Instructor: Yanning Shen Winter Quarter 2025

EECS 298 Final Project Report

Students: Riwa Karam, Diana Morales, & Alex Nguyen

1 Introduction

Heterogeneous multi-robot systems are typically utilized, over their homogeneous counterparts, due to their ability to accomplish challenging missions in a coordinated and distributed fashion [1]. For instance, heterogeneous robots have been deployed for applications ranging from environmental monitoring to search and rescue operations [2, 3].

The standard approach to heterogeneous multi-robot teams is to first assign task(s) to each robot based on their capabilities [4] and then have them execute appropriate coordination strategies to achieve their respective tasks [5]. Another approach is to allow the team of heterogeneous robots to work together (i.e., collaborate) such that they can acquire new functionalities that are not possessed by any single robot operating in isolation [6, 7]. However, the challenge lies in having these robots collaborate effectively such that they can accomplish the task at hand better than if the standard approach (i.e., task assignment and then coordinated control) were adopted [8].

This research project seeks to address this challenge by exploring advanced graph-based learning techniques in network science, such as graph neural networks (GNNs) [9, 10], and/or optimization-based strategies [11] to enhance the performance and adaptability of heterogeneous multi-robot systems functioning in complex and, possibly even dynamic, environments. Particularly focusing on problems related to environmental monitoring [2] and coverage control [12], such as for wildfire tracking or exploration of unknown environments.

2 Related Works

In this section we discuss research that collectively explores graph-based learning, task assignment (both with and without temporal constraints), sensor fusion, and coverage control in the context of heterogeneous multi-robot systems.

2.1 Graph Neural Networks

Graph neural networks (GNNs) are designed to process and learn from graph-structured data [9, 10]. GNNs work by aggregating information from neighboring nodes to update each node's representation (i.e., set of features), allowing them to capture patterns and dependencies in graphs; e.g., perform tasks like node classification, link prediction, and graph classification. A dynamic cross-attention mechanism enables robots to gather information from their neighbors, improving spatial awareness. In particular, [13] focuses on obtaining spatial relationships, such as depth estimation, within a multi-robot network using this mechanism. This work lays the foundation for a generalized multi-robot coordination framework, with future directions exploring sensor fusion integration into swarm control and planning. [14] builds on this by investigating spectral graph convolutions and GNNs for decentralized multi-agent exploration and navigation in indoor environments. By leveraging spatio-temporal graphs, the study models unstructured data for robot exploration to improve spatial representation, addressing the challenge of processing real-world complexity. This could lead

to action recognition and forecasting, enabling more efficient exploration strategies. Considering a graph where robots are represented as nodes, edges corresponding to possible movements, this allows for flexible and adaptive coordination. Another study on graph-based modeling develops an evolutionary graph neural network for traffic prediction [15], where the semantic adjacency matrix evolves throughout training. This allows the model to adaptively capture traffic complexity and variability, providing insights into dynamic system behavior—concepts that could translate to multi-robot navigation and coordination.

2.2 Task Assignment

Task assignment concerns itself with how to best categorize into subgroups based on respective capabilities [16, 17]. Typically, the problem of instantaneous task assignment (without temporal constraints) has been tackled with either market-based [18] or optimization-based [19] approaches, whereas time-extended task assignment (with temporal constraints; i.e., task-scheduling) has been tackled with temporal planning-based [20] or reinforcement learning-based [21] approaches. However, more recently, there has been work that attempts to solve such problems using a GNN. For instance, [22] introduces RoboGNN – a homogenous robot scheduling framework that applies graph attention learning and imitation learning to manage tasks in environments with spatial and temporal constraints. Unlike application-specific solutions, RoboGNN provides a generalized scheduling approach. However, they have not yet explored heterogeneous robot teams, transfer learning, and alternative objective functions to ensure adaptability and applicability in real-world scenarios. [23] proposes a novel Decentralized Graph Neural Network approach for multi-robot Goal Assignment (DGNN-GA), which leverages a heterogeneous graph representation to model the inter-robot communication topology and assign relations between goals and robots in a decentralized fashion. Additionally, the authors investigate the trade-off between performance and the communication burden of DGNN-GA as a function of the number of communication exchanges.

