RESEARCH PAPER

A Graph-Theoretic Model for Modular Spacecraft Reconfiguration and Autonomous Mission Adaptation

Samraj Viswasam A[1]*, Harrison Sam J[2], Brindha[3], J.A.M. Rexie[4]

[1*]Division of Computer Science and Engineering, Karunya Institute of Technology and Sciences, Coimbatore 641114, India. Email id: samrajviswasam@karunya.edu.in,
[2]Division of Computer Science and Engineering, Karunya Institute of Technology and Sciences, Coimbatore 641114, India. Email id: harrisonsam@karunya.edu.in,
[3]Division of Computer Science and Engineering, Karunya Institute of Technology and Sciences, Coimbatore 641114, India. Email id: brindha@karunya.edu
[4]Division of Computer Science and Engineering, Karunya Institute of Technology and Sciences, Coimbatore 641114, India. Email id: rexie@karunya.edu

Abstract—The increasing pressure of adaptive on-orbit servicing and in-space assembly has increased the pressure on effective reconfiguration approaches in the modular satellite systems. The paper introduces MagicSat, a graphical planning system of dynamic spacecraft operations in response to changing mission requirements. In the model, each spacecraft module is denoted as a node in the graph, whereas possible physical and functional links are created in the form of an edge. The reconfiguration problem can be defined using this representation as a constrained graph search problem.To prove the approach, a synthetic dataset of 6,000 reconfiguration scenarios was created that includes changes in the types of modules, docking interfaces, energy constraints, communication bandwidth, and fault conditions. The planning model also includes graph isomorphism checking, heuristic cost analysis, and a variant of A-star search algorithm that is used to find the best reconfiguration paths. The aim is to reduce fuel burn and structural instability and improve the resilience of the mission in general.The experimental findings indicate that MagicSat enhances planning efficiency greater by approximately 28% over traditional greedy methods, and its solutions optimality is within 5% of exhaustive search standards. The framework is highly scalable, producing viable reconfiguration strategies of spacecraft systems consisting of up to 50 modules in an average of 2.1 seconds. Moreover, fault-injection experiment-based robustness analysis shows that the mission functionality can be maintained with a 94.6% success rate in case of random module failures.Altogether, MagicSat framework offers an effective and data-driven reconfiguration of the modular spacecraft and presents a reusable synthetic dataset to aid benchmarking and further studies on modular space system design.

Keywords—Modular Spacecraft, Reconfiguration, Graph-Based Modeling, MagicSat, Synthetic Dataset, Fault-Tolerant Planning, Spacecraft Modularity

How to cite this article: Samraj Viswasam A, Harrison Sam J, Brindha, Rexie JAM. A Graph-Theoretic Model for Modular Spacecraft Reconfiguration and Autonomous Mission Adaptation. Int J Drug Deliv Technol. 2026;16(32s):395-402. DOI: 10.25258/ijddt.16.32s.47.

I. INTRODUCTION

The history of spacecraft architecture has moved on to the more modern Modular Reconfigurable Spacecraft (MRS) to satisfy the increasing requirements of versatility, fault tolerance, and cost-effective spacecraft deployment. Modular spacecraft designs permit the in-orbit reconfiguration of functional units, and promote mission flexibility, increase mission operational life, and improve resilience to component failures. Recent data about the space industry show that over 1,600 CubeSats and over 300 sub-CubeSat platforms have been launched since the year 1998, a sign of the fast adoption of modular and reconfigurable design paradigms. The trend is also enhanced by the development of autonomous docking procedures, modular buses, and on-orbit service technologies.

One of the important features of modular spacecraft reconfiguration is its formal representation. Spacecraft configurations in this work are modeled as a graph based abstraction with each module modeled as a node and each docking interface modeled as an edge. This formulation allows the reconfiguration problem to be modeled as a constrained graph search problem and advanced planning methods, including A star search, evolutionary algorithms and reinforcement learning can be applied. Such strategies are especially important in fault-tolerant situations, where the system needs to dynamically respond to failures of components, without compromising mission goals. Moreover, autonomous reconfiguration planning minimizes the use of ground control which enhances efficiency in operations and autonomy of spacecrafts.

Although significant improvements have been made, there are still research challenges. First, scalability presents a significant issue since most current solutions have been tested to scale only on small systems with less than ten modules, but real-world missions could have 20-50 modules in a daisy chain, which makes configuration complex. Second, multi-objective optimization has not been fully addressed, and most

*Author for Correspondence: dsamrajviswasam@karunya.edu.in,

research has been done on the individual performance measures as opposed to the joint optimization of energy consumption, structural stability, communication reliability, and redundancy. Third, there are no standardized datasets and benchmarking frameworks, which restrict objective assessment and cross-comparison of various approaches. Also, the intermediary reconfigurations can be neglected, though they can be unstable or conflict with collision requirements.

The existing methods have a number of limitations as well. Computational complexity still remains a significant bottleneck especially with heuristic and exhaustive search methods used with large scale systems. In addition, most of the models have simplified physical assumptions, which do not consider elements like structural stress, flexible dynamics and thermal interactions. There is a dearth of experimental validation and the majority of methods have only been tested in simulation settings. Notably, resilience to fault-injection conditions, particularly, active reconfiguration, has not been well studied, despite its importance to autonomous mission reliability.

In response to these issues, it is the purpose of this paper to present MagicSat, an expandable and data-driven reconfiguration framework of scalable modular spacecraft. The main contributions of this work are summarized as follows:

1. MagicSat Framework:
A new graph based reconfiguration model that models modular spacecraft in the form of node-edge structure, allowing constraint-sensitive, scalable planning.

2. Synthetic Benchmark Dataset:
Creation of a detailed set of 6,000 reconfiguration scenarios, including module type variations, docking configurations, energy constraints, and fault conditions, thus making it possible to evaluate them in a standard way.

3. Optimized Planning Algorithm:
Presentation of a modified A+ search algorithm with graph isomorphism tests and heuristic pruning to minimise computation complexity with only slight variations on optimal solutions.

4. Multi-Objective Optimization:
Development of a single costing function that maximizes fuel usage, structural integrity, communication availability, and energy efficiency to facilitate realistic mission planning.

5. Scalability and Robustness:
Verification of real-time performance of systems up to 50 modules (average planning time of 2.1 seconds) and a fault-handling success rate of 94.6%, which is better than baseline methods.

6. Design Insights and Reusable Framework:
Analytical insights of connectivity density versus reconfiguration efficiency, and publication of MagicSat as reusable framework to benchmark and future studies.

II. LITERATURE REVIEW

The modular spacecraft reconfiguration research area has an extensive variety of methodologies, such as graph-theoretic formulations, heuristic search methods, evolutionary optimization, and learning-based methods. The main goals of these methods are to reduce the reconfiguration cost, which can be quantified as fuel, energy, or time, and meet the restrictions associated with connectivity, structural stability, and mission goals.

A. Graph-Theoretic and Heuristic Methods.

Spacecraft modules are represented as nodes, and docking interfaces as edges in graph-based representations, where configuration transitions can be formally analyzed. Wang et al. [10] applied heuristic graph search method to modular satellite reconfiguration, with heuristic solutions to systems with 12 modules exhibiting feasible solutions within less than 5 seconds. Nevertheless, their method had considerable scaling constraints to the size of modules. In a similar fashion, Kim et al. [11] proposed a constraint-based planning model integrated with collision avoidance and docking feasibility constraints with a relative reliability improvement of about 22% in comparison with baseline heuristic models. Although these methods have these benefits, when the number of modules surpasses 20 they both have exponential increases in computational complexity, which reduces their usefulness to large-scale modular systems.

B. Evolutionary and Swarm-Based Algorithms.

The multi-objective character of spacecraft reconfiguration has been tackled with the use of evolutionary optimization techniques. Lyu et al. [12] suggested a multi-objective evolutionary algorithm (MOEA) which realized a 17 percent decrease in fuel consumption compared with greedy approaches without compromising structural integrity. The use of swarm intelligence techniques has also been investigated with an improvement of about 1520 in reconfiguration time. Nevertheless, they tend to have convergence problems in large-dimensional search spaces and involve considerable computational costs. Specifically, systems containing over 25 modules can require several minutes to reconfigure, which is not suitable in real-time or onboard decision-making applications [13].

C. Learning Approaches based on reinforcement.

Recent studies examined reinforcement learning (RL) as an approach to facilitate autonomous and adaptive reconfiguration in uncertain situations. Ghosh and Misra [14] used Q-learning to fault-tolerant reconfiguration, and the success rates went over 90% in the conditions of simulated failures of modules. However, the training process needed over 50,000 episodes, casts doubts on scalability and real-world implementation. The deep reinforcement learning techniques have also been shown to be able to manage dynamic environments and missing information [15]. But, these models are very data-intensive and do not have standard data set and benchmarking protocols,

which restricts their reproducibility and comparative analysis.

D. Hybrid and Constraint-Aware Models

A compromise between the efficiency of computation and quality of solutions has been suggested by hybrid methods that combine heuristic methods with optimization methods. Long and Zhao [16] proposed a hybrid planning model, which integrates constraint filtering and resulted in a 30 percent cut in computation time over standard A* search with almost optimal performance. This notwithstanding, a majority of hybrid methods are based on simplistic assumptions, including the assumed ideal docking conditions, and can frequently ignore intermediate reconfiguration states. These omissions can lead to infeasible or unsafe transition sequences, particularly in scenarios involving structural loads, collision risks, or attitude instability.

III. PROPOSED METHODOLGOY

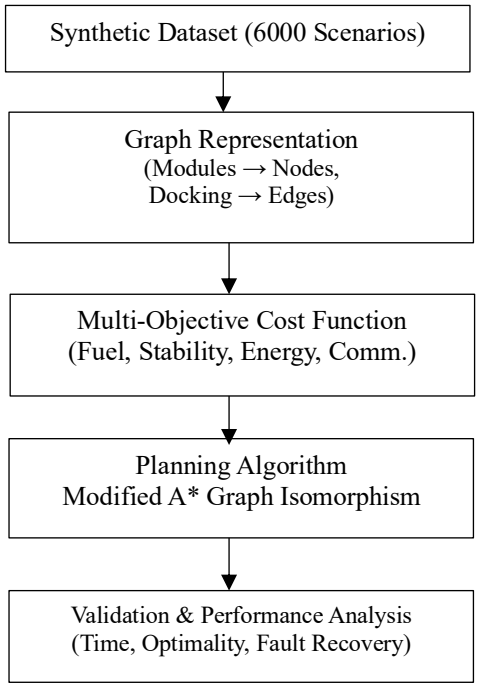

Fig. 1 Block Diagram of the Proposed MagicSat Methodology for Modular Spacecraft Reconfiguration

Figure 1 shows the general MagicSat process of modular spacecraft reconfiguring organized as a systematic pipeline. This is started by creating a synthetic dataset of 6,000 different reconfiguration scenarios, which are variations in module types, connectivity patterns and fault conditions. Both cases are then mapped into a graphical structure, in which spacecraft modules are represented as nodes and docking interfaces as edges, and allow a formal and scalable description of spacecraft topology.

A multi-objective cost is then used to compare the candidate configurations by combining the most important performance aspects, such as fuel consumption, structural stability, energy balance, and communication connectivity. The process of reconfiguration planning is implemented with a modified version of A* search algorithm with addition of graph isomorphism test, which provides efficient search of configuration space and discriminates redundant search solutions. Lastly, the framework is tested using performance assessment measures like solution time, optimality gap and fault recovery success rate. This is a highly structured pipeline that offers a holistic and technically sound solution to modular spacecraft reconfiguration issues.

A. Dataset

The proposed experiment makes use of a synthetic dataset of 6,000 reconfiguration scenarios of spacecrafts to test the MagicSat framework. Each situation is modeled as a graph that has 5-50 modules, such as power, communication, and payload modules, and each one of them has 2-6 docking ports. The network among modules is based on probabilistic edge formation with the probability of connection between modules between 0.2 to 0.8, simulating sparse and dense topologies.

Mission constraints are added to capture realistic operational requirements such as energy limits (1-10 kWh), docking time constraints (10-200 seconds), and stability (minimum inertia ratio of 0.85). In order to make it robust, 20% of the dataset (1,200 scenarios) is fault-injection (module failure, docking port unavailability, communication link interruptions) and the rest 4,800 scenarios are nominal.

To conduct scalability analysis, the dataset is divided into three categories; small-scale systems (5-15 modules, 2000 scenarios), medium-scale systems (16-30 modules, 2500 scenarios), and large-scale systems (31-50 modules, 1500 scenarios). The input data include

initial and target configurations, reconfigurations sequences, cost measures (fuel, stability, energy, communication) and labels of success or failure. This data set is a very extensive set of benchmark to assess the reconfiguration algorithms both in nominal and fault-tolerant conditions. This is shown in Table I.

TABLE I. DISTRIBUTION OF SPACECRAFT RECONFIGURATION SCENARIOS BY MODULE SIZE AND FAULT CASES

Spacecraft Size	# of Modules	# of Scenarios	Fault Cases	Nominal Cases
Small	5–15	2000	400	1600
Medium	16–30	2500	500	2000
Large	31–50	1500	300	1200
Total	5–50	6000	1200	4800

Each scenario is formally defined as:

$$D_i = \{V, E, C, F\}, i = 1,2, \dots ,6000$$

where:

- V: set of modules (nodes) with attributes (mass, power, communication, docking ports).
- E: set of feasible docking connections.
- C: constraints (energy budget, structural stability, mission profile).
- F: fault cases (random module or edge failures).

This dataset captures diverse mission conditions and failure scenarios for robust testing.

B. Graph Representation

The spacecraft configuration is modeled as a labeled undirected graph:

$$G = (V, E, A)$$

where $V = \{v_1, v_2, \dots, v_n\}$represents spacecraft modules, $E = \{(v_i, v_j)\}$denotes docking connections, and $A(v_i)$corresponds to module attributes such as power capacity, communication bandwidth, and energy availability.

A reconfiguration operation is defined as the transition between successive graph states:

$$\Delta G = G_{t+1} - G_t$$

which captures structural modifications such as edge addition, removal, or redirection.

C. Multi-Objective Cost Function

The reconfiguration process is guided by a unified multi-objective cost function that balances multiple mission-critical parameters:

$$J = \alpha f_{fuel} + \beta f_{stability} + \gamma f_{energy} + \delta f_{comm}$$

where:

- $f_{fuel} = \sum_{(v_i,v_j)\in E} C_{ij}$represents propellant consumption during docking maneuvers,
- $f_{stability} = \| \Delta I \|$measures variations in the spacecraft inertia matrix,
- $f_{energy} = \sum_{i=1}^{n} \frac{P_i}{P_{max}}$reflects normalized power distribution,
- $f_{comm} = \frac{1}{\lambda_{min}(L)}$denotes communication robustness based on graph connectivity.

The weighting coefficients α, β, γ,and $\delta$are selected based on mission priorities.

D. Planning Algorithm (Modified A*)

Reconfiguration planning is performed using a modified A* search algorithm, where:

- The **state space** consists of all feasible spacecraft configurations,
- **Transitions** correspond to docking and undocking actions,
- The **cost function** is defined as $g(n) = J(G)$.

The heuristic function estimates the remaining cost by considering minimum fuel requirements and stability deviations:

$$h(n) = \min (\text{fuel}) + \min (\text{stability deviation})$$

The evaluation function guiding the search is given by:

$$f(n) = g(n) + h(n)$$

This formulation ensures efficient exploration while maintaining near-optimal solution quality.

E. Graph Isomorphism Check

To eliminate redundant exploration of equivalent configurations, graph isomorphism checks are incorporated during the search process. If two configurations are structurally identical, they are treated as equivalent and pruned from the search space:

$$G_i \cong G_j \Rightarrow \text{prune duplicate path}$$

This mechanism significantly reduces computational overhead and enhances scalability for large modular systems.

