The University of Texas at Austin
Walker Department of Mechanical Engineering
Cockrell School of Engineering

Minimum-Time Scheduling of Autonomous Mobile Robots

Advanced Power Systems and Control Laboratory

TEXAS
The University of Texas at Austin

ABSTRACT

Manufacturing automation has been under intensive study due to increased productivity. This research effort presents an algorithm on scheduling of an AMR that traverses desired locations on a manufacturing floor. The algorithm enables the AMR to find an efficient tour within the time constraint, using which it can travel through all coordinates before returning to the starting point or a specified stop, within a stipulated time of thirty seconds on a PC.

With the number of AMRs increasing, how to optimally schedule them in a timely manner such that a large school of AMRs can finish all assigned tasks within the shortest time also presents a significant challenge. We introduce a novel two-step algorithm for fast scheduling of AMRs that perform prioritized tasks involving transportation of materials from a pick-up point to a drop-off point on the factory floor.

The ultimate goal of this research is to develop a development platform that can generate an optimal path to route the AMR in real time considering both the shortest route and the task priority.

INTRODUCTION AND PROBLEM FORMULATION

The research effort focuses on the following problems:

a) Given a large set of coordinates (~1000) on a manufacturing floor and the cost of travel, under arbitrary starting and ending points, find the most efficient schedule for an AMR, such that it travels to all desired points without repetitions. This problem can be interpreted as a variation of the Traveling Salesman Problem (TSP), mathematically defined as follows:

$$\min_{t_i \in V^N} \sum_{i=1}^{N} \left\| t_{\{i+1\}} - t_i \right\|_2$$

$$\text{subject to} \quad t_i \neq t_j, \forall j = 1, 2, \ldots, N, j \neq i$$

$$t_{\{N+1\}} = t_1$$

where $T = t_1, t_2, \cdots, t_N, t_1$ is a tour starting and ending at the same point, and V^N is the point set of size N.

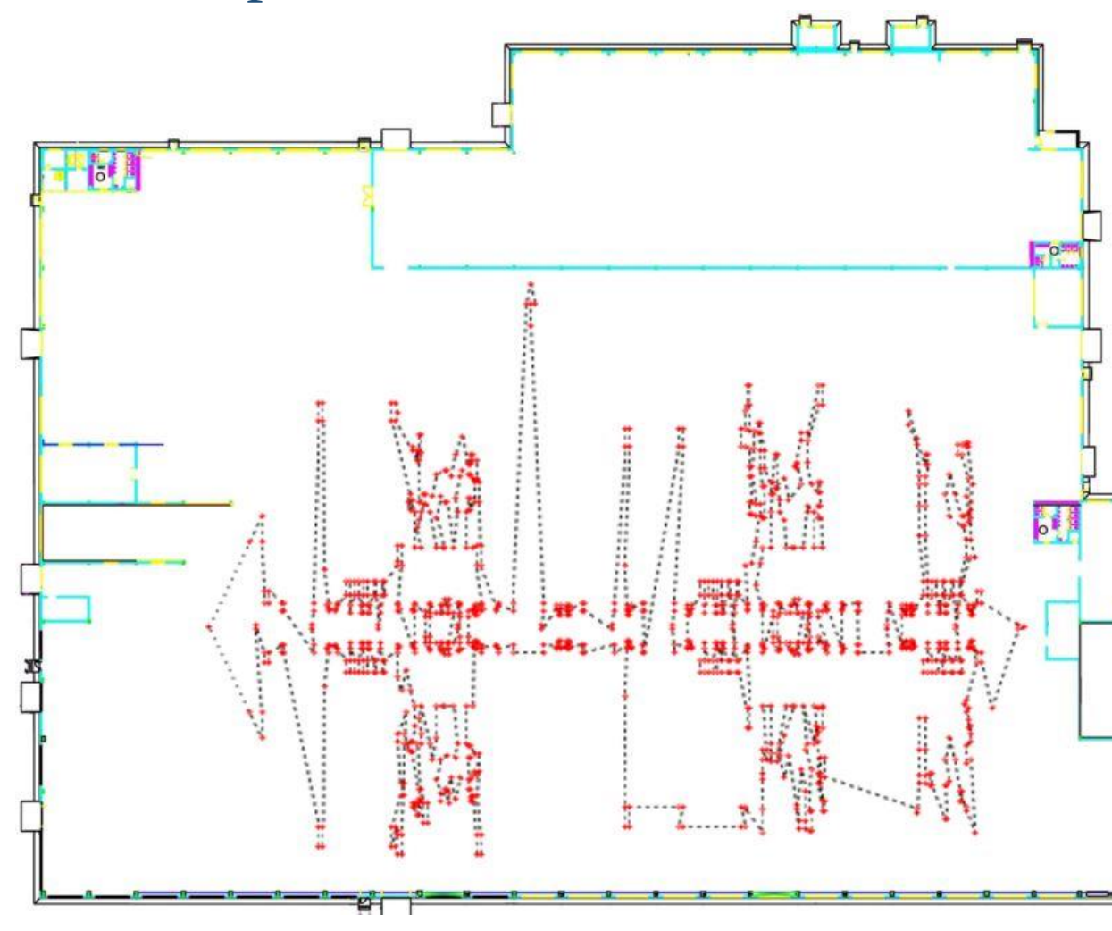

Fig. 1. Example of AGV routing for drilling on a factory floor for assembly line setup using a TSP solution

b) Scheduling of prioritized tasks for multiple AMRs within a manufacturing facility in a timely manner. Specifically, we focus on problems where 20 to 40 AMRs need to travel to about 1000 points to complete prioritized tasks. The prioritized tasks involve transporting materials from a specified pick-up location to a target drop-off location. Here, each task is associated with a start point s, an end point e, and a priority value p. Because the task is predefined on a 'point space' and the start and end points are not interchangeable, the search of an optimal scheduling is asymmetric.

c) Hardware setup for verification and validation of the designed algorithms. The prototype built for testing purposes is a 4-wheel ground vehicle designed for warehouse applications in which small cargo up to 5kg must be transported quickly and reliably. A photo of the prototype is shown in Fig. 2.

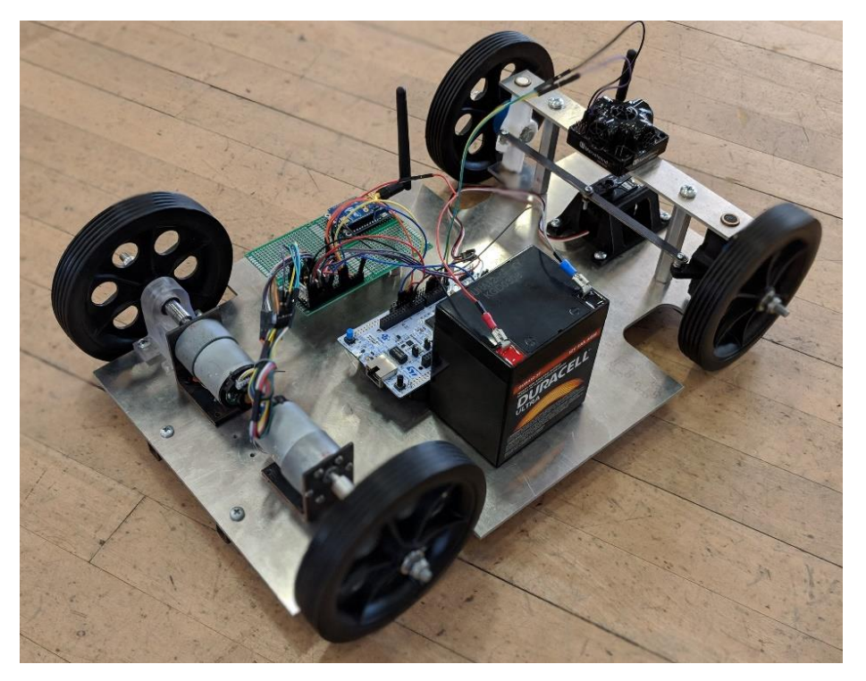

Fig. 2. AMR designed for use in testing

ALGORITHM

For problem (a), we propose the min-min algorithm:

Min-Min Algorithm with Maximum Triangle Initialization

Given a set of n points V^N in a symmetric Euclidean space, the distance between any two points is given by $d(t_i, t_j), t_i, t_j \in V^N$. T^N is defined as a complete tour, $\{t_1, t_2, \ldots, t_N, t_1\}, t_i \in V^N$, and $D(T^N)$ is the tour length. The proposed algorithm can then be broken down into three parts:

1. Selection of the Initial Triangle [1]:

a) Search for $\{t_1, t_2\} \in V^N$ such that $d(t_1, t_2) = \max_{\{t_i, t_j\} \in V^N} \{t_i, t_j\}$.

b) From the remaining points, select $t_3 \in V^{\{N-2\}}$ such that $D(T^3) = \max[d(t_1, t_3) + d(t_2, t_3) + d(t_1, t_2)]$.

2. Min-Min Iterative Addition: For $i = 3, 4, \cdots, N-1$

a) In T^i, for each edge, $\{t_k, t_{k+1}\}, k = 1, 2, \cdots, i, t_{\{i+1\}} = t_1$, select a point, $v_j \in V^{\{N-i\}}$, such that the disturbance introduced is minimized, i.e., $\Delta(t_k, v_j) = \min_{\{v_l \in V^{\{N-i\}}\}} [d(t_k, v_l) + d(t_{\{k+1\}}, v_l) - d(t_k, t_{\{k+1\}})]$.

b) Select $\{t_m, t_{\{m+1\}}\}$ and v_l such that $\Delta(t_m, v_l) = \min_{\{t_k \in T^i\}} [\Delta(t_k, v_l)]$.

c) Add point v_l in tour $T^{\{i+1\}}$ between points $\{t_m, t_{\{m+1\}}\}$.

3. Add pairwise exchange heuristics.

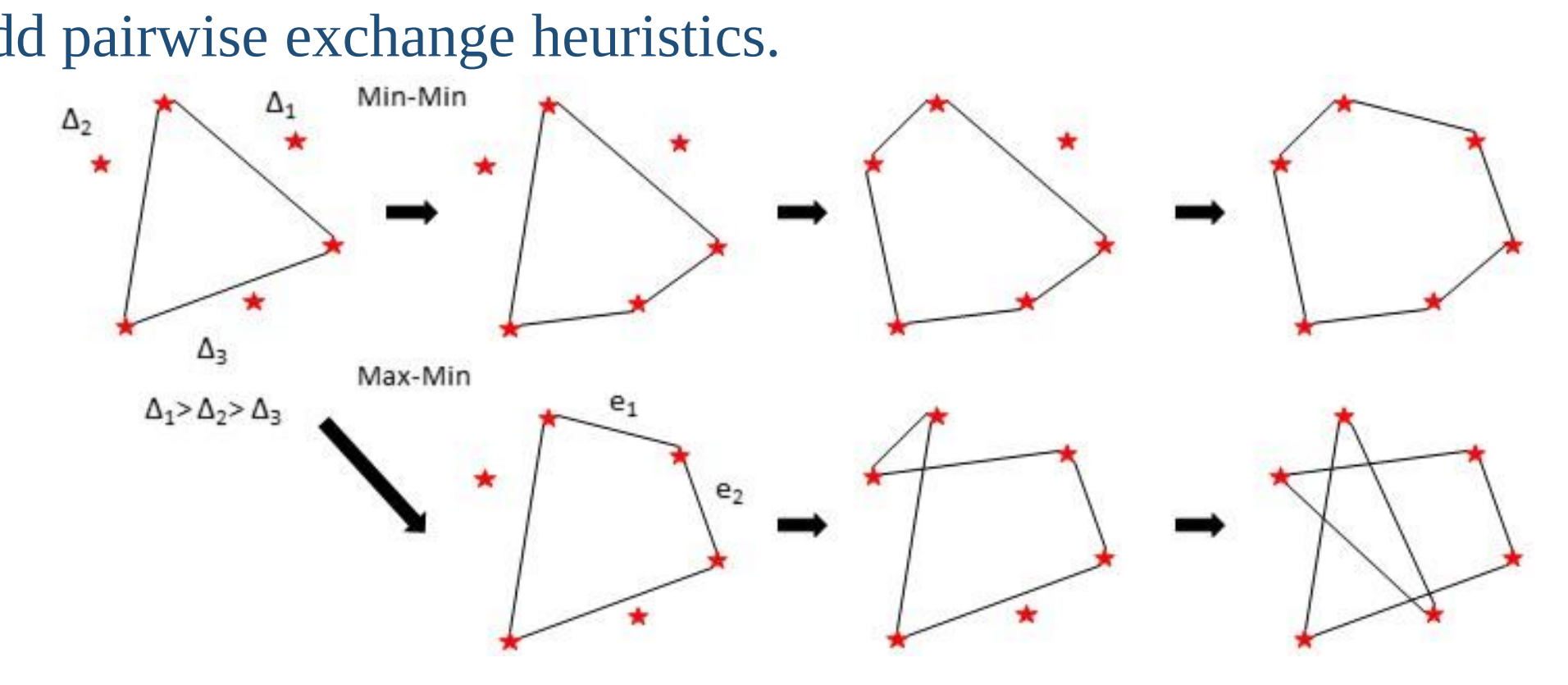

Fig. 3. Evolution of min-min and Yatsenko's [1] algorithms

For problem (b), we propose a novel two-stage algorithm:

(A) Asymmetric Clustering

Priority Separation

To ensure that there is no concentration of tasks of a single priority assigned to a single AMR, we define the priority of each task as a 3rd coordinate. Therefore, for each of the priority values, all tasks of that priority value are separated into k clusters using the clustering algorithm described later, after which the 'centroids' of these clusters are matched across different priority values.