2.3 Robot Perception

Robots must be able to fuse relevant information gathered by the onboard sensor(s) to perceive the environment with high precision. One study for environmental monitoring highlights the need for sensor fusion in methane emission detection, where laser spectrometers, light detection and ranging (LiDAR), and sun-glint geometry could enhance offshore monitoring, improve cost efficiency, and characterize methane signatures across industries to assess climate impact [24]. A study looking at system monitoring combines sensors such as inertial measurement units (IMUs), rotary encoders, and ultra-wideband beacons with an extended Kalman filter to improve swarm robotics formation, showing how multi-sensor data fusion strengthens localization and control [25]. Another work integrates real-time kinematic, LiDAR, and IMU within a simultaneous localization and mapping (SLAM) framework for a single unmanned surface vehicle. This approach addresses traditional SLAM challenges, demonstrating how sensor fusion can provide a more accurate environmental representation and improve navigation in uncertain conditions [26].

2.4 Coverage Control

Coverage control is a technique used to effectively monitor or cover a specific area by a set of agents (e.g., robots) [12]. Specifically, the goal is to optimize the placement and movement of these agents such that the collective sensing capability over a given area is maximized, i.e., minimize gaps in coverage. In recent years, learning-based approaches to coverage control have been gaining

traction. For example, [27] explores graph-based approaches for this type of graph to coordination and decision-making, examining dynamic unidirectional graphs where the robot movements define the evolving topology. A key challenge in their work is deploying Graph Neural Networks (GNNs) in real-world distributed teams, where asynchronous and intermittent communication can impact performance. [28] develops a decentralized coverage control approach for a multi-robot team with limited sensing capabilities. In particular, this paper uses a GNN to realize the decentralized control policy – leveraging non-local information from multi-hop neighbors for decision-making. The GNN framework allows the robots to share information about their positions, states, and the environment, which helps them make better decisions about where to move in order to achieve complete coverage. This achieves a higher quality of coverage when compared to classical approaches that only use local information.

3 Problem Statement

Consider a team of N heterogeneous robots, which are deployed in a known region of interest, $\mathcal{D}$, for monitoring purposes. Each robot is assumed to possess a sensor suite, enabling it to gather information about the environment, and be able to move, according to its dynamics $\dot{x} = f(x, u)$. Our project explores how robots with different types of sensing and mobility characteristics can collaborate to "paint a better picture of the world" (i.e., enhance the overall team's perception of the environment).

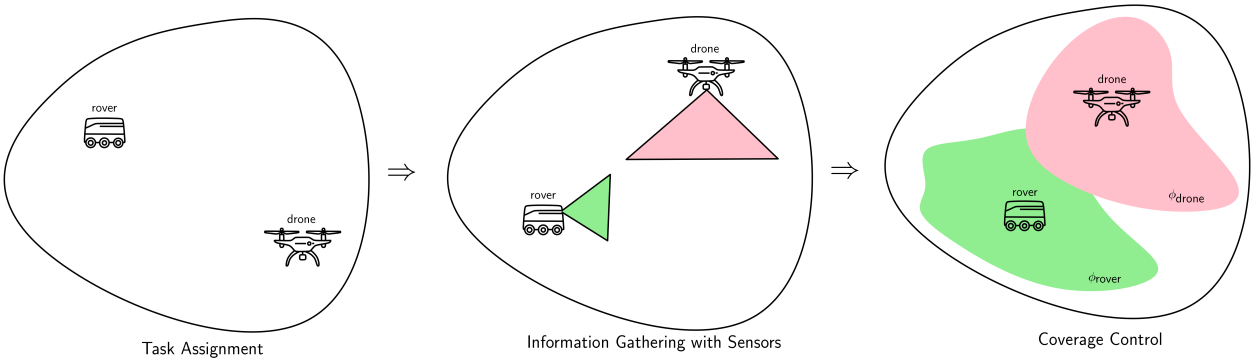

Figure 1: Example scenario: A drone and rover tasked with monitoring a region of interest. The robots are first assigned tasks based on their respective capabilities, then they will fuse the information gathered by their onboard sensor suite and execute a coverage control strategy.

This problem can be broken up into a few steps, as illustrated in Figure 1:

- Task assignment – assign the most suitable robots to regions of interest in the environment, based on their respective capabilities, over certain time windows;
- Information gathering with sensors – each robot gathers information while moving throughout the environment, and then robots will perform sensor fusion such that they can perceive the environment as a distribution (i.e., estimate a density function);
- Coverage control – robots execute coordination strategies, based on the estimated density function, such that they can maximize their respective quality of coverage.

Note. Our final project will focus on task-scheduling in a heterogeneous multi-robot team, while the information gathering with sensors and coverage control steps are left as future work.

4 Heterogeneous Graph Attention Networks for Task Scheduling

This section first provides an overview of graph attention networks. Next, we introduce the ScheduleNet framework – a heterogeneous graph attention-based model that solves the multi-robot task scheduling problem – and our contribution to it.

