Dynamic Scheduling for Pickup and Delivery with Time Windows

Shudong Liu, Peng Hui Tan, Ernest Kurniawan, Peng Zhang, Sumei Sun
Institute for Infocomm Research, 1 Fusionopolis Way
#21-01 Connexis (South Tower), Singapore 138632
Email: {liush, phtan, ekurniawan, zhangp, sunsm}@i2r.a-star.edu.sg

***Abstract*—We consider the dynamic scheduling of a pickup and delivery system in which mobile robots are used to transport materials among a set of places. With the help of Internet of Thing, the system has real-time information for demands of transportation and status of robots. How to dynamically scheduling the robots to fulfill the transportation tasks is a key challenge. We propose a method for it in which an innovative and adaptive mixed integer programming model (MIP) is developed. This MIP model can be called in a periodic or event-driven or hybrid way. Numerical results show it can solve real problems fast and significantly reduce transportation cost, comparing with a heuristic policy.**

I. INTRODUCTION

In this paper we consider dynamic scheduling of a pickup and delivery system in which mobile robots are used to pickup and deliver materials among a set of locations. The transportation tasks arise over time and have time windows such that a task must be finished before its due time. With the development of Internet of Thing (IOT), the system can have more and real time information about status of robots and tasks. How to effectively schedule robots utilizing the real time information to finish the tasks with lowest cost is a key problem in the system.

This type of problems rise in many situations. With the great advancement in robotics in recent years, mobile robots are been using in many industries ([1], [2]). In manufacturing environment, mobile robots are used to transport material/components in factories. In health-care area, hospitals use mobile robots to transport materials such as tools (to operation rooms) or food for in-hospital patients. In Singapore, Changyi General Hospital is doing pilot projects for it. In our industrial engagement, a testing center wants to use mobile robots to transport testing samples among labs. Another example is the auto-driven vehicles which are used to transport people in smart cities. These vehicles are smart and work like taxi, being able to pickup a customer and send to his/her destination.

This type of problem belongs to traditional Pickup Delivery Problem with Time Window (PDPTW). However, due to great development of IOT, the problems in the new situation has some important differences from traditional ones. First, in IOT environment, we have more real-time information for tasks and status of robots/vehicles, hence we are at a better position for effective scheduling by using this information. Second, in IOT environment, more emphasis is put on dynamic scheduling and real-time response to customers' demands. Final, in practice demands of transportation arise over time and are stochastic. The traditional PDPTW assumes all information about tasks are given before optimization, and it is a one-time static optimization and assume vehicles will come back to depot after finishing all tasks. While in IOT dynamic environment, we need to dynamically schedule robots and run optimization many times, and in each optimization the robot/vehicle will not come back to depot as they will continue serve other demands.

The PDPTW problem, as a special case of vehicle routing problem (VRP), has been studied by some researchers. Some formulations and solution methods are proposed ([3], [4], [5], [6], [7]). The PDPTW is often formulated as a three-index vehicle flow. However, the number of variables can be very large as the number of requests (transportation task) and vehicle increase. PDPTW is a NP-hard problem and needs huge amount of computation time for real large-size problems. Hence some researchers proposed two-index formulations ([8], [7]). Some researchers proposed customized algorithms (e.g. branch-and-bound, branch-and-cut) to solve these MIP problems, while others use general purpose commercial solvers such as CPLEX to solve them. For comprehensive review, see [9], [10], [11], [12] and [13].

In above literature, the models are solved only one time, assuming vehicle will return back to depot at the end. Dynamic scheduling is critical for real problems and has been studied by many researchers, mainly for manufacturing operations. For a comprehensive review, see [14]. There are three policies to reschedule: periodic, event-driven and hybrid.

We propose an innovative method for dynamic scheduling robots in IOT environment. First we develop an innovative two-index MIP model for scheduling robots for a given batch of tasks of pickup and delivery. The model minimizes the weighted sum of travelling time and finishing time, not forcing the robots to return depot. The model can be solved by commercial solvers. Then we embed this MIP model into a dynamic scheduling framework which can dynamically call the MIP model using a periodic way, event-driven or hybrid way. Our method is flexible and can solve real problems. Numerical results show our method can save traveling time by more than 20%, comparing with a heuristic policy based on human intuitiveness.

The remainder of the paper is organized as follows. In Section II, problem statement is described. In Section III, dynamic scheduling framework and the MIP model is presented. A case

study is reported in section IV. Finally, conclusions are given in Section V.

II. PROBLEM DESCRIPTION

Consider a testing center (or a factory/hospital) in which materials are needed to transported from one place to another and mobile robots are used to do these tasks. Tasks of transporting materials rise over time. Each task is determined by the origin, destination and quantity of transportation. Some tasks are known or predictive, and some are quite uncertain. The tasks have time windows such that the task must be finished before its due time. The robots have its carrying capacity. When the capacity is allowed, a robot may pickup material of task A and then go to another place to pickup material of task B, then go to destinations of these tasks.

When a robot finishes its tasks and no new tasks are assigned to it, the robot will go to some berth point to stay. There is a map for origins and destinations of tasks and berth points. The traveling distance/time from one place to another is deterministic and known. An example of transportation layout is shown in Figure 1. Note that sometimes the places/points may not be at the same floor of a building.

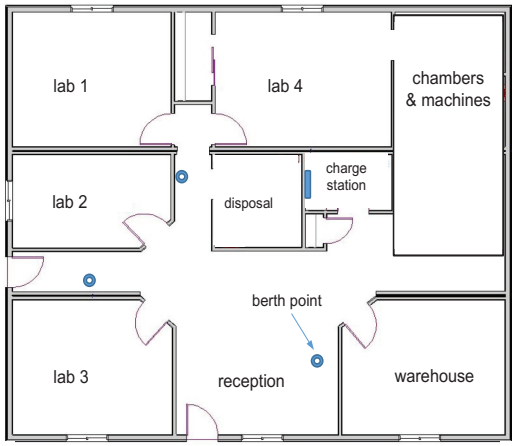

Fig. 1. Layout of pickup and delivery locations

There is a fleet of robots to transport the materials. The capacity and speed of robots are given. Hence we know the travelling time from one place to another. The loading and unloading time of robots are also known. The status of robots and tasks are continuously monitored with the help of IOT system and a central control system. We need to develop dynamic scheduling algorithms/models for the central control system.

III. DYNAMIC SCHEDULING FRAMEWORK AND MATHEMATICAL MODEL

Our solution for the dynamic scheduling consists of two parts: a) a dynamic scheduling framework which controls when to do rescheduling and do preprocessing and post-processing for the scheduling module; b) a scheduling model which is an MIP model.

A. Dynamic Scheduling Framework

We use the general framework in the literature for dynamic scheduling. The rescheduling is taken in a hybrid way: in normal case, scheduling module is called in a periodic way. Whenever an emergent task arrives, the system immediately call scheduling model to generate a new schedule. The length of periods for rescheduling can be set based on practical situations such as rate of task arrival, capacity of robots. Before calling scheduling model, the system consolidates unfinished tasks and check status of robots, and then prepare input files to the scheduling model.

B. Mathematical Model for Scheduling

Assumptions and Notations

When a robot finishes its tasks and no new tasks are assigned to it, the robot will go to a berth point to wait for new tasks. We need a policy for assigning berth points for robots. Here we assume the robots just go to the nearest berth point. In addition, when the system conducts a rescheduling, we assume the system cannot disrupt robots' current unfinished task. For example, a robot is transporting material from A to B. It cannot do new tasks until it finishes this task from A to B.

We develop a MIP model for scheduling, given a set of transportation tasks and the status of robots (earliest starting time and original locations after finishing current tasks). We model it based on the network representation of the problem. Each request is divided into two nodes (origin and destination). Following are the notations for the sets, parameters and variables in the model.

P_M, set of locations in the map including berth locations.

$L_{ber} \subseteq P_M$, set of berth location.

$L_{ini} \subseteq P_M$, initial locations of robots.

$Tr_M\{P_M, P_M\}$, parameter, travelling time from one location to another location in the map.

N_R, parameter, number of requests for transporting material.

$NR = \{1...N_R\}$, set of index for requests.

M_{QR}, parameter, maximum number of samples to transport in one request, which can be set as a very large value, or be equal to the capacity of robots.

$QR = \{1, ..., M_{QR}\}$, set of possible values for number of samples in one request.

$R \subseteq \{NR \times P_M \times P_M \times QR\}$, set of requests which is defined by index of the request, original location, destination location, and number of samples to transport.

$P_P \subseteq \{NR \times P_M\}$, points of pickups which is defined by index of request and the original locations. The values are obtained by extracing from the request set R.

$P_D \subseteq \{NR \times P_M\}$, points of delivery which is defined by index of request and the destination locations.

N_V , number of available robots.

$P_{ini} = \{(N_R + i, l_i), \forall i \in \{1..N_V\}, l_i \in L_{ini}\}$, the set of nodes for initial locations of robots in the transportation network.

$P_{ber} = \{(N_R + N_V + 1, l), \forall l \in L_{ber}\}$, the set of nodes for ending locations (berth) when robots finish all tasks in the transportation network.

T_{ini}, the set of earliest available time for robots.

$NR^+ = NR \cup \{N_R + 1..N_R + N_V\}$, the extended set including indexes for initial locations of robots.

$depot \in P_M$, location of the depot for robots. Here assume all robots have the same depot. If needed, we may extend it to multiple depots in the future.

$P_{PD} = P_P \cup P_D$, set of nodes for pickup and delivery in the transportation network.

$P = P_{PD} \cup P_{ini} \cup P_{ber}$, set of all nodes in the network.

C_{rb}, capacity of a robot, i.e. maximum number of samples that a robot can carry. Assume all robots have the same capacity for simplicity.

$Q\{P\}$, demands at nodes in the network. For node $(i, o) \in P$, $Q[i, o]$ is the amount to transport in request i. For node $(i, o) \in D$, $Q[i, o]$ is (-1) times the amount to transport in request i, and the demand at the initial locations and berth locations is 0.

$Tr\{P, P\}$, parameter, travelling time from one location to another location.

$TE\{P\}$, the earliest time to start service at each node.

$TL\{P\}$, the latest time to start service at each node.

$TS\{P\}$, the service time at each node, e.g. upload and download time.

M, parameter, a big number for modelling.

T_{ft}, finishing time of all deliveries

W_{ft}, parameter, weight for finishing time in the objective function.

W_{tt}, parameter, weight for total traveling time in the objective function.

We have the following decision variables:

$x_{i,o}^{j,d} \in \{0, 1\}$, $(i, o) \in P$, $(j, d) \in P$, 1 if a robot travels directly from node (i, o) to node (j, d), 0 otherwise.

$q_{i,o} \geq 0$, $(i, o) \in P$, the robot load just after visiting node (i, o).

$t_{i,o} \geq 0$, $(i, o) \in P$, the time when a robot starts service at node (i, o).

The MIP Model

As the MIP model needs to run many times for different scenario, it should be flexible. For example, it needs to consider the current locations and status of robots when the system needs to reschedule. The optimization also needs to consider effect of current decisions on later rescheduling and later tasks. We want to minimize both travelling time and finishing time of current batch of tasks. The MIP model is as follows:

$$min \quad W_{tt}\left[\sum_{(i,o)\in P}\sum_{(j,d)\in P} Tr_{i,o}^{j,d} \cdot x_{i,o}^{j,d}\right] + W_{ft} \cdot T_{ft} \quad (1)$$

$$s.t. \quad \sum_{(i,o)\in P} x_{i,o}^{j,d} = 1, \forall (j, d) \in P_{PD} \quad (2)$$

$$\sum_{(j,d)\in P} x_{i,o}^{j,d} = 1, \forall (i, o) \in P_{PD} \quad (3)$$

$$t_{j,d} \geq t_{i,o} + TS_{i,o} + Tr_{i,o}^{j,d} - M(1 - x_{i,o}^{j,d}), \forall (i, o) \in P, (j, d) \in P \quad (4)$$

$$q_{j,d} \geq q_{i,o} + Q_{j,d} - M(1 - x_{i,o}^{j,d}), \forall (i, o) \in P, (j, d) \in P \quad (5)$$

$$t_{i,o} \geq TE_{i,o}, \forall (i, o) \in P \quad (6)$$

$$t_{i,o} \leq TL_{i,o}, \forall (i, o) \in P \quad (7)$$

$$q_{i,o} \geq max(0, Q_{i,o}), \forall (i, o) \in P \quad (8)$$

$$q_{i,o} \leq min(C_{rb}, C_{rb} + Q_{i,o}), \forall (i, o) \in P \quad (9)$$

$$t_{i,d} \geq t_{i,o} + TS_{i,o} + Tr_{i,o}^{i,d}, \forall (i, o) \in P_P \quad (10)$$

$$x_{i,o}^{i,o} = 0, \forall (i, o) \in P_{PD} \quad (11)$$

$$x_{i,o}^{j,d} = 0, \forall (i, o) \in P_{PD}, (j, d) \in P_{ini} \quad (12)$$

$$x_{j,d}^{i,o} = 0, \forall (i, o) \in P_{PD}, (j, d) \in P_{ber} \quad (13)$$

$$x_{i,o}^{j,d} = 0, \forall (i, o) \in P_P, (j, d) \in P_{ber} \quad (14)$$

$$x_{i,o}^{j,d} = 0, \forall (i, o), (j, d) \in P_{PD} \text{ s.t. } TE_{i,o} + TS_{i,o} + Tr_{i,o}^{j,d} > TL_{j,d} \quad (15)$$

$$x_{i1,o1}^{i2,o2} = 0, \forall (i1, o1), (i2, o2) \in P_P \text{ s.t. } i1 \neq i2, Q_{i1,o1} + Q_{i2,o2} > C_{rb} \quad (16)$$

$$x_{i1,o1}^{i2,d2} = 0, \forall (i1, o1), (i2, o2) \in P_P \text{ s.t. } i1 \neq i2, Q_{i1,o1} + Q_{i2,o2} > C_{rb} \text{ and } (i2, d2) \in P_D \quad (17)$$

$$x_{i1,d1}^{i2,d2} = 0, \forall (i1, o1), (i2, o2) \in P_P \text{ s.t. } i1 \neq i2, Q_{i1,o1} + Q_{i2,o2} > C_{rb} \,\&(i1, d1), (i2, d2) \in P_D \quad (18)$$

$$q_{j,d} = 0, \forall (j, d) \in P_{ini} \quad (19)$$

$$t_{j,d} \leq T_{ft}, (j, d) \in P_D \quad (20)$$

$$x_{i,o}^{j,d} \in \{0, 1\}, \forall (i, o), (j, d) \in P \quad (21)$$

$$q_{i,o} \geq 0, (i, o) \in P \quad (22)$$

$$t_{i,o} \geq 0, (i, o) \in P \quad (23)$$

The objective function (1) is to minimize the weighted sum of total travelling time and finishing time. Constraint (2) and (3) ensure that each node is visited once and only once. Constraints (4) and (5) ensure the consistancy of time variables and load variables. Constraints (6)-(9) ensure the time time windows and vehicle capacity. Constaint (10) guarantees the precedence and pairing relations of pickup and delievery. Constraints (11)-(18) are to fix some variables. Constraint (19) ensure robots having no load when they leave from initial locations. Constraint (20) ensures the starting time of all destination nodes is less than the finishing time of all

tasks. Constraint (21) is to impose integrality for variables x. Constraints (22)-(23) are to define continuous variables.

The above model can be solved by general purpose commercial solvers such as CPLEX to get good schedules.

IV. CASE STUDY

In this section we report a case study for a research center to check the feasibility and benefits of our approach. The robots needs to transport materials among seven places including experimental labs and reception (collecting materials). Currently each day has about 30-100 requests. The travelling time between two locations are 2-5 mins. The loading or unloading time is 2 mins. Currently they do not have automatic dynamic scheduling systems and do scheduling by human experience.

In order to investigate the performance of our method, we also developed a heuristic which combines human intuitiveness and a first-come-first-serve (FCFS) dispatching policy. In the heuristic, the requests which have the same or nearby origins and destinations are grouped and served together if the robot has enough capacity to do it.

We found one robot is enough for it. The MIP models are solved at a notebook with Intel Core i7 CPU @ 2.50 GHz and 16G RAM. It can solve 12 requests in 1 min. The saved travelling time (in percentage) using our method is shown in the Figure 2, compared with the FCFS heuristic.

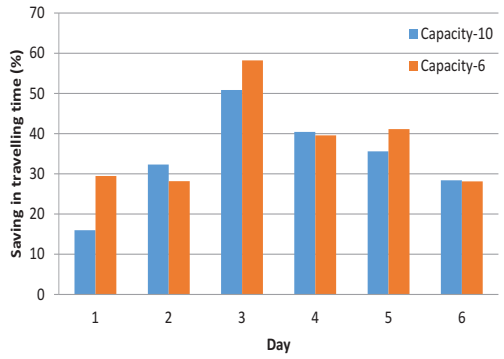

Fig. 2. Saved travelling time by MIP method

From the figure, we can see that the saving is more than 30% in average. When we change capacity of robot from 6 units to 10, there is no obvious trend for saving. From these results, our method can deal with real problems and bring significant saving in cost, and enable automation in the system.

Note that in the above study we did not consider the robot battery charging. In fact, our model can address this issue. We may add a request of charging in the task list and then do the optimization. The loading/unloading time can be set as the charging time.

V. CONCLUSION

We have considered dynamic scheduling for pickup and delivery problems with time windows. We proposed a solution: developed an innovative and flexible MIP model for scheduling and then embed it into a general dynamic scheduling framework. The MIP model minimizes both total travelling time in current batch of tasks and finishing time so that the robots can be at a better position for later tasks. Each scheduling utilizes the information from its previous scheduling and real-time information. The numerical results show our method can deal with real problems in terms of computation time and flexibility. Comparing with the heursitic, our method can significantly reduce travelling cost and time.

There are a few interesting directions for future work. First, we may predict the future demands and actively assign robots to appropriate berth locations to increase response time and save cost. This may be useful in busy environment such as factories and big warehouses. Another direction is to consider robot failure and maintenance in the model. Finally, in current problem setting, each robot can finish a task independently. We may consider the situations where some tasks need multiple robots coordination.

ACKNOWLEDGMENT

The authors would like to express their great gratitude to Abraham Metta of Procter & Gamble for his valuable help in scoping and defining the problem, data collection, validation and verification.

REFERENCES

[1] E. Levner, L. Meyzin, and F. Werner, "Hierarchical scheