Task Insertion and Reassignment in Networked Robots for Topological Morphing

Lantao Liu and Dylan Shell
Dept. of Computer Science and Engineering
Texas A&M University
{lantao, dshell}@cse.tamu.edu

***Abstract*— This paper considers a particular form of incremental assignment problem that can be used to achieve efficient topological morphing in a network of robots. The assignment problem arises from a task-allocation formulation in which destination locations for newly deployed robots are added as tasks to an existing allocation. We adapt the bipartite matching variant of the Hungarian algorithm—originally designed to solve the optimal assignment problem—to perform path routing by showing that there is a three-dimensional spatial interpretation of the Hungarian graph. The assignment is globally reallocated in an efficient manner when new agent-task pairs are inserted. The algorithm uses a parameter to adjust the optimization criterion: from minimizing global time (or energy, or distance), to minimizing disruption (*i.e.,* obtaining the fewest number of robot reassignments), and mixtures in-between.**

The algorithm can be directly applied to wirelessly networked robot systems in which graph edges reflect robots whose signal strength can be measured and used for gradient-based traversal from a particular node. The adjustable optimization criterion permits the degree to which potentially unreliable long traversals are favored over more costly short traversals to be tuned. Our results suggest that the algorithm works well for sparse graphs making it well-suited to networks of robots with limited communication.

I. Introduction

Topal et. al [1] introduced the following multi-robot morphing problem: a set of robots (at particular, known initial locations) are added to a mesh network of already deployed robots, along with a specification of a set of new desired target locations. The problem is to decide which nodes should be moved, and to where. Typically the movement that minimizes (global) summed distance while ensuring that there is a robot at each target location, would involve allocating a newly deployed robot to a new target location. However, if the robots are interchangeable, a whole chain of robots may simultanously move toward the target, with each robot taking the place of the last. Despite the total distance increasing, improvements in other aspects (*e.g.,* reduced time until deployment) can make this worthwhile. In this paper, we develop a parameterized algorithm permits one to obtain solutions that balance the minimization of summed distance against number of robots that are redeployed.

While developing this algorithm, we realized that this form of multi-robot path routing is applicable more generally. The optimization criteria do not depend on particular locations, rather known or estimated distances or traversability costs between robots can suffice. As a motivating example, consider a swarm of wirelessly networked robots whose positions are not known. Local signal strength readings can be used both to estimate costs, and actually to perform gradient-based traversal from a particular node to another.

In greater detail, the assignment problem treatment of the task is as follows: A group of robots are assigned to different task locations and each robot has reached the target place. Assume that new tasks are added and redundant robots are to be deployed to these new task locations. Robots need to be reassigned so that globally all robots can arrive at the newly assigned task locations as quickly as possible. Such a scenario is also useful in monitoring tasks. Different focuses on minimal distance versus time can be important in wireless networking when the traversal is performed by gradient seeking to a particular location: we want to minimize time while also limiting the number of slow, unreliable seeking movements (*i.e.,* limiting the number of reallocations). The model can also be extended to topological change of large scale agents. For instance, by disconnecting individual agent-task pairs from the topology and reconnect them with new task assignments, we can "morph" the topology very quickly using our algorithm. This can be well utilized in the simulation of flocking formations, or global shape morphing of particle system, or the topological variations of mobile robots in irregular confined environments, etc.

In this paper, we propose the following ideas:

- Extension of the classic Hungarian complete bigraph (bipartite graph) treatment to sparse bigraphs, along with the conditions for producing valid solutions.
- We identify that the bigraph can be treated as a two-layered 3D graph that models a path routing problem.
- We design a tunable strategy that can adjust the optimization output to trade between minimal distance/time and minimal reallocation solutions.
- Investigate cases that involve insertion of multiple redundant robots and/or tasks. (Note, however, that incremental assignment problems have been considered in the literature, *cf.* [2] — but without considering the particular routing problem instantiation.)

II. Hungarian Algorithm

The Kuhn-Munkres Hungarian algorithm [3] can efficiently solve an $n \times n$ assignment problem in $O(n^3)$ time. The algorithm based on the graph matching formulation

is presented as Algorithm II.1 on the following page. It treats the input $n \times n$ utility matrix as a weighted adjacency matrix for a complete bipartite graph G (or bigraph). In the algorithm, steps 2 to 4 describes the procedure of looking for and flipping an augmenting path. We call a single iteration of this procedure a *stage*. Note that each stage finds exactly one augmenting path and this increases with exactly one matching edge, thus the Hungarian algorithm repeats at most n stages to obtain all n matching edges, which form the optimal assignment solution. The algorithm works well on the complete $n \times n$ edged bigraph, and we show below that it also works for some non-complete bigraphs. We use the term *sparse bigraph* if a bigraph satisfies $|E| < n^2$.

Preliminary 2.1: We define x_{ij} such that:

$$x_{ij} = \begin{cases} 1 \text{ if } e(i,j) \in E, \\ 0 \text{ otherwise.} \end{cases}$$

Then for any sparse bigraph with each partition of size n, the Hungarian algorithm outputs an optimal solution if and only if there is such a set $|\{x_{ij}\}| = n$ of utility matrix elements that satisfies:

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, \cdots, n$$

and

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, \cdots, n. \tag{1}$$

Proof: Necessity follows the constraint definition of the assignment problem. Sufficiency: $|\{x_{ij}\}| = n$ implies that there must be at least n edges in the bigraph. We can prove the sufficiency by discussing two cases: $(i.)$ if $|\{x_{ij}\}| = n$ and all x_{ij} satisfy (1), it indicates each vertex in the bigraph has unit degree, thus the n edges form a set which is a perfect matching, which is the optimal solution. $(ii.)$ if $|\{x_{ij}\}| > n$ and there is a subset that satisfies (1), then the Hungarian algorithm does not output an optimal solution only because it halts without finding an augmenting path. However, this cannot happen since each vertex has at least one augmenting path in this case. ■

Two corollaries are given based on the proof of Preliminary 2.1:

Corollary 2.2: In a sparse bigraph, a Hungarian stage continues if and only if an augmenting path can be found connecting a free pairwise agent and task.

Corollary 2.3: In a sparse bigraph, newly introduced agents and tasks can be incrementally assigned if and only if one can incrementally find augmenting paths between the newly introduced agents set and task set.

III. From Assignment Problem to Morphing

Multi-robot morphing involves performing efficient spatial changes to a network of robots when new available agent-task pairs are inserted. More formally, morphing is the global reconfiguration of a multi-robot network topology, and ideally this reconfiguration is a seamless transition from the prior task assignment to a new task assignment that will redeploying the fewest agents and/or using the shortest time to complete such a transition.

Algorithm II.1 The Hungarian Algorithm

Input:

A valid $n \times n$ assignment matrix represented as the equivalent complete weighted bipartite graph $G = (X, Y, E)$, where $|X| = |Y| = n$.

Output:

A perfect matching, M.

1: Generate an initial labelling l and matching M in G_e.

2: If M perfect, terminate algorithm. Otherwise, randomly pick an exposed vertex $u \in X$. Set $S = \{u\}$, $T = \varnothing$.

3: If $N(S) = T$, update labels:
$\delta = min_{x \in S, y \in Y-T}\{l(x) + l(y) - w(x,y)\}$

$$l'(v) = \begin{cases} l(v) - \delta & \text{if } v \in S \\ l(v) + \delta & \text{if } v \in T \\ l(v) & \text{otherwise} \end{cases}$$

4: If $N(S) \neq T$, pick $y \in N(S) - T$.
(a) If y exposed, $u \to y$ is augmenting path. Augment M and go to step 2.
(b) If y matched, say to z, extend Hungarian tree: $S = S \bigcup \{z\}$, $T = T \bigcup \{y\}$, and go to step 3.

Notes:

- Equality graph $G_e = \{e(x,y) : l(x) + l(y) = w(x,y)$;
- Neighbor $N(u)$ of vertex $u \in X$: $N(u) = \{v : e(u,v) \in G_e\}$.

Fig 1 illustrates how the Hungarian bigraph for morphing results in a routing problem. In Fig 1(a), a new agent-task pair is introduced to a prior perfect matching: the bold edges are the previous matched edges, and the node a_4 (leftmost node in top partition) and t_4 (rightmost node in bottom partition) are newly inserted agent and task, respectively. One more incremental Hungarian stage will obtains a new assignment solution, as shown in Fig 1(b). Notice that the augmenting path connecting a_4 to t_4 is $a_4 \to t_1 \to a_1 \to t_2 \to a_2 \to t_4$. The nodes in each partition are shown as on a straight line, but when the target positions and robots are on a 2D plane, the bigraph can be visualized within 3D, as illustrated in Fig 1(c). (This is merely a visualization,the matching configuration still holds because the weights on the edges do not change.) The dashed graph on the top layer, and projection of the matching (as arrows) to this plane is a routing solution.