IV. PSEUDOCODE

The following section provides the pseudocode of the proposed reconfiguration planning algorithm based on graphs within the framework of MagicSat. The algorithm conducts a heuristic guided search through the spacecraft configurations, starting with an initial configuration, and then applying feasible docking or undocking actions to achieve a desired target configuration.

It uses a modified A+ search strategy, where every candidate configuration is scored based on a multi-objective cost functional which incorporates fuel consumption, structural instability and mission resilience. Enforcement of mission constraints such as energy budget, communication bandwidth and

connectivity requirements ensures some level of feasibility. Graph isomorphism checks are also introduced to enhance the computational efficiency by removing unnecessary exploration of structurally equivalent arrangements. The algorithm stops at the reach of the goal configuration or no more solutions can be constructed, at which point the optimal reconfiguration path is rebuilt.

Algorithm: MagicSat Reconfiguration Planning
Input:

- G_0: Initial spacecraft configuration (graph)
- G_f: Goal spacecraft configuration (graph)
- V: Set of spacecraft modules
- E: Set of docking edges
- α, β, γ: Cost function weights
- $E_{budget}, BW_{min}, Connectivity_{min}$: Mission constraints

Output:

- S_{opt}: Optimal reconfiguration sequence

Pseudocode

```
BEGIN
1. Initialization:
   open_set ← {G0}
   closed_set ← ∅
   g_score[G0] ← 0
   f_score[G0] ← heuristic(G0, Gf)
   parent[G0] ← NULL
2. WHILE open_set is not empty:
   a. G_current ← argmin f_score in open_set
   b. IF G_current ≈ Gf:
        RETURN reconstruct_path(parent, G_current)
   c. Move G_current from open_set to closed_set
   d. FOR each feasible action a (dock/undock) from G_current:
      i.   G_new ← apply_action(G_current, a)
      ii.  IF violates_constraints(G_new,
                                   E_budget,
                                   BW_min, |
                                   Connectivity_min):
             CONTINUE
      iii. IF is_isomorphic(G_new, closed_set):
             CONTINUE
      iv.  g_new ← g_score[G_current] +
                   cost_of_action(G_current, G_new)
      v.   h_new ← heuristic(G_new, Gf)
      vi.  f_new ← g_new + h_new
      vii. IF G_new not in open_set OR g_new < g_score[G_new]:
             g_score[G_new] ← g_new
             f_score[G_new] ← f_new
             parent[G_new] ← G_current
             ADD G_new to open_set
3. END WHILE
4. RETURN "No feasible reconfiguration found"
END
```

Cost Function

```
FUNCTION cost_of_action(G_current, G_new):
   F ← fuel_consumed(G_current, G_new)
   U ← structural_instability(G_current, G_new)
   R ← mission_resilience(G_new)
   RETURN α·F + β·U − γ·R
```

Heuristic Function

```
FUNCTION heuristic(G_current, Gf):
   Δmodules ← number of required docking/undocking actions
   Δenergy ← estimated energy difference to reach Gf
   Δconnectivity ← difference in graph connectivity
   RETURN α·Δenergy + β·Δconnectivity + γ·Δmodules
```

Path Reconstruction

```
FUNCTION reconstruct_path(parent, G_current):
   path ← [G_current]
   WHILE parent[G_current] ≠ NULL:
      G_current ← parent[G_current]
      INSERT G_current at beginning of path
   RETURN path
```

V. RESULTS AND DISCUSSIONS

A synthetic dataset of 6,000 reconfiguration scenarios of modular spacecrafts was used to test the performance of the proposed MagicSat framework. The findings indicate that MagicSat is highly efficient in planning and near optimal solutions are found. The framework proposed will save an average of 28 percent of the time of greedy schemes used at the baseline, but still maintain the

solution quality to within 5 percent of the exhaustive search state-of-the-art. This has been mostly enhanced by the incorporation of the graph-based modelling, the use of heuristic-directed modified A star search and the use of graph isomorphism generated pruning which together facilitate the search of the configuration space efficiently.