4.1 Graph Attention Network

Graph attention networks (GATs) are an extension of graph convolutional networks (GCNs) [29], which update the representation of a target node via a weighted average of its one-hop neighbors' features as

$$h_{N(j)} = \sum_{i \in N_i} \sqrt{\frac{1}{d_j}} \sqrt{\frac{1}{d_i}} h_i = w_{i,j} h_i, \tag{1}$$

where d_i is the degree of the source nodes, d_j is the degree of the target node, and N_i is the neighborhood set of node i.

Unlike GCNs, however, GATs replace the weight $w_{i,j}$ with an attention mechanism e_{ij}, which considers both the source and target nodes, allowing the weights to be determined by more than just the number of neighbors (i.e., local graph structure information) [30]. To determine the mechanism of attention we make use of a learnable linear projection matrix, $\mathbf{W}$, which is a weight matrix allowing us to focus on specific features and $\vec{\mathbf{a}}$ is a learnable weight vector of parameters, which is a shared attentional mechanism used to compute attention coefficients

$$e_{ij} = \sigma(\vec{\mathbf{a}}^{\mathsf{T}}[\mathbf{W}\vec{h_i} \| \mathbf{W}\vec{h_j}]), \tag{2}$$

where $\vec{h_i}$ and $\vec{h_j}$ are the raw feature message vectors of node i and j, respectively, and $\|$ represents the concatenation operation. Furthermore, by applying a nonlinear activation function, σ, such as LeakyReLU, we can compute the normalized masked attention, given as

$$\alpha_{ij} = \text{softmax}(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in N_i} \exp(e_{ik})}. \tag{3}$$

The normalized attention coefficients, α_{ij}, are then used to compute the updated representation of features for each node, given as

$$\vec{h_i'} = \sigma\left(\sum_{j \in N_i} \alpha_{ij} \mathbf{W} \vec{h_j}\right), \tag{4}$$

Since $\mathbf{W}$ and $\vec{\mathbf{a}}$ are learnable parameters, there can be multi-head attention, i.e, there exist more than one attention mechanism.

If the multi-head attention happens in an intermediate layer, then the nonlinearity activation function is applied and then we concatenate to get the updated representation of the target node

$$\vec{h_i'} = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j \in N_i} \alpha_{ij}^k \mathbf{W}^k \vec{h_j}\right). \tag{5}$$

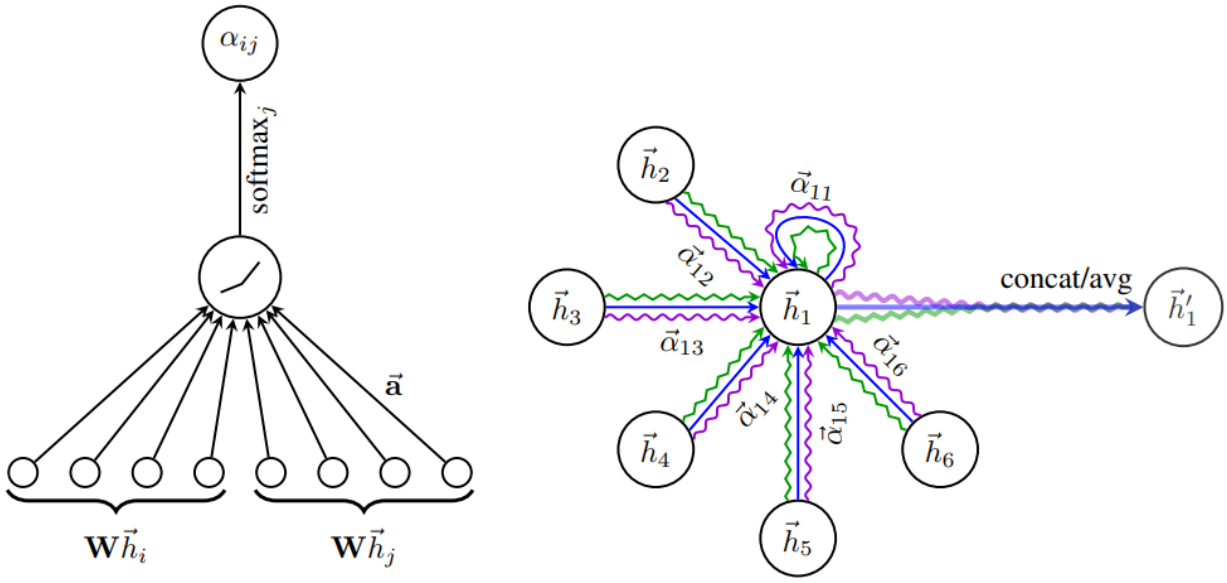

Figure 2: Left: The attention mechanism which applies LeakyReLu activation. Right: Multi-head attention where each head are either concatenated or averaged to obtain $\vec{h}'_1$ [30].

