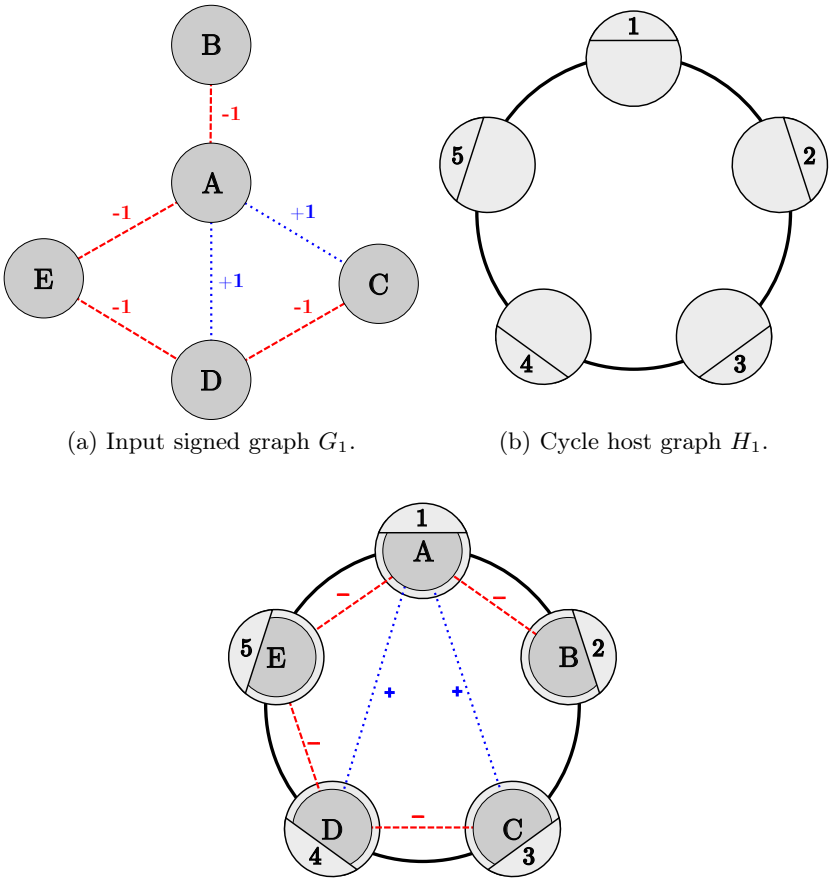
3 Literature review

The SA family of problems was introduced in 2011 in a work titled "Can everybody sit closer to their friends than their enemies?" [8], formulated as a decision problem with an algorithm for the path host graph. Several theoretical studies followed, establishing that the problem is NP-complete [6] and extending its study to other host graphs, including cycles [2] and trees [1].

An optimization variant on the path host graph was introduced in 2015 [12], proposing multiple constructive methods. In 2020, a more effective heuristic exploited the theoretical properties of the problem [10], with a constructive phase based on positive cliques, subsets of vertices in which every pair is connected by a positive edge, complemented by a local search based on an insertion operator. The proposal significantly outperformed previous ones, establishing a new state-of-the-art algorithm.

The use of a cycle graph as a host graph was originally proposed in [2], using the continuous space of the circumference as a host graph. Later, the first heuristic addressing an optimization variant of the problem on a cycle host graph (denoted as CMinSA) was proposed in 2022 [16], adapting the state-of-the-art algorithm from the path variant [10] and demonstrating that optimizing the SA problem on a path does not transfer directly to a cycle, as the cyclic structure requires a tailored algorithmic approach. This algorithm was subsequently outperformed in 2025 [19] by a Multi-Armed Bandit (MAB) hyper-heuristic, which chooses among different available constructive strategies at each iteration based on their relative performance throughout the search.

In 2025, the CMinMaxSA variant was introduced [17], in which individual vertex conflicts are aggregated using the maximum rather than the sum. Alongside the problem definition, the authors proposed the first algorithm for this variant, based on Variable Formulation Search (VFS) [11], using four alternative



(a) Input signed graph G_1.