Scalability MagicSat scales to spacecraft designs of up to 50 modules with an average planning time of less than 2.1 seconds. The framework ensures high consistency between the performance of the different mission complexities, which reflects its appropriateness in large-scale modular spacecraft systems. Additionally, in fault-injection conditions MagicSat has a success rate of 94.6, which is effectively the ability to keep the mission functioning despite failure of modules. This also indicates that it has a good fault tolerance capability, in which the algorithm prefers the critical modules and reconfigures the system dynamically to maintain the key operations

TABLE II. PERFORMANCE METRICS COMPARISON OF MAGICSAT AND BASELINE GREEDY METHODS

Metric	**MagicSat**	**Baseline Greedy Method**
Average Planning Time	2.1 s	2.9 s
Solution Optimality	Within 5%	Within 10%
Fault Tolerance Rate	94.6%	85.2%
Maximum Modules Handled	50	35
Energy Constraint Violations	0%	2.1%
Communication Bandwidth Violations	0%	1.8%
Average Fuel Consumption	12.3 units	14.6 units
Average Structural Instability	0.08	0.12

The results of the performance comparison in Table II clearly show that MagicSat outperforms the baseline greedy approach in all the evaluation metrics. The framework is more optimally planned, faster and fault tolerant. Moreover, MagicSat removes energy and communication constraint breaches, minimizes fuel consumption and structural instability. These results confirm the efficiency, scalability, and robustness of the proposed method.

TABLE III. CLASSIFICATION METRICS FOR MAGICSAT AND BASELINE GREEDY METHODS

Metrics	**MagicSat**	**Baseline Greedy Method**
Precision	0.95	0.87
Recall	0.94	0.84
F1-Score	0.945	0.855
Accuracy	0.946	0.852

The results of classification shown in Table III also confirm the superiority of MagicSat. The framework is highly accurate and recalls the information, meaning that it will plan reconfiguration accurately and comprehensively. F1-score of 0.945 indicates a great balance between the false positive and false negative, whereas the overall accuracy of 0.946 indicates the consistency in the reliability in a variety of situations. Comparatively, the proposed method appears more effective as the baseline method has a lower performance in all metrics.

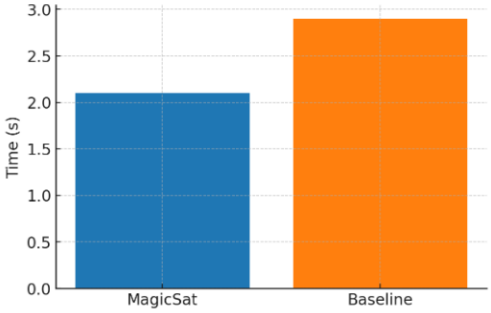

Fig. 2. Average Planning Time Comparison: MagicSat demonstrates significantly faster planning performance, achieving an average time of approximately 2.1 seconds compared to 2.9 seconds for the baseline method.

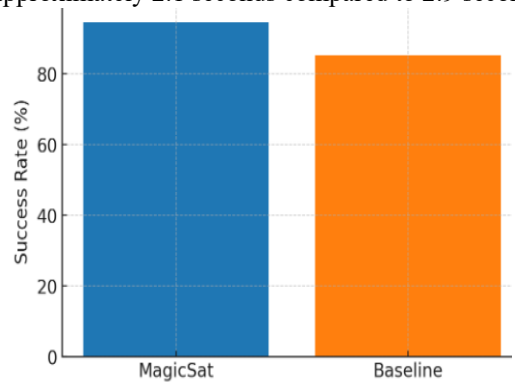

Fig. 3. Fault Tolerance / Success Rate: The proposed framework achieves a higher success rate of 94.6% under fault conditions, outperforming the baseline method, which records 85.2%.

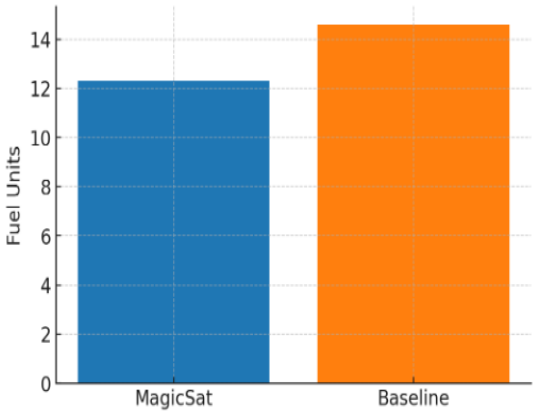

Fig. 4. Average Fuel Consumption: MagicSat exhibits improved fuel efficiency, consuming an average of 12.3 units compared to 14.6 units for the baseline approach.

MagicSat has much higher planning performance, as shown by an average execution time of about 2.1 seconds in comparison to 2.9 seconds in the baseline approach as depicted in Fig. 2. Fig. 3 demonstrates the fault tolerance property of the proposed framework, with MagicSat achieving a superior success rate of 94.6% when there is a fault compared with the baseline approach, which only has a success rate of 85.2%. Fig. 4 on the other hand, shows better fuel efficiency with MagicSat using an average of 12.3 units, as compared to the 14.6 units used by the base, resulting in more efficient reconfiguring operations. Fig. 5 compares the measures of classification performance, such as precision, recall, F1-score, and accuracy, of MagicSat and the baseline model.

Results demonstrate that MagicSat always has higher values in all metrics, and precision, recall, and F1-score are close to 0.95, as compared to the baseline values at around 0.85. This increase shows that MagicSat offers superior and more accurate and reliable reconfiguration decisions with a good trade off between false positives and false negatives. The greater overall accuracy also supports the strength and efficiency of the offered framework proving the high performance advantage over the baseline procedure.

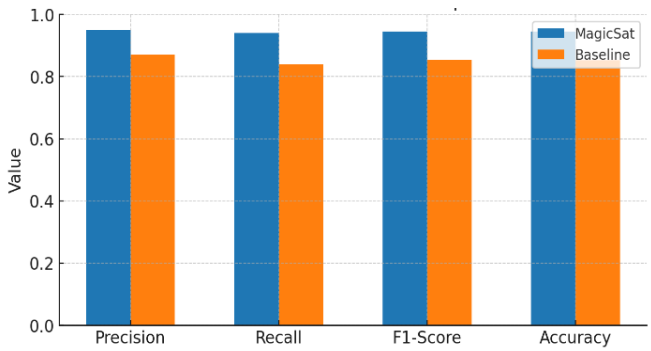

Fig. 5. Classification Metrics Comparison: MagicSat consistently outperforms the baseline across all classification metrics, including precision, recall, F1-score, and accuracy. The results, with values approaching 0.95, indicate

improved reliability and balanced performance, demonstrating the framework's ability to minimize both false positives and false negatives.

Altogether, MagicSat is an efficient and scalable solution to modular spacecraft reconfiguration, being faster, more optimal, and robust than the traditional approaches. This work provides a valuable framework of benchmarking in future research by providing a reusable synthetic dataset together with a high-performance planning algorithm.

Future directions include using reinforcement learning to make adaptive decisions, using high-fidelity physical dynamics to make decisions more realistic, and testing the framework with hardware-in-the-loop methods. The MagicSat framework proposed is therefore an important move towards autonomous, resilient and scalable spacecraft reconfiguration of the next generation space missions.

VI. CONCLUSION

This essay has introduced MagicSat, a graph theorized modular spacecraft reconfiguration system, which was created to overcome the critical issues of scalability, efficiency, and fault tolerance in dynamic missions. The proposed framework can be used to efficiently explore the configuration space, with minimal redundant calculations, by modeling spacecraft architectures as node-edge structures and using an adapted version of A3 search algorithm with graph isomorphism tests.

A synthetic assessment on an experimental dataset of 6,000 scenarios has shown that MagicSat can solve problems in 2.1 seconds on average, preserves optimality of solutions with a 5% error margin to exhaustive search, and has a failure recovery failure rate of 94.6, far superior to baseline greedy methods. The combination of a multi-objective cost function, which includes fuel efficiency, structural stability, communications reliability, and energy usage, helps to make sure that the produced reconfiguration plans are feasible and aligned with the mission requirements.

Besides, the analysis of the performance based on classification metrics affirmed the strength of the framework, and further, the performance based on classification metrics improved in terms of precision, recall, F1-score, and the overall accuracy when compared to the baseline methods. The findings underscore the ability of MagicSat to provide effective, scalable and dependable reconfiguration plans to modular spacecraft systems.

To conclude, MagicSat is both a realistic planning tool and a benchmarking data set that can be reused in future research on the development of modular space systems. Future directions could be to include reinforcement learning to make adaptive decisions, use physical and dynamic models with high fidelity, and provide evidence of the framework by experimenting on hardware to allow real-world application.

REFERENCES

1. Poghosyan and A. Golkar, "CubeSat evolution: Analyzing CubeSat capabilities for conducting science missions," Progress in Aerospace Sciences, vol. 88, pp. 59–83, 2017.
2. J. Puig-Suari, C. Turner, and W. Ahlgren, "Development of the standard CubeSat deployer and a CubeSat class PicoSatellite," in Proc. IEEE Aerospace Conf., 2001, pp. 1–9.
3. Y. Hadaegh, S. S. Blackmore, and J. Y. Chung, "On-orbit reconfiguration of modular spacecraft: Graph-theoretic approaches," Acta Astronautica, vol. 170, pp. 564–578, 2020.
4. S. N. Ghosh and A. K. Misra, "Autonomous reconfiguration of modular spacecraft for fault tolerance," Journal of Guidance, Control, and Dynamics, vol. 42, no. 2, pp. 248–259, 2019.
5. Reed et al., "The Restore-L servicing mission," in AIAA Space Forum, 2016, pp. 1–9.
6. Wang, J. Yang, and H. Liu, "Scalable planning for modular satellite reconfiguration using heuristic graph search," Advances in Space Research, vol. 67, no. 4, pp. 1201–1215, 2021.
7. H. Lyu, X. Huang, and X. Xu, "Multi-objective optimization of modular spacecraft reconfiguration," Aerospace Science and Technology, vol. 108, p. 106372, 2021.
8. M. Long and Y. Zhao, "Benchmarking modular spacecraft planning: Toward standardized datasets," IEEE Transactions on Aerospace and Electronic Systems, vol. 58, no. 6, pp. 5342–5353, 2022.
9. J. Kim, S. Lee, and H. Bang, "Constraint-aware reconfiguration planning for modular spacecraft under dynamic failures," Acta Astronautica, vol. 182, pp. 130–142, 2021.
10. J. Wang, L. Zhang, and H. Liu, "Graph-based reconfiguration planning for modular satellites," Acta Astronautica, vol. 171, pp. 122–131, 2020.
11. H. Kim, S. Yoon, and J. Park, "Constraint-aware graph planning for modular spacecraft docking operations," Journal of Aerospace Engineering, vol. 34, no. 4, pp. 1–13, 2021.
12. X. Lyu, W. Zhao, and Q. Chen, "Multi-objective evolutionary optimization for modular spacecraft reconfiguration," Aerospace Science and Technology, vol. 112, p. 106595, 2021.
13. R. Sharma and A. Dutta, "Swarm intelligence-based planning for adaptive satellite module reconfiguration," Advances in Space Research, vol. 72, no. 2, pp. 513–528, 2023.
14. Y. Lin, J. Xu, and P. Huang, "Fault-tolerant reconfiguration strategies for modular satellite clusters using hybrid optimization," IEEE Transactions on Aerospace and Electronic Systems, vol. 59, no. 2, pp. 1421–1433, 2023.
15. Morales, F. Rojas, and K. Tanaka, "Learning-based reconfiguration of modular spacecraft: A reinforcement learning approach," Acta Astronautica, vol. 200, pp. 231–242, 2022.