If the multi-head attention happens at the final layer, then averaging is performed and the nonlinearity activation function is then applied to get the updated representation of the target node

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right) \tag{6}$$

The aggregation process of multi-head graph attention is portrayed in Figure 2, which is taken from [30].

4.2 ScheduleNet

To tackle the problem of task-scheduling, we leverage the ScheduleNet framework [31] – a novel heterogeneous graph attention network model that can learn heuristics for per-edge-type message passing and node-type feature reduction mechanisms to solve such problems (Figure 3). ScheduleNet extends the simple temporal network formulation [32], which allows one to encode temporal constraints into a heterogeneous graph, by introducing robot and resource location nodes to capture various constraints with a graph structure. Further, ScheduleNet generalizes and scales well to large, unseen problems when trained on small-scale problem settings.
A graph learning-based architecture is considered because solving the scheduling problem as a mixed-integer linear program (MILP) with spatial and temporal constraints is typically a NP-hard problem [33], meaning it does not scale well to large problem settings. For instance, [34] introduces

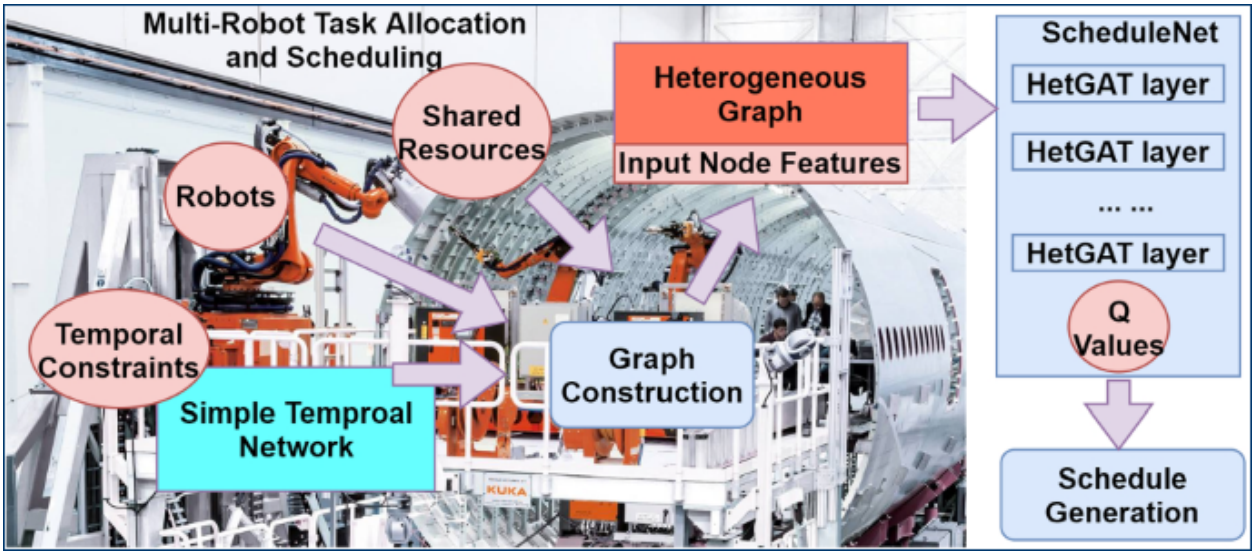

Figure 3: Overview of the ScheduleNet Framework [31]

a MILP for the task-scheduling problem, given as

$$
\begin{aligned}
&\min\ z && (7)\\
&\text{s.t.}\quad \sum_{r\in R} A_{r,i} = 1.\ \forall i \in \tau\\
&\quad f_i - s_i = dur_i,\ \forall i \in \tau\\
&\quad f_i - s_0 \le d_i,\ \forall d_i\{d\}\\
&\quad s_i - f_j \ge w_{i,j},\ \forall w_{i,j} \in \{w\}\\
&\quad (s_j - f_i)A_{r,i}A_{r,j}X_{i,j} \ge 0,\ \forall i,j \in \tau,\ \forall r \in R\\
&\quad (s_i - f_j)A_{r,i}A_{r,j}(1 - X_{i,j}) \ge 0,\ \forall i,j \in \tau,\ \forall r \in R\\
&\quad (s_j - f_i)X_{i,j} \ge 0,\ \forall (i,j) \in L_{same}\\
&\quad (s_i - f_j)(1 - X_{i,j}) \ge 0,\ \forall (i,j) \in L_{same}
\end{aligned}
$$

where, at a high-level, this optimization problem aims to minimize the makespan (z; i.e., the total time required to complete all assigned tasks) subject to temporal constraints – such as deadline and relative relationship (i.e., correlation) between multiple tasks – and resource constraints – such as tasks needing to be performed at specific locations. Hence, to circumvent the need to solve (7), the ScheduleNet framework was proposed for task-scheduling.

4.3 Contribution to ScheduleNet

Our project investigated how the makespan and number of decisions changed when the layers of the heterogeneous graph were merged with different aggregation configurations. In particular, we consider the following methods:

- 'cat', 'cat', 'cat', 'avg' (ScheduleNet default)
- 'cat', 'cat', 'avg', 'avg' (half averaging)

- ‘cat’, ‘avg’, ‘avg’, ‘avg’ (3/4 averaging)
- ‘avg’, ‘avg’, ‘avg’, ‘avg’ (all averaging)

where the ‘cat’ and ‘avg’ operation for layers are represented as (5) and (6), respectively. For example, we can define the layers of the heterogeneous graph for the ScheduleNet default aggregation configuration as

$$\vec{h}_i^{(1)} = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in N_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_i^{(0)}\right), \tag{8}$$

$$\vec{h}_i^{(2)} = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in N_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_i^{(1)}\right), \tag{9}$$

$$\vec{h}_i^{(3)} = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in N_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j^{(2)}\right), \tag{10}$$

$$\vec{h}_i^{(4)} = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j^{(3)}\right), \tag{11}$$

where $\vec{h}_i^{(0)} = \vec{h}_{i,0}$ is the initial (i.e., layer 0) embedding, typically set to be equal to node features, and $\vec{h}_i^{(\ell)}$ is the representation of features for layer ℓ. Note that the other aggregation configurations can similarly be defined.

5 Findings

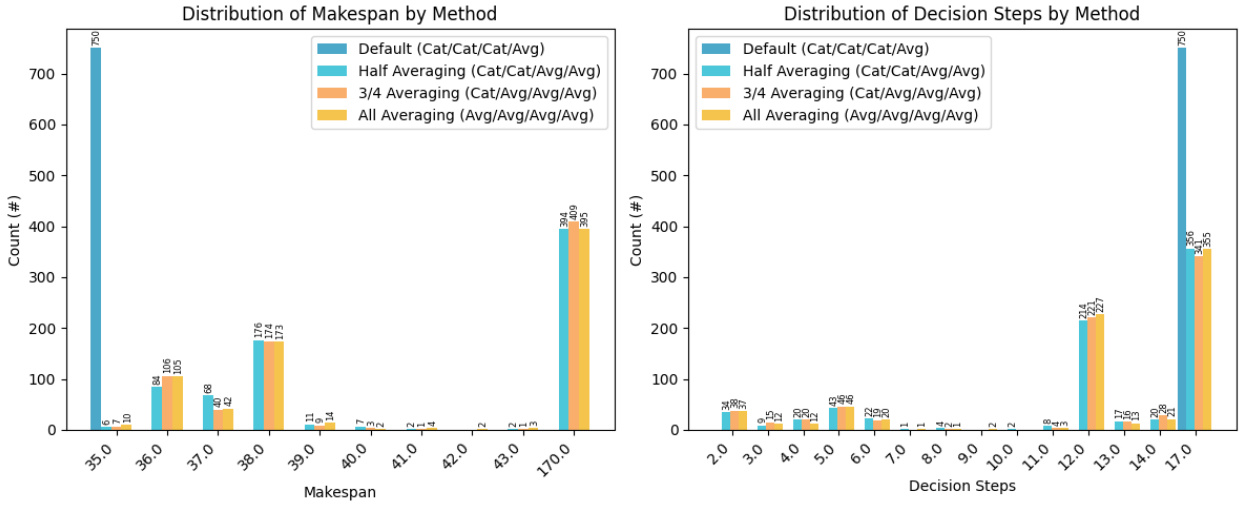

Figure 4: Distribution of the makespan and number of decision steps for solving the task-scheduling problem with different configurations of aggregations (i.e., ‘cat’ and ‘avg’) over 750 realizations.