(b) Cycle host graph H_1.

(c) Embedding φ_1 of G_1 into H_1.

Fig. 1. Representations of G_1 and H_1, and the embedding φ_1.

problem formulations to escape local optima and continue improving solutions even in flat fitness landscapes where all neighbors share the same objective function value.

In 2026, a second algorithm for the CMinMaxSA was presented as a Basic Variable Neighborhood Search (BVNS) extending the VFS framework [18]. Three main enhancements were introduced: a constructive method specifically designed for the problem structure, a problem-aware shake procedure replacing random perturbation with targeted moves guided by problem-specific information, and efficient delta-evaluation strategies that reduce the computational cost of assessing neighboring solutions. Alongside the heuristic, the first mathematical programming model for a cyclic GLP was formulated and evaluated using commercial solvers: Hexaly, Gurobi, and CPLEX.

In summary, research on the SA problem family has progressively encompassed new host graphs, alternative aggregation criteria, and increasingly sophisticated algorithmic strategies, with trajectory-based metaheuristics emerging as the dominant approach. However, no prior work has addressed the criterion of minimizing the sum of vertices with a conflict rather than the total or maximum conflict load. This gap in the literature motivates the analysis of the CMinVSSA in the present paper.

4 Evaluated heuristic algorithms

As discussed in the literature review, the current state of the art for the Cyclic Sitting Arrangement problem family comprises two algorithms: the Multi-Armed Bandit (MAB) hyper-heuristic [19] and the Basic Variable Neighborhood Search (BVNS) [18], originally developed for the CMinSA and CMinMaxSA variants, respectively. In this section, we describe the specific mechanics of both methods in greater detail, as they will be adapted and evaluated on the CMinVSSA problem. Because their core search strategies are highly effective for cyclic host graphs, their adaptation to this new variant is straightforward, requiring only the substitution of the objective function while leaving all other structural components unchanged.

4.1 Multi-Armed Bandit

The Multi-Armed Bandit (MAB) algorithm, originally proposed in [19] for the CMinSA problem, operates as a selection hyper-heuristic [3]. It dynamically selects among three constructive heuristics based on their historical performance, balancing the exploration of new strategies with the exploitation of those that have proven effective. Arm selection is guided by the UCB1 framework [20], employing a Lower Confidence Bound formulation that favors arms with low average objective values while maintaining sufficient exploration of less-tried alternatives. Once an initial solution is generated by the selected arm, it undergoes a common improvement phase based on Variable Neighborhood Descent (VND) [7]. This phase systematically explores a swap neighborhood followed by an insert neighborhood under a first-improvement strategy.

The three arms differ exclusively in their construction phase. The first arm, denoted as "*Arm 1 – Cliques*," generates candidate cliques from the positive edges of the input graph using the Bron–Kerbosch algorithm [5], embedding them sequentially in descending order of size. By keeping positively connected vertices in close proximity, this strategy reduces the likelihood of conflicts. The second arm, named "*Arm 2 – Louvain*," follows a similar philosophy but replaces clique detection with the Louvain community detection method [14], which is less computationally demanding. The detected communities are embedded greedily by selecting, at each step, the community that minimizes the partial objective function value. The third arm, referred to as "*Arm 3 – McAllister*," adopts a different approach by selecting individual vertices based on a greedy criterion.

This criterion uses a linear combination of the number of assigned and unassigned positive and negative neighbors. The tuned expression for this arm reduces to $g_3(\varphi', u) = |A^-(u)| - |U^-(u)|$, where $A^-(u)$ and $U^-(u)$ denote the sets of assigned and unassigned negative neighbors of vertex u, respectively. Consequently, this arm prioritizes vertices with a high number of unassigned negative neighbors.