Within the context of the morphing, the 3D model in Fig 1(c) represents the spatial relationship between the agents and task locations in the following way: the nodes in bottom layer represent the task locations, and the nodes in top layer are the agent assigned or to be assigned. Before the introduction of new agent-task pairs, the positions of nodes in top layer and bottom layer are vertically aligned and are pairwise matched with bold edges, indicating that they have reached their assigned task locations and executing respective tasks. When a new agent-task pair is inserted and connected to the mesh, an addition stage of the Hungarian algorithm incrementally run will find an augmenting path

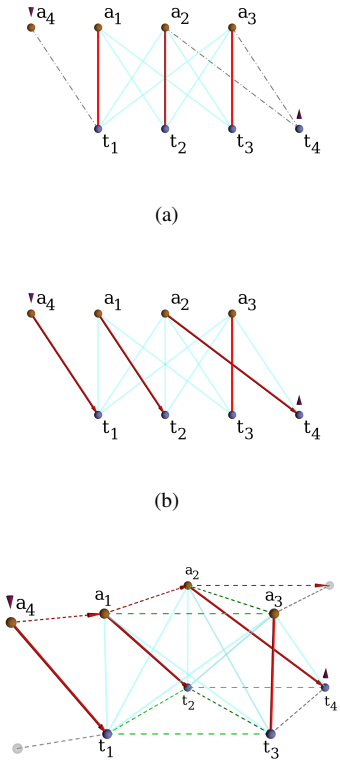


Fig. 1. Morphing

to connect this new agent-task pair, and the matched edges in the augmenting path provide exactly the global morphing solution. Take Fig 1 as an example, the augmenting path of $a_4 \rightarrow t_1 \rightarrow a_1 \rightarrow t_2 \rightarrow a_2 \rightarrow t_4$ means that agent a_4 should move to task location t_1, and a_1 move go to t_2, and a_2 to t_4. This is reflected in the planer routing solution of the top layer in Fig 1(c), as the routing path in red dashed arrow.

Theorem 3.1: So long as the 2D network for routing problem is connected, there are always morphing paths.

Proof: The graph being connected means that any node can reach some other node through a routing path. Because any routing path represents a specific morphing solution, we can always find at least one morphing paths for any agent-task pair. ■

IV. Morphing Control

We are primarily interested in two properties of the resultant morphing paths: (1.) The total number of agents to be reassigned, since usually each agent reassignment is not costless; (2.) The total time of the morphing, since and we usually expect to minimize the summed morphing time.

The Hungarian algorithm aims at maximizing the utility, while a stereotypical routing problem aims at minimizing the path length; we transform the routing minimization problem to maximization problem by negating all utility values (the Euclidean distance) in the input. In addition, when all

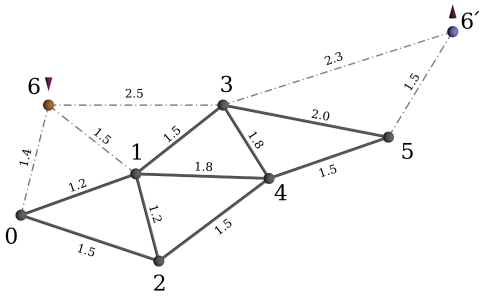


Fig. 2. Example topology of networked robots

agents have reached their task locations, to guarantee the vertical pairwise agent-task assigned (matched) relationship, we assign the utility of this vertical edge with at least the maximal utility of its all outgoing edges. Therefore, if we define utility matrix U:

$$U = \begin{pmatrix} u_{11} & u_{12} & \dots & a_{1n} \\ u_{21} & u_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ u_{n1} & u_{n2} & \dots & u_{nn} \end{pmatrix}$$

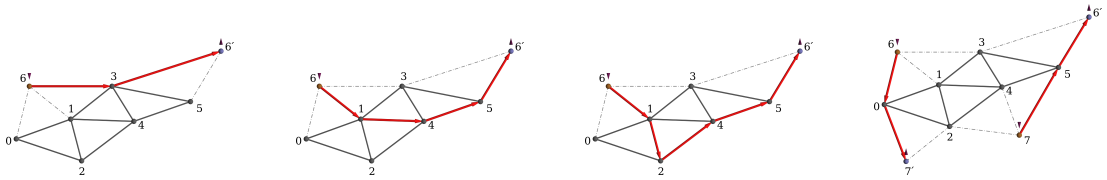
then $u_{ij} \leq 0$, and $u_{ii} \geq max\{u_{ij}\}$. If $e(i,j) \notin E$, then $u_{ij} = -\infty$, which read simply as that edge $e(i,j)$ not existing. The larger the value of u_{ii}, the more likely that edge $e(i,i)$ remain matched when incrementally run Hungarian stages, and vice versa. Let $diag(U) = diag(u_{11}, u_{22}, ..., u_{nn})$ denote the diagonal matrix of U. Then, with scaling parameter $\lambda \in [0,1]$, we get a new diagonally scaled utility matrix:

$$U^{'} = U + (\lambda - 1)diag(U) \quad (2)$$

$$= \begin{pmatrix} \lambda u_{11} & u_{12} & \dots & a_{1n} \\ u_{21} & \lambda u_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ u_{n1} & u_{n2} & \dots & \lambda u_{nn} \end{pmatrix} \quad (3)$$

In $U^{'}$, only the diagonal is scaled, *i.e.*, $diag(U^{'}) = \lambda diag(U)$. We initialize each u_{ii} with $u_{ii} = max\{u_{ij}\}$, then by tuning $\lambda \in [0,1]$, the diagonal utilities vary as $u_{ii}^{'} \in [u_{ii}, 0]$ correspondingly.

To demonstrate this, we use the example in Fig 2, in which nodes 0–5 are agents previously assigned which have reached their respective task locations. Nodes 6 and $6^{'}$ are a newly introduced agent and task respectively. Fig 3 shows different morphing paths as a function of λ, and Table I provides statistics for total path length, number of reallocated robots, average path edge length and overall morphing time. To compare the difference between the sparse and dense graph (bigraph), we also run the experiments on the dense graph constructed from Fig 2, where each node is connected to all other nodes. The statistics of routing property testing for this dense graph is shown in Table II. Both Table I and Table II are similar except when $\lambda \in [0, 0.4]$, this is because in dense graph there are edges directly connecting the newly



(a) $\lambda = .1$—.4 (b) $\lambda = .7$ (c) $\lambda = .8$—.9 (d) Insertion of multiple agent-task pairs

Fig. 3. Examples of routing results

introduced agent-task pair, thus a shorter alternative path will be found. The similarities between the sparse graph and dense graph solutions reflects the fact that edges connecting nearest neighbors, despite producing a sparse graph, permits most morphing paths to be captured. We provide the results from experiments with large scale networks in Section V. In addition, we are also interested in the condition of simultaneously inserting multiple agent-task pairs, as illustrated in Fig 3(d). Following Corollary 2.3, this can simply be solved by incrementally running multiple Hungarian stages via constructing the Hungarian tree rooted at the new inserted agent set.

TABLE I

ROUTING PROPERTY TESTING WITH SPARSE GRAPH

λ	Total Len	Realloc #	Avg. Len	Total time
0–.4	4.8	2	2.4	2.4/v
.5–.6	5.3	3	1.77	1.77/v
.7	6.3	4	1.58	1.58/v
.8–.9	7.2	5	1.44	1.44/v
1	8.3	6	1.38	1.38/v

TABLE II

ROUTING PROPERTY TESTING WITH DENSE GRAPH

λ	Total Len	Realloc #	Avg. Len	Total time
0–.3	4.3	1	4.3	4.3/v
.4	4.8	2	2.4	2.4/v
.5–.6	5.3	3	1.77	1.77/v
.7	6.3	4	1.58	1.58/v
.8–.9	7.2	5	1.44	1.44/v
1	8.3	6	1.38	1.38/v

V. EXPERIMENTS AND RESULTS

A. Large Scale Network Testing

The algorithm was tested in Matlab. We constructed sparse graphs consisting of hundreds of vertices in order to simulated a large scale networked multi-robot system. In the graph, vertices represent the stationary working robots, and edges represent traversability links. The vertices were randomly generated, each connected with its k nearest neighbors (*i.e.,* the degree of each vertex is k), where k can be controlled with specific input.

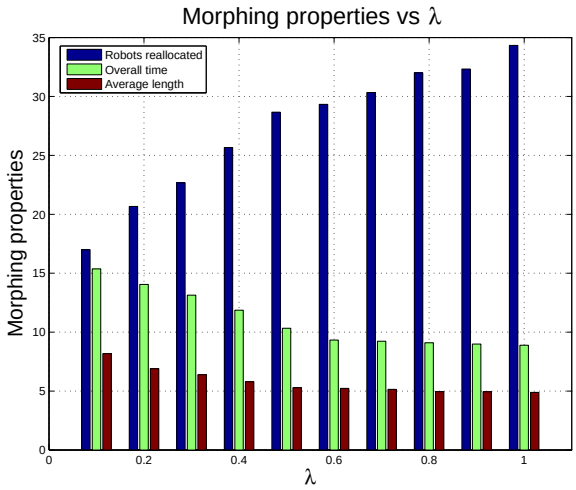
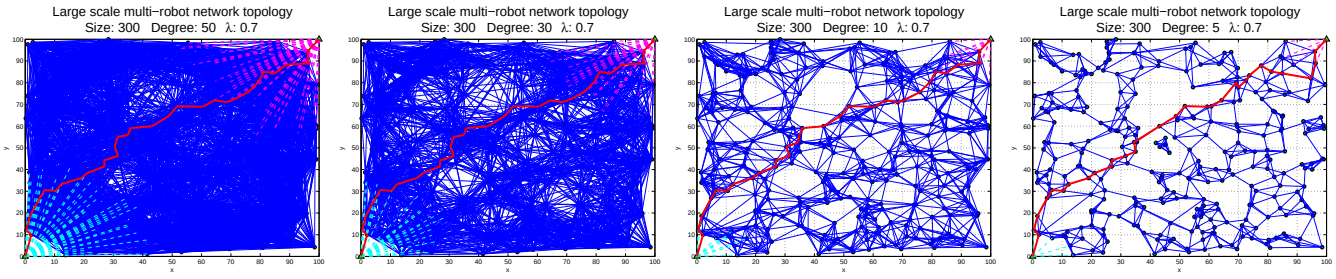


Fig. 4. Morphing properties as a function of weighting parameter λ

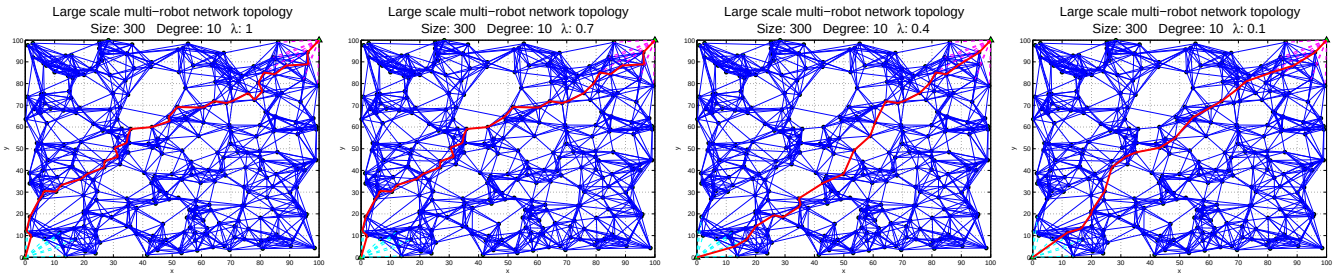
We first investigated the influence of degree on the quality of the morphing paths. We evaluated the effect of degree by using fixed number of vertices in fixed locations, and generating data after gradually reducing the degree by limiting the number of nearest neighbors. Fig 5 is an example illustrating the morphing paths in graphs of different degrees: Fig 5(a)—5(d) have the same size and same tuning parameter λ, but the degrees are controlled to be ~ 50, ~ 30, ~ 10, and ~ 5, respectively. The newly inserted agent-task pair is located in the bottom left corner (agent) and upper right corner (task). The morphing paths have few changes for graphs with degrees above 10, whereas there is an obvious change happens when the degree is 5. We ran many different experiments with graph size from tens of vertices to hundreds of vertices, and found a trend that: despite the size, the graphs with degree of above ~ 10 generate very similar morphing paths, but the consistence is not guaranteed under the degree of ~ 7. This characteristic suggests a good fit in practical applications, involving communication with few neighboring robots.

Next we examined the relationship between the properties of morphing paths and tuning parameter λ. Fig 6 shows different paths when λ decreases from 1 to 0.1, which gradually "straightens" the paths with shorter total lengths. We collected statistics from ten sets of experiments, each



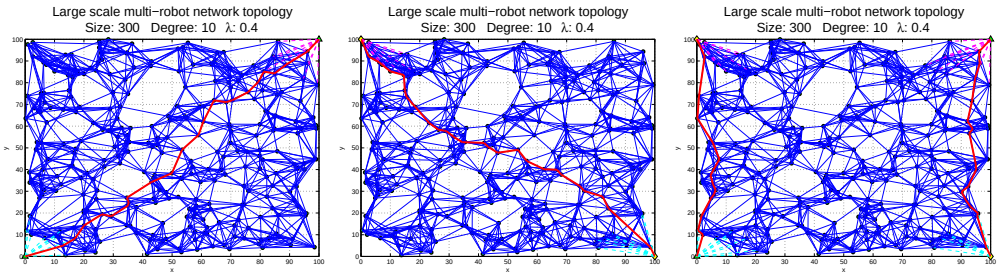
(a) (b) (c) (d)

Fig. 5. Morphing paths in large scale multi-robot topologies of varying degree.



(a) (b) (c) (d)

Fig. 6. Morphing paths in large scale multi-robot topologies of varying λ.



(a) (b) (c)

Fig. 7. Multiple optimal morphing paths

of which has a size of 300 and a degree of 10, and the mean of the results are plotted in Fig 4. We are interested in three properties: the number of robots reallocated (blue bars), the overall morphing time (green bars) and the average distance each robot moves, i.e., average edge length in the morphing path (maroon bars). From Fig 4 we observe that: as λ increases, the number of reallocated robots increases, whereas both the overall morphing time and average moving distance decrease and appear to flatten out. Moreover, the rates of either the increment or decrement are bigger when $\lambda < 0.5$ than those in the other part. This implies that the tuning step should be smaller when λ is close to 0.

We also tested the cases when multiple agent-task pairs are introduced simultaneously. Fig 7 is an example showing two agent-task pairs that are inserted at the same time. The agents (robots) are located in the bottom corners and tasks are assigned in upper corners. Fig 7(a) and Fig 7(b) describe two separate paths when only one agent-task pair is inserted, and Fig 7(c) is the case when the two pairs are added simultaneously. The new paths in Fig 7(c) have better qualities than the cases of single pairs, and this is because the morphing solutions are generated from the Hungarian algorithm, which always finds the optimal solutions globally.

B. Analysis

In this paper we consider a routing problem via a task allocation treatment, and we believe it has the following advantages compared to other routing algorithms which may also be utilized for the morphing problem:

1) Low computational complexity: each Hungarian stage requires $O(n^2)$ for dense bigraph [2], and for a sparse bigraph with edge degree of k, the computational complexity is bounded by $O(kn)$. When the problem arises from a network of robots communicating with nearby neighbors, we expect that this latter bound will be applicable.
2) Incremental paradigm: the assignment results are always saved for future usage, thus multiple and redundant agents and/or tasks can be morphed with low computational complexity. Moreover, because of the special aligned matching relation, any vertical matched pair can be easily removed from the 3D topology by disconnecting incoming and outgoing edges.
3) Tunable paths with easy scaling method: The algorithm does not modify any information of the 2D network, but achieves different featured reassignment solutions via scaling a special "virtual" edge for each node via λ, which can be easily computed.

VI. Conclusion

In this paper we propose a solution to the morphing problem in a multi-robot system. Our experience suggests that this same model can be used in route planning for wirelessly networked robot systems. Our approach transfers the Hungarian algorithm, which is designed to solve optimal assignment problem, to a path routing problem, which is implemented in a two-layered 3D bipartite graph. This method solves an incremental multi-robot task assignment problem where the tasks represent new destination locations for newly deployed robots. The algorithm globally reallocates the assignment when new agent-task pairs are inserted, with different required target optimizations.

References

[1] S. Topal, I. Erkmen, and A. M. Erkmen, "Morphing a Mobile Robot Network to Dynamic Task Changes over Time and Space," in *International Conference on Automation, Robotics and Control Systems (ARCS-09)*, Orlando, Florida, USA, July 2009, pp. 192–199.

[2] I. H. Toroslu and G. Üçoluk, "Incremental Assignment Problem," *Information Sciences*, vol. 177, no. 6, pp. 1523–1529, Mar. 2007.

[3] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1, pp. 83–97, 1955.