The analysis is based on a scheduling scenario involving 17 distinct tasks and 5 available robots. Each task has a set of durations corresponding to each robot – represented in a 17×5 matrix –

reflecting the time each robot takes to process a given task. Additionally, every task is assigned a two-dimensional location, which can be used to infer proximity and potential sequencing constraints. Temporal constraints are imposed through deadlines and wait constraints: for instance, two tasks (task 2 and task 15) must be completed by time 24 and 26, respectively, while specific pairs of tasks have mandatory wait periods between them (e.g., a wait of 3 time units between tasks 2 and 12). Together, these constraints define a complex scheduling environment where the goal is to assign tasks to robots in a manner that minimizes the overall makespan while respecting the processing times, spatial considerations, deadlines, and inter-task wait times.

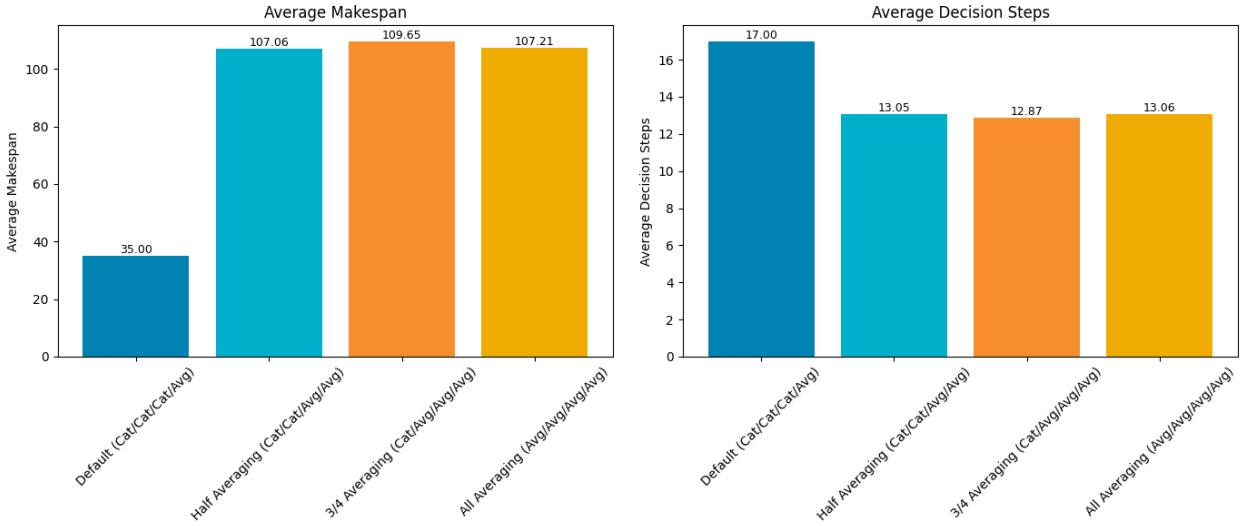

Figure 5: Bar plot of the average makespan and average number of decision steps for solving the task-scheduling problem with different configurations of aggregations (i.e., 'cat' and 'avg') over 750 realizations.

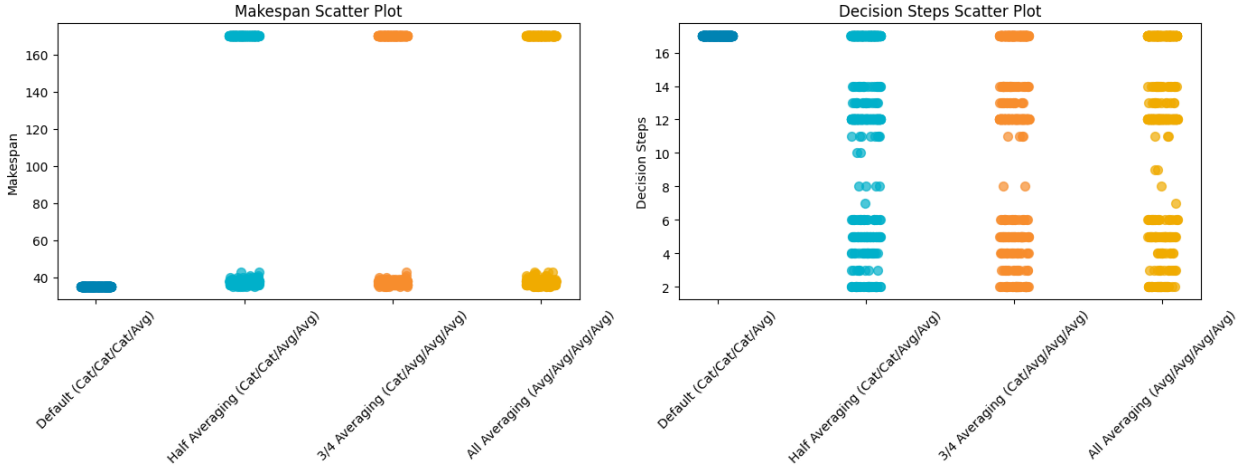

Figure 6: Scatter plot of the makespan and average number of decision steps for solving the task-scheduling problem with different configurations of aggregations (i.e., 'cat' and 'avg') over 750 realizations.

The results highlighted in Figures 4, 5, and 6, illustrate not only the clear advantage of a pretrained

network (Default) in achieving consistently lower makespan but also highlight the potential benefits for the three untrained methods. Notably, the untrained aggregations (Half, 3/4, All) exhibit fewer decision steps, indicating a more "direct" scheduling approach that —if trained— could leverage this lower step count to converge on a more efficient final schedule. The fact that their makespans jump from around 40 to over 170, as shown in Figure 6, with no intermediate values, suggests they are not partially converging toward an optimal schedule; rather, they either stumble into a near-optimal result or remain at a much higher makespan. By introducing training for these aggregations, it is plausible that they could refine their decision-making process, retaining their lower step count while closing the gap in schedule quality. In short, the analysis made by comparing different configurations of aggregations underscores the value of training these models—both to capitalize on their lower decision-step tendencies and to fill the "missing middle" in makespan outcomes, ultimately producing more consistent and efficient scheduling solutions.
Note. We made the necessary changes to the code provided in [35] (Github repository) to test the performance of the different aggregation configurations.

6 Conclusion and Future Works

In conclusion, our analysis focused on evaluating the impact of different aggregation strategies within the ScheduleNet framework on multi-robot scheduling performance. We tested and compared four aggregation configurations using a fixed scheduling scenario; the results revealed that the pretrained default configuration consistently yields lower makespan, indicating more effective scheduling, while the untrained alternatives, despite achieving lower decision steps, resulted in a higher makespan. This gap underscores the importance of both effective aggregation and training and these findings suggest that further training of alternative aggregations could potentially bridge the performance gap observed in our experiments. Moreover, we chose to employ a GNN for task scheduling due to its inherent ability to process large, heterogeneous graphs, enabling it to generate feasible solutions even in very large-scale settings, making it a scalable algorithm for complex scheduling problems.
A natural next step is to extend ScheduleNet to consider dynamics graphs, as it currently only applies to static graphs (e.g., a fixed number of robots and tasks), with the reason for this being that robots and tasks can appear on the fly. Hence, altering the current GNN model to accommodate time-varying networks would make this multi-robot task-scheduling architecture more resilient to dynamic environments. Additionally, moving forward, we also want to continue studying the problem of collaborative perception (Figure 1) while using graph-based learning techniques, such as graph neural networks, in heterogeneous multi-robot teams. In particular, we will focus on the last two steps of our proposed framework: robot perception (i.e., information gathering and sensor fusion) and coverage control.

References

[1] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems: A survey," *ACM Comput. Surv.*, vol. 52, no. 2, pp. 1–31, Apr. 2020.

[2] M. Dunbabin and L. Marques, "Robots for environmental monitoring: Significant advancements and applications," *IEEE Robot. Autom. Mag.*, vol. 19, no. 1, pp. 24–39, Mar. 2012.

[3] J. P. Queralta, J. Taipalmaa, B. C. Pullinen, V. K. Sarker, T. N. Gia, H. Tenhunen, M. Gabbouj, J. Raitoharju, and T. Westerlund, "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision," *IEEE Access*, vol. 8, pp. 191 617–191 643, Oct. 2020.

[4] F. Shkurti, A. Xu, M. Meghjani, J. C. G. Higuera, Y. Girdhar, P. Giguère, B. B. Dey, J. Li, A. Kalmbach, C. Prahacs, K. Turgeon, I. Rekleitis, and G. Dudek, "Multi-domain monitoring of marine environments using a heterogeneous robot team," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 1747–1753.

[5] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 12, p. 399, Jan. 2013.

[6] I. D. Miller, F. Cladera, T. Smith, C. J. Taylor, and V. Kumar, "Stronger together: Air-ground robotic collaboration using semantics," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 9643–9650, Oct. 2022.

[7] S. Mayya, D. S. D'antonio, D. Saldaña, and V. Kumar, "Resilient task allocation in heterogeneous multi-robot systems," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1327–1334, Feb. 2021.

[8] J. J. Roldán-Gómez and A. Barrientos, "Special issue on multi-robot systems: Challenges, trends, and applications," *Applied Sciences*, vol. 11, no. 24, p. 11861, Dec. 2021.

[9] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI open*, vol. 1, pp. 57–81, Apr. 2021.

[10] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[11] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[12] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243–255, Feb. 2004.

[13] Y. Zhou, J. Xiao, Y. Zhou, and G. Loianno, "Multi-robot collaborative perception with graph neural networks," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2289–2296, April 2022.

[14] F. Pistilli and G. Averta, "Graph learning in robotics: A survey," *IEEE Access*, vol. 11, pp. 112 664–112 681, October 2023.

[15] W. Ma, Z. Chu, H. Chen, and M. Li, "Spatio-temporal envolutional graph neural network for traffic flow prediction in uav-based urban traffic monitoring system," *Scientific Reports*, vol. 14, p. 26800, Nov. 2024.

[16] B. P. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," *Int. J. Robot. Res.*, vol. 23, no. 9, pp. 939–954, Sept. 2004.

[17] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: A review of the state-of-the-art," *Cooperative Robots and Sensor Networks*, vol. 604, pp. 31–51, May 2015.

[18] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 912–926, Aug. 2009.

[19] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, no. 1-2, pp. 83–97, Jan. 1955.

[20] M. C. Gombolay, R. J. Wilcox, and J. A. Shah, "Fast scheduling of robot teams performing tasks with temporospatial constraints," *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 220–239, Feb. 2018.

[21] X. Zhao, Q. Zong, B. Tian, B. Zhang, and M. You, "Fast task allocation for heterogeneous unmanned aerial vehicles through reinforcement learning," *Aerospace Science and Technology*, vol. 92, pp. 588–594, Sept. 2019.

[22] Z. Wang and M. Gombolay, "Learning scheduling policies for multi-robot coordination with graph attention networks," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4509–4516, July 2020.

[23] M. Goarin and G. Loianno, "Graph neural network for decentralized multi-robot goal assignment," *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4051–4058, May 2024.

[24] S. Zhang, J. Ma, X. Zhang, and C. Guo, "Atmospheric remote sensing for anthropogenic methane emissions: Applications and research opportunities," *Science of The Total Environment*, vol. 893, p. 164701, Oct. 2023.

[25] A. V. Le, K. G. S. Apuroop, S. Konduri, H. Do, M. R. Elara, R. C. C. Xi, R. Y. W. Wen, M. B. Vu, P. V. Duc, and M. Tran, "Multirobot formation with sensor fusion-based localization in unknown environment," *Symmetry*, vol. 13, no. 10, p. 1788, Sept. 2021.

[26] H. Lu, Y. Zhang, C. Zhang, Y. Niu, Z. Wang, and H. Zhang, "A multi-sensor fusion approach for maritime autonomous surface ships berthing navigation perception," *Ocean Engineering*, vol. 316, p. 119965, Jan. 2025.

[27] E. Tolstaya, J. Paulos, V. Kumar, and A. Ribeiro, "Multi-robot coverage and exploration using spatial graph neural networks," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 8944–8950.

[28] W. Gosrich, S. Mayya, R. Li, J. Paulos, M. Yim, A. Ribeiro, and V. Kumar, "Coverage control in multi-robot systems via graph neural networks," in *Proceedings of IEEE/RSJ International Conference on Robotics and Automation*, 2022, pp. 8787–8793.

[29] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: A comprehensive review," *Comput. Soc. Netw.*, vol. 6, no. 1, pp. 1–23, Nov. 2019.

[30] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.

[31] Z. Wang, C. Liu, and M. Gombolay, "Heterogeneous graph attention networks for scalable multi-robot scheduling with temporospatial constraints," *Autonomous Robots*, vol. 46, pp. 249–268, Aug. 2021.

[32] R. Dechter, I. Meiri, and J. Pearl, "Temporal constraint networks," *Artificial Intelligence*, vol. 49, no. 1–3, pp. 61–95, May 1991.

[33] M. M. Solomon, "On the worst-case performance of some heuristics for the vehicle routing and scheduling problem with time window constraints," *Networks*, vol. 16, no. 2, pp. 161–174, 1986.

[34] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multi-robot systems," *IEEE Trans. Robot.*, vol. 33, no. 3, pp. 661–674, Jun. 2017.

[35] Z. Wang, "Mrcscheduling," 2022. [Online]. Available: https://github.com/phejohnwang/MRCScheduling/