4.2 Basic Variable Neighborhood Search

The Basic Variable Neighborhood Search (BVNS) algorithm, originally proposed in [18] for the CMinMaxSA problem, is a trajectory-based metaheuristic that systematically explores a sequence of neighborhoods to escape local optima [7]. Starting from an initial solution, the algorithm alternates between a *Shake* perturbation and a *Local Search* improvement phase. It resets the neighborhood index upon finding an improvement and increments it otherwise, continuing until a predefined maximum index $k_{\max}$ is reached.

The initial solution is constructed by a greedy heuristic that iteratively selects unassigned vertices according to an urgency function $\beta(u)$:

$$\beta(u) = w_1 \cdot |\Gamma_A^+(u)| + w_2 \cdot |\Gamma_U^+(u)| + w_3 \cdot |\Gamma_A^-(u)| + w_4 \cdot |\Gamma_U^-(u)|, \tag{3}$$

where $\Gamma_A^+(u)$, $\Gamma_U^+(u)$, $\Gamma_A^-(u)$, and $\Gamma_U^-(u)$ denote the sets of assigned and unassigned positive and negative neighbors of vertex u, respectively, and w_1, w_2, w_3, $w_4 \in \mathbb{R}$ are empirically tuned weights. In the original CMinMaxSA configuration, the weights reduce the function to $\beta(u) = 2 \cdot |\Gamma_A^+(u)| - |\Gamma_U^+(u)|$, meaning that only positive adjacency is relevant. Consequently, the heuristic prioritizes vertices with many already-assigned positive neighbors, keeping positively connected vertices close together in the embedding.

The local search applies a swap operator under a first-improvement strategy. Since Min-Max problems often exhibit flat fitness landscapes, the algorithm incorporates a tie-breaking criterion that progressively compares the distribution of conflicts across vertices to distinguish between equally-valued solutions. The shake procedure applies targeted perturbations of increasing size, relocating the vertices with the highest individual conflict contribution to their best available host vertex.

5 Experimental results

This section presents the empirical framework designed to evaluate the CMinVSSA problem and the adapted heuristic algorithms. The experimentation is divided into two main studies. Section 5.1 conducts a structural comparison of the CMinVSSA with the CMinSA and CMinMaxSA variants by analyzing their optimal solution sets. Specifically, we aim to determine if an optimal solution for any of the previous variants is also optimal for the CMinVSSA. Then, Section 5.2

evaluates the computational performance of the MAB and BVNS algorithms introduced in Section 4 adapted for the CMinVSSA.

Both studies use instances from a publicly available repository [15], which contains graphs of varying size, density, and edge composition. For the first study, we focus on a subset of 5 small instances (10 to 11 vertices). This size restriction is necessary because our methodology requires computing the complete set of optimal solutions via exhaustive backtracking, a task that becomes computationally intractable for larger graphs due to the NP-Hard nature of these problems. The second study, dedicated to evaluating the heuristic algorithms, utilizes a set of 30 representative instances. To ensure structural diversity and representativeness, these instances were selected using the software and clustering methodology presented in [9], which identifies the k most diverse instances within a repository based on their topological properties. In this selected set, the number of vertices ranges from 10 to 663. To ease the interpretation of the results, these 30 instances are divided into three subsets according to their size: small (10–30 vertices), medium (40–90 vertices), and large (>100 vertices).

5.1 Analysis of optimal solution spaces

To understand the structural uniqueness of the solutions of the CMinVSSA we examine whether an optimal solution for the CMinSA or CMinMaxSA variants is inherently optimal for the CMinVSSA. We compute the complete set of optimal solutions via backtracking for each variant and identify their intersection. While coincident sets would imply structural equivalence, disjoint sets confirm that the CMinVSSA defines a distinct optimization landscape. Table 1 presents the comparison between the CMinSA and the CMinVSSA, showing the total number of optimal solutions for each problem and the number of solutions that are optimal for both problems.

Table 1. Comparison of optimal solution sets: CMinSA vs. CMinVSSA.