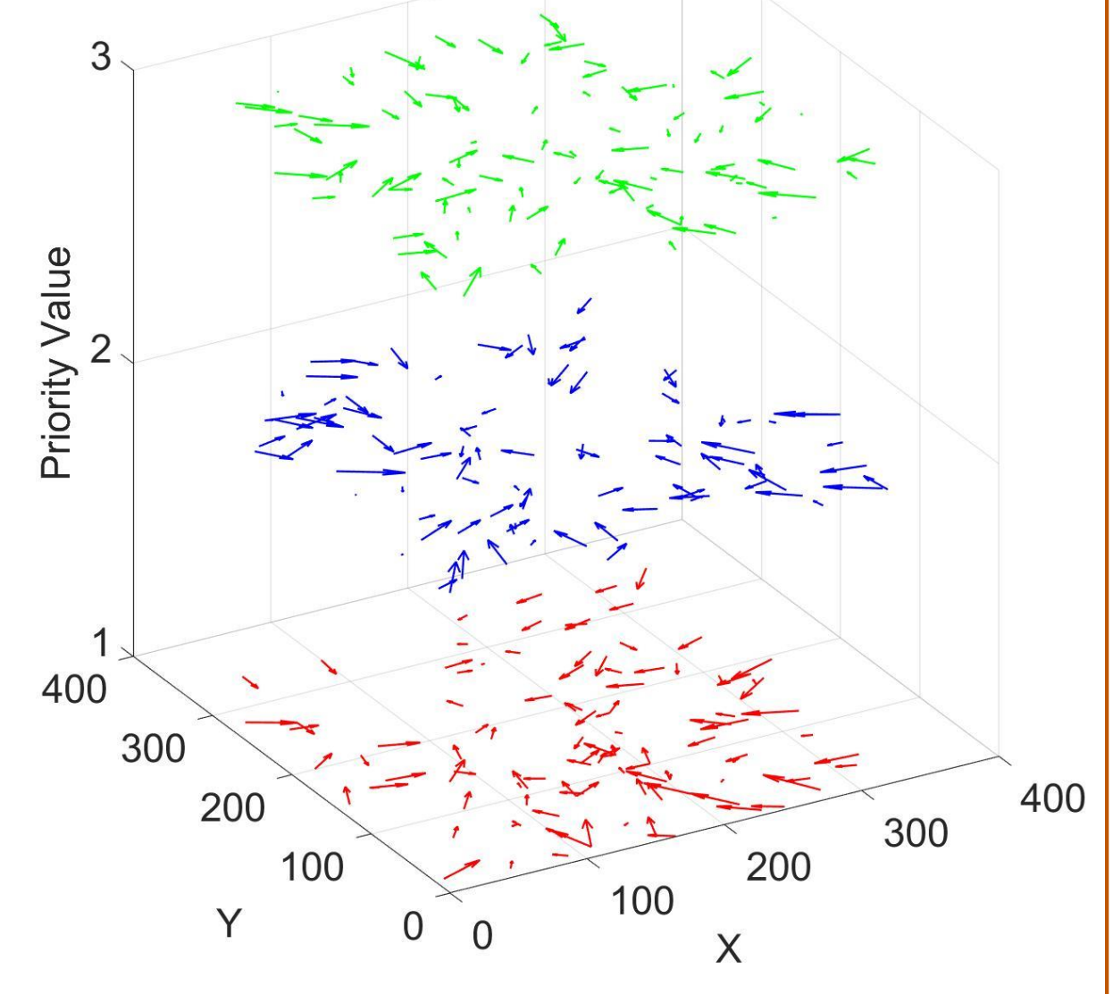

Fig. 4. Priority separation

MKM Algorithm

The objective of clustering is to find k clusters within a priority level such that if a pair of tasks is chosen from the same cluster, the sum of the cost of travel from the end point of the first task to the start point of the second task, and vice versa, should be small relative to tasks in other clusters. Mathematically, the clustering problem can be defined as follows:

$$J(\mu, L) = \sum_{i=1}^{|\Delta|} \left(d_{\delta_i \mu_j(L_i)} \right)^2$$

where μ is the set of cluster 'centroids', representing a 'center' for that particular cluster. L is the cluster label of each task in the set, Δ is the entire task set, and $\mu_j(L_i)$ is the 'centroid' of the cluster to which the task $\delta_i \in \Delta$ is assigned.

Cluster Recombination

Each of the clusters is represented by its pseudo-task centroid. We combine them to form k final clusters, each containing three separate clusters, one from each priority level.

ALGORITHMS (contd.)

(B) Task Ordering for Single AMR

Once the task set has been clustered, each cluster can be handled by a single AMR. Using the task distance matrix, a new model based learning technique is proposed for task ordering with priorities for a single AMR.

Model Structure

The model structure consists of two recurrent neural network sections with LSTM (Long Short-Term Memory) units. The first section of the model is the encoder network, into which the task list for each vehicle is inputted one at a time. It is followed by the decoder network which uses the encoded form of the task list to generate a sequential list of tasks to be carried out by that particular vehicle.

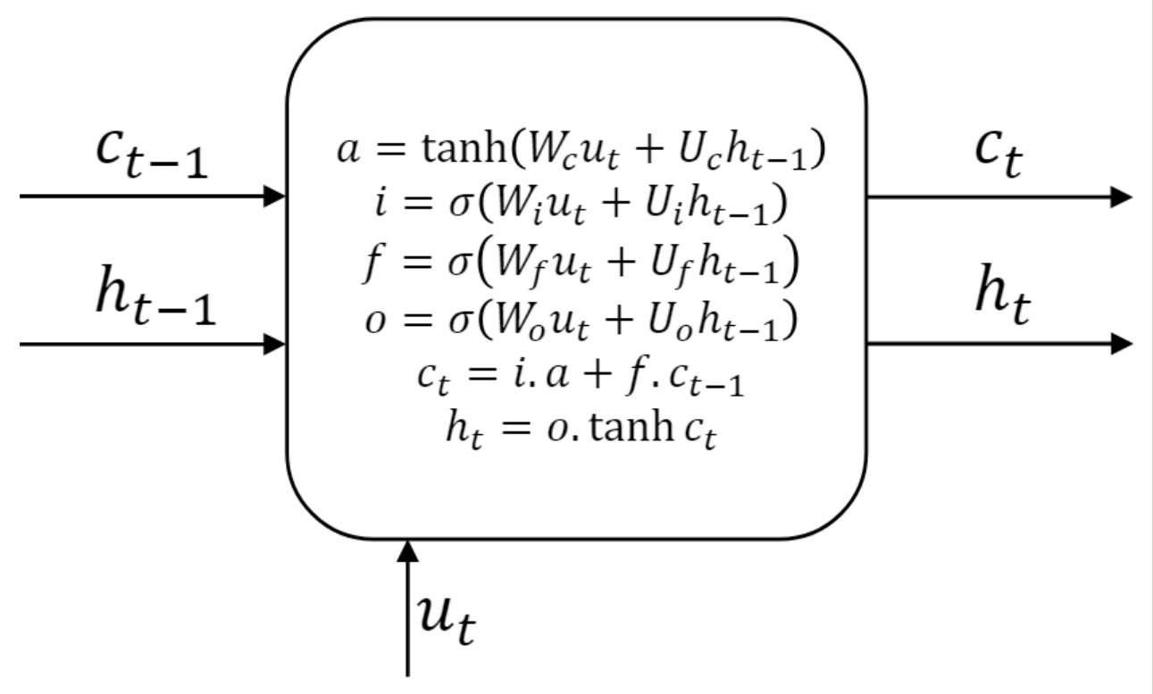
c_{t-1}
h_{t-1}
$a = \tanh(W_c u_t + U_c h_{t-1})$
$i = \sigma(W_i u_t + U_i h_{t-1})$
$f = \sigma(W_f u_t + U_f h_{t-1})$
$o = \sigma(W_o u_t + U_o h_{t-1})$
$c_t = i.a + f.c_{t-1}$
$h_t = o.\tanh c_t$
c_t
h_t
u_t

Fig. 5. LSTM Cell

Reward Function

The reward function for training the parameters of the neural network (Φ) is the negative of the priority adjusted cost between the tasks in the task list, plus the negative of the travel cost between the last task and the depot (without priority adjustment) as the vehicle has to return to the depot.

$$R(\Omega) = -\sum_{i=1}^{n-1} D_{tasks,p,\Omega_i\Omega_{i+1}} - D_{tasks,\Omega_n\Omega_1}$$

The expected value of the reward is $J(\Phi) = \mathrm{E}_{p_\Phi(\Omega)} R(\Omega)$ where $p_\Phi(\Omega)$ is the probability of getting the task sequence Ω given the parameter set Φ, and takes into account the recurrent network. Policy gradient methods can be used to maximize expected rewards, using the REINFORCE algorithm.

RESULTS

For problem (a), results are presented below:

(A) Shortest Tour Ending at Origin

Results on various data sets for the proposed algorithm are presented, and the performance of the algorithm is compared to other commonly used algorithms such as the insertion algorithms by Rosenkrantz *et al.* [2] and the nearest neighbor heuristic [3].

Table 1. Results for Randomly Generated Instances

No. of	Average Time (s)					Average % Error (Standard Deviation)				
Points	FI	NI	CI	NN	Min-Min	FI	NI	CI	NN	Min-Min
200	0.134	0.138	0.177	0.026	0.158	17.79 (1.82)	23.86 (2.76)	19.17 (2.19)	24.54 (4.62)	13.90 (1.90)
500	0.615	0.653	1.183	0.028	1.181	23.10 (1.25)	25.14 (1.13)	20.61 (1.33)	26.03 (2.53)	16.18 (1.13)
700	1.370	1.369	2.392	0.031	2.359	24.30 (1.45)	24.84 (1.66)	20.60 (1.05)	26.20 (3.37)	16.25 (1.13)
850	2.852	2.554	5.382	0.038	5.376	24.80 (1.53)	25.00 (1.05)	20.91 (1.16)	25.86 (2.44)	17.00 (1.15)
1000	4.582	4.648	8.899	0.048	8.869	25.19 (1.02)	25.27 (1.22)	20.49 (1.33)	25.69 (3.06)	17.44 (1.28)

The results for the proposed algorithm and the optimal solution for a 60-point data set is shown in Fig. 6 and Fig. 7.

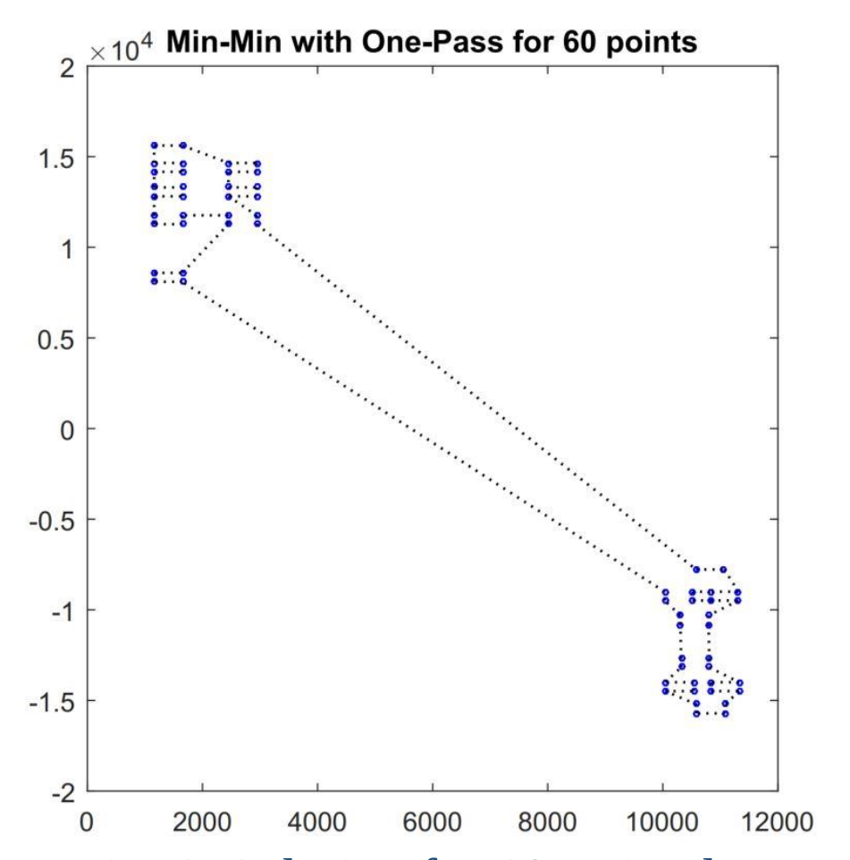

Fig. 6. Solution for 60-point data set using proposed algorithm

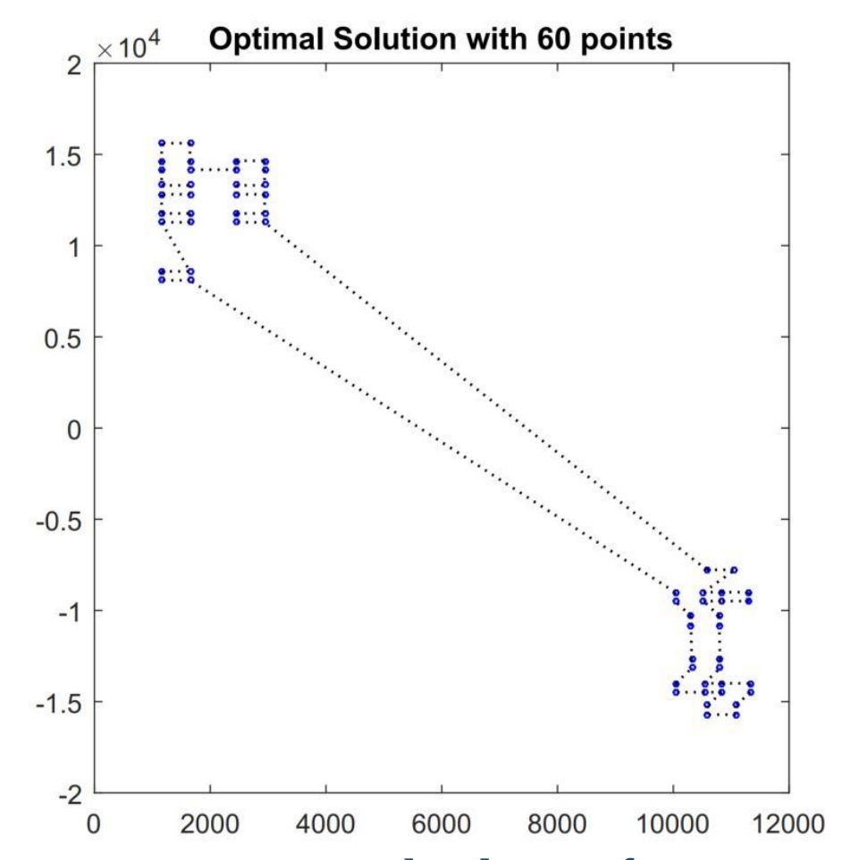

Fig. 7 Optimal solution for 60-point data set

RESULTS (contd.)

(B) Shortest Tour with Fixed/ Free Ends

The proposed algorithm can be extended to tours with arbitrary starting and ending points. One example of using the proposed method with 101 data points is shown in Fig. 8, where the starting location, A, and ending location, B, are marked by green and red markers respectively. One can see that the algorithm enables the AGV to travel to the points with A and B as the two defined end points.

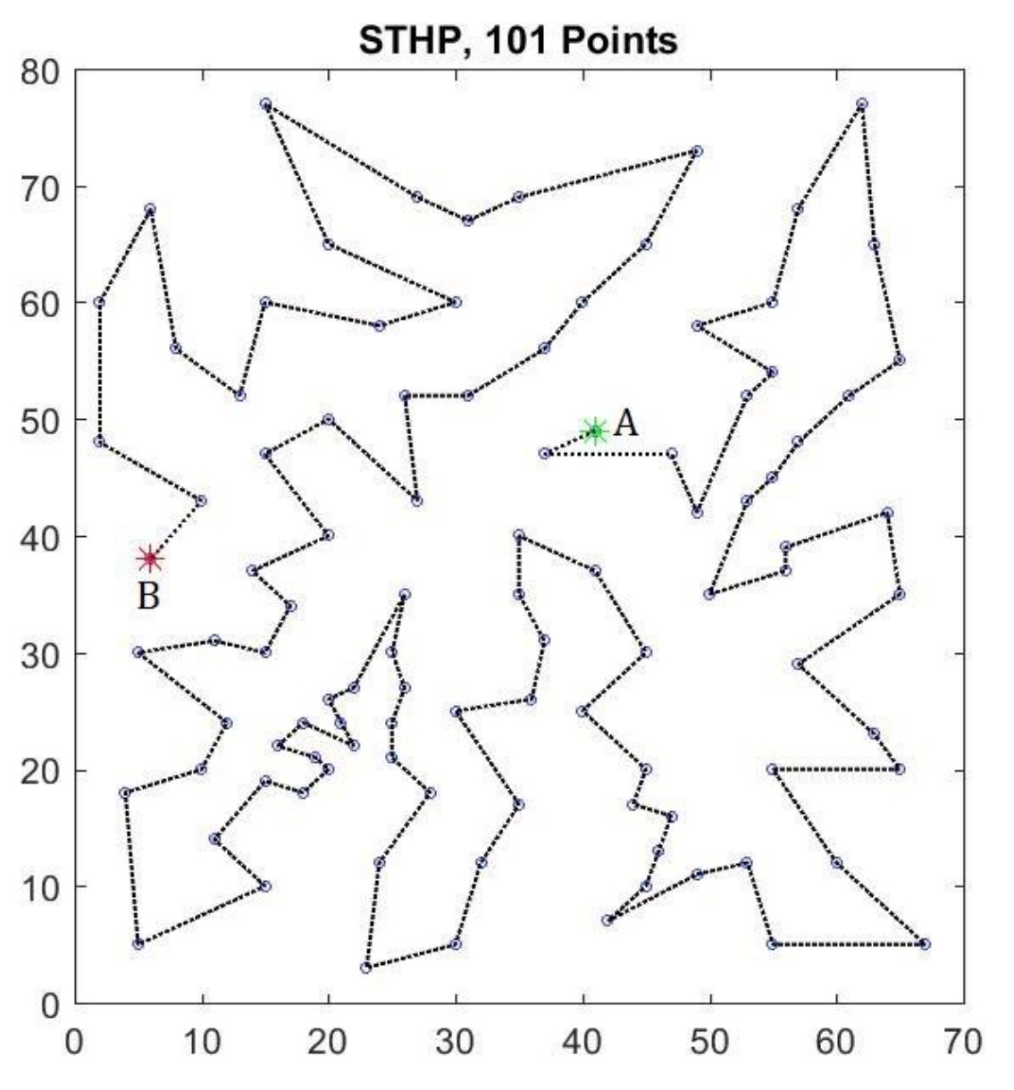

Fig. 8. Solution for 101-point data set with fixed ends using proposed algorithm

For problem (b), results are presented below:

(A) Asymmetric Clustering

The MKM algorithm is compared to the asymmetric K-medoid method (AKMD) and the results are shown in Fig. 9.

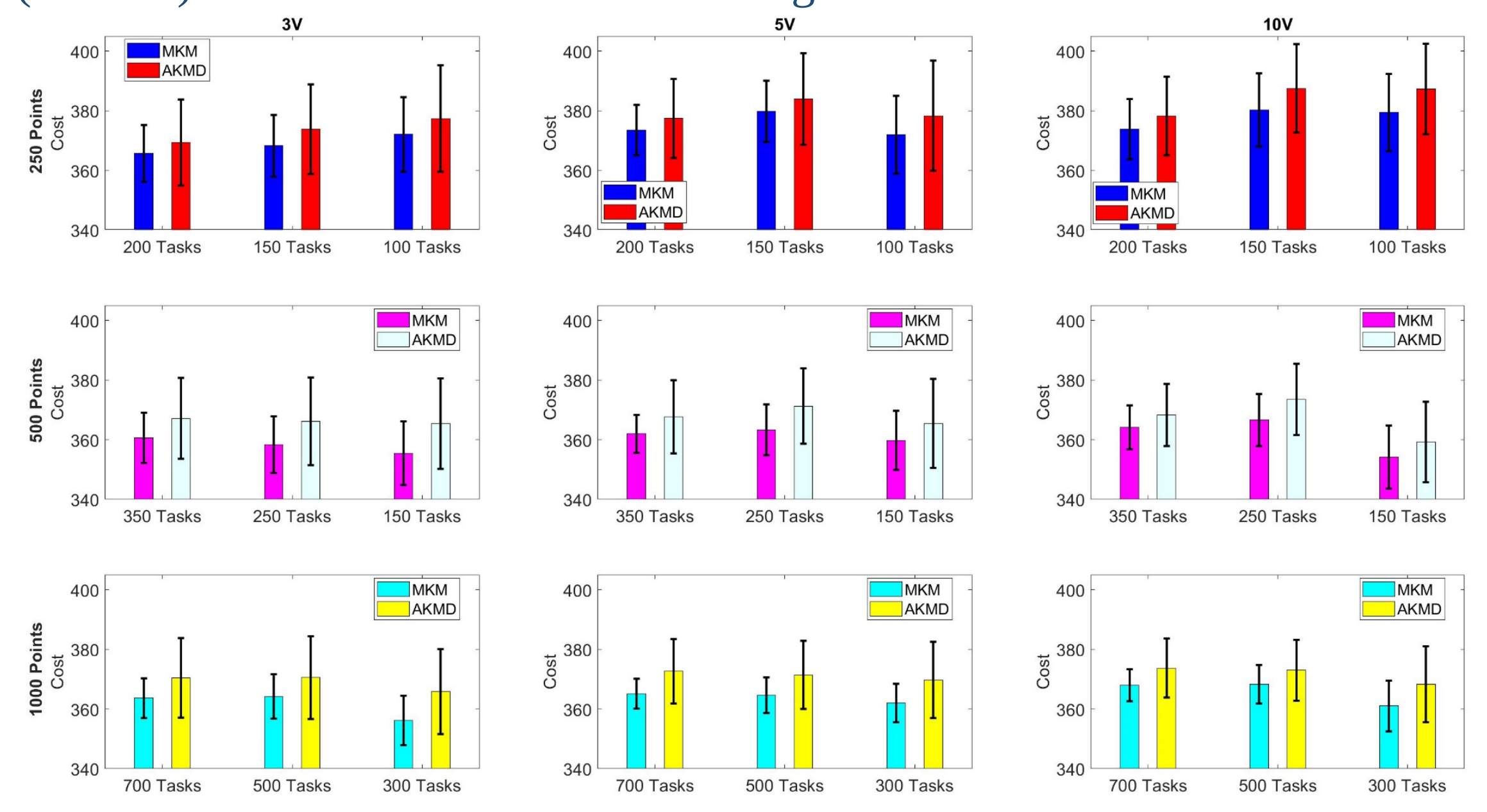

Fig. 9. Comparison of MKM with AKMD methods

(B) Task Ordering

Results from the learning technique are presented in Fig. 10 and Fig. 11.

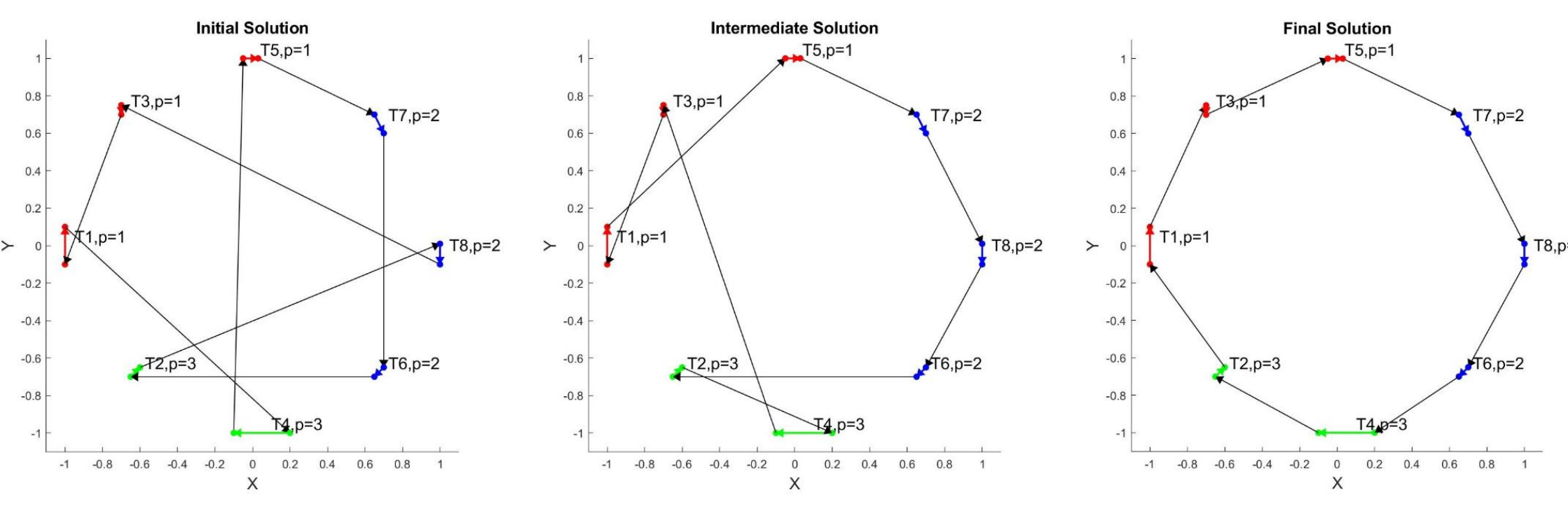

Fig. 10. Test Case for ED Network

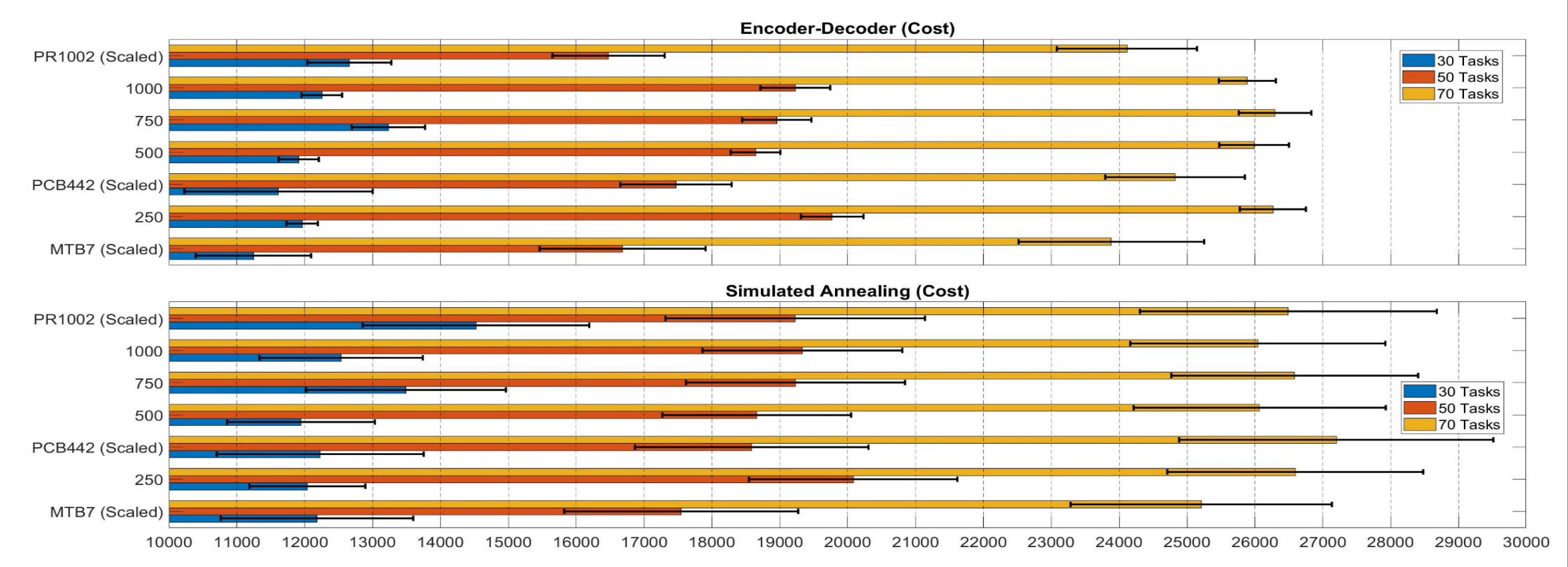

Fig. 11. ED vs Simulated Annealing for different test sets

REFERENCES

[1] Yatsenko, V. Fast exact method for solving the Travelling Salesman Problem. On the WWW. FTP: arxiv.org/ftp Directory: cs/papers/0702 File: 0702133.pdf.

[2] Rosenkrantz, D. J., Stearns, R. E., and Lewis, P. M., 1977. "An analysis of several heuristics for the traveling salesman problem". *SIAM J. Comput.*, **6**(3), pp. 563–581.

[3] Jain, A. K., 1988. Algorithms for Clustering Data. Prentice Hall.

[4] Sheng, W., Xi, N., Song, M., and Chen, Y., 2005. "Robot path planning for dimensional measurement in automotive manufacturing". ASME Journal of Manufacturing Science and Engineering, 127, pp. 420–428.