Instance	#Optimal sol. CMinSA	#Optimal sol. CMinVSSA	#Optimal sol. both problems
complete_004_10x45_100_20	168	216	24
complete_002_10x45_100_50	4	48	0
complete_006_10x45_100_80	8	36	4
381-R_edges-0.75-jgl011.hwb	176	96	0
487-R_vertices-0.15-jgl011.hwb	11520	81792	11520

The data reveal that the optimal solution sets are largely disjoint. In most instances, a solution that minimizes the total number of conflicts (CMinSA) fails to minimize the number of affected vertices (CMinVSSA). For instance, in *complete_002*, there is no overlap between the two sets. This divergence occurs because the CMinVSSA is indifferent to the number of conflicts a single vertex accumulates; it prioritizes concentrating all conflicts on a few vertices to keep

the rest conflict-free. In contrast, the CMinSA penalizes every individual conflict, often spreading them across more vertices to reduce the total sum.

A similar, yet more intense trend is observed in Table 2 when comparing the CMinMaxSA (minimizing the maximum conflict load) and the CMinVSSA.

Table 2. Comparison of optimal solution sets: CMinMaxSA vs. CMinVSSA.

Instance	#Optimal sol. CMinMaxSA	#Optimal sol. CMinVSSA	#Optimal sol. both problems
complete_004_10x45_100_20	384	216	0
complete_002_10x45_100_50	32	48	0
complete_006_10x45_100_80	8	36	0
381-R_edges-0.75-jgl011.hwb	1516	96	0
487-R_vertices-0.15-jgl011.hwb	103680	81792	34560

The separation here is even more pronounced: in four out of five instances, the optimal sets are entirely disjoint (zero overlap). The CMinMaxSA objective actively seeks to distribute conflicts evenly to avoid high peaks at any single vertex, which is opposed to the CMinVSSA strategy of concentrating conflicts on as few vertices as possible.

These results confirm that the CMinVSSA represents a fundamentally different optimization goal. The lack of overlap between the optimal solution sets proves that the CMinVSSA cannot be solved through existing variants and justifies the necessity of the dedicated heuristic evaluation presented next.

5.2 Computational evaluation of state-of-the-art algorithms

Having established that the CMinVSSA constitutes a structurally distinct problem, we now evaluate the performance of the two adapted state-of-the-art algorithms: the MAB [19] and BVNS [18]. All experiments were conducted on the 30-instance benchmark described above, using a time limit of 600 s per instance. The MAB algorithm is evaluated both as a complete hyper-heuristic framework and through its three individual arms: *Arm 1 – Cliques*, *Arm 2 – Louvain*, and *Arm 3 – McAllister*. Since the architecture allows each strategy to be executed independently, this comparison aims to determine whether the hyper-heuristic effectively manages the trade-off between the different constructive approaches when applied to the CMinVSSA. The results are summarized in Table 3.

The results reveal a consistent pattern: each strategy performs best on a specific instance size. *Arm 3* leads on small instances, *Arm 1* on medium, and *Arm 2* on large. This behavior can be attributed to the computational profile of each method. *Arm 3* is the most lightweight, enabling a higher number of construction iterations within the time limit, which benefits smaller instances where search diversity is critical. *Arm 1* exploits the positive graph structure directly, but exact clique enumeration is computationally expensive and becomes a bottleneck as the graph size increases. *Arm 2* follows a similar grouping philosophy

Table 3. Comparison of the MAB framework and its independent arms.

Instance set	Algorithm	Avg. O.F.	Avg. CPU T.	# Best
Small (10)	Arm 1 – Cliques	6.60	300.50	9
	Arm 2 – Louvain	6.90	300.62	8
	Arm 3 – McAllister	**6.40**	300.74	**10**
	MAB	**6.40**	300.73	**10**
Medium (10)	Arm 1 – Cliques	**21.40**	364.51	7
	Arm 2 – Louvain	24.00	363.25	6
	Arm 3 – McAllister	22.20	368.29	**8**
	MAB	22.30	363.40	7
Large (10)	Arm 1 – Cliques	33.00	540.51	6
	Arm 2 – Louvain	**32.30**	548.05	**9**
	Arm 3 – McAllister	34.20	614.34	2
	MAB	32.60	529.37	8
Total (30)	Arm 1 – Cliques	**20.33**	401.84	22
	Arm 2 – Louvain	21.07	403.97	23
	Arm 3 – McAllister	20.93	427.79	20
	MAB	20.43	397.83	**25**

but at a much lower computational cost, making it the most effective strategy for large instances. Across all subsets, the MAB framework consistently matches or closely approaches the performance of its best individual arm, successfully balancing the time budget without relying on a single strategy.

Following the evaluation of the MAB framework, the leading approach for the CMinSA, we now turn our attention to the BVNS, which currently represents the state of the art for the CMinMaxSA. For this problem, we investigated whether the urgency function weights required adaptation for the CMinVSSA objective. Preliminary tests revealed that re-tuning the constructive weights $w_1, \ldots, w_4$ (see Eq. 3) yielded no significant improvements in solution quality. This indicates that the constructive mechanism is inherently robust across both variants and that the specific weighting of positive and negative adjacencies is not a determining factor for this problem. Therefore, the urgency function is defined as $\beta(u) = 2 \cdot |\Gamma_A^+(u)| - |\Gamma_U^+(u)|$, the same one used in the CMinMaxSA proposal [18].

Table 4. Final performance comparison: MAB vs. BVNS.

Instance set	Algorithm	Avg. O.F.	Avg. CPU T.	# Best
Small (10)	MAB	6.40	300.73	8
	BVNS	**6.11**	300.00	**10**
Medium (10)	MAB	22.30	363.40	5
	BVNS	**18.01**	**300.00**	**9**
Large (10)	MAB	32.60	529.37	5
	BVNS	**29.91**	**241.10**	**10**
Total (30)	MAB	20.43	397.83	18
	BVNS	**18.01**	**280.37**	**29**

The results shown in Table 4 demonstrate that the BVNS consistently outperforms the MAB across all instance sizes, achieving superior solution quality in significantly less CPU time. This advantage is primarily attributable to the flat-landscape navigation mechanisms embedded within the BVNS. Both its tie-breaking criterion and its targeted shake procedure are explicitly designed to discriminate between equally-valued solutions and avoid stagnation when multiple neighboring solutions share the same objective value, a common occurrence in the CMinVSSA due to its non-summation aggregation.

In contrast, the MAB was originally tailored for the CMinSA, where the sum-based objective function provides a fine-grained signal to guide the search. While the strategy of MAB, consisting of grouping positively connected vertices, effectively reduces the total volume of conflicts, it is less aligned with the CMinVSSA objective, which remains indifferent to the intensity of conflicts per vertex as long as the number of affected vertices is minimized. Consequently, the trajectory-based approach of the BVNS proves more effective for this specific variant.

6 Conclusions

This paper has introduced and analyzed the Cyclic Minimum Vertex Sum Sitting Arrangement (CMinVSSA). Our structural study confirms that minimizing the number of affected vertices creates a distinct optimization landscape compared to minimizing total or maximum conflict loads. The analysis of optimal solution sets revealed that CMinMaxSA and CMinVSSA are particularly divergent, sharing almost no common optima, while CMinVSSA generally admits a larger set of optimal arrangements due to its indifference to the conflict amount per vertex.

Regarding algorithmic performance, the BVNS emerged as the superior approach, significantly outperforming the MAB hyper-heuristic. The key to this success lies in the ability of BVNS to navigate flat fitness landscapes through its tie-breaking and targeted shake mechanisms, features that are essential when the objective function provides a coarse-grained signal, as in count-based aggregation.

Future research should focus on three areas: (i) providing a formal complexity proof for the CMinVSSA, (ii) developing lower and upper bounds to assess the heuristic gap, and (iii) designing metaheuristics with operators that explicitly reward the total elimination of conflicts in a vertex.

Acknowledgement: This research has been partially supported by grants: PID2021-125709OA-C22; CIAICO/2021/224; C1PREDOC24-047 and M2988; TEC-2024/COM-404; TSI-100930-2023-3 (MCA07); and MCA06.

References

1. Becerra, R., Caro, C.T.: The Sitting Closer to Friends than Enemies Problem in Trees (2021). https://doi.org/10.48550/arXiv.1911.11494

2. Benítez, F., Aracena, J., Caro, C.T.: The sitting closer to friends than enemies problem in the circumference. arXiv preprint arXiv:1811.02699 (2018). https://doi.org/10.48550/arXiv.1811.02699
3. Burke, E.K., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Özcan, E., Qu, R.: Hyper-heuristics: a survey of the state of the art. Journal of the Operational Research Society **64**(12), 1695–1724 (2013)
4. Cavero, S., Pardo, E.G., Duarte, A., Martí, R.: Graph Layout Problems, pp. 1–31. Springer Nature Switzerland, Cham (2025)
5. Conte, A.: Review of the bron-kerbosch algorithm and variations. School of Computing Science, University of Glasgow pp. 1–9 (2013)
6. Cygan, M., Pilipczuk, M., Pilipczuk, M., Wojtaszczyk, J.O.: Sitting closer to friends than enemies, revisited. In: International Symposium on Mathematical Foundations of Computer Science. pp. 296–307. Springer (2012)
7. Hansen, P., Mladenović, N., Brimberg, J., Pérez, J.A.M.: Variable neighborhood search. Springer (2019)
8. Kermarrec, A.M., Thraves, C.: Can everybody sit closer to their friends than their enemies? In: International Symposium on Mathematical Foundations of Computer Science. pp. 388–399. Springer (2011)
9. Martín-Santamaría, R., Cavero, S., Herrán, A., Duarte, A., Colmenar, J.M.: A practical methodology for reproducible experimentation: an application to the double-row facility layout problem. Evolutionary Computation pp. 1–36 (2023)
10. Pardo, E.G., García-Sánchez, A., Sevaux, M., Duarte, A.: Basic variable neighborhood search for the minimum sitting arrangement problem. Journal of Heuristics **26**(2), 249–268 (2020)
11. Pardo, E.G., Mladenović, N., Pantrigo, J.J., Duarte, A.: Variable Formulation Search for the Cutwidth Minimization Problem. Applied Soft Computing **13**(5), 2242–2252 (2013)
12. Pardo, E.G., Soto, M., Thraves, C.: Embedding signed graphs in the line. Journal of Combinatorial Optimization **29**(2), 451–471 (2015)
13. Petit, J.: Experiments on the minimum linear arrangement problem. Journal of Experimental Algorithmics (JEA) **8** (2003)
14. Que, X., Checconi, F., Petrini, F., Gunnels, J.A.: Scalable community detection with the louvain algorithm. In: 2015 IEEE international parallel and distributed processing symposium. pp. 28–37. IEEE (2015)
15. Robles, M., Cavero, S.: Signed Graph Collection (2025), https://github.com/MRoblesR/signed-graphs
16. Robles, M., Cavero, S., Pardo, E.G.: BVNS for the minimum sitting arrangement problem in a cycle. In: International Conference on Variable Neighborhood Search. pp. 82–96. Springer (2022)
17. Robles, M., Cavero, S., Pardo, E.G.: Variable formulation search for the cyclic min-max sitting arrangement problem. In: International Conference on Variable Neighborhood Search. pp. 134–149. Springer (2025)
18. Robles, M., Cavero, S., Pardo, E.G.: Mathematical formulation and heuristic methods for the min-max sitting arrangement problem in the cycle. European Journal of Operational Research (2026)
19. Robles, M., Cavero, S., Pardo, E.G., Cordón, O.: Multi-armed bandit for the cyclic minimum sitting arrangement problem. Computers & Operations Research **179**, 107034 (2025)
20. Slivkins, A.: Introduction to multi-armed bandits. Foundations and Trends in Machine Learning **12**(1-2), 1–286 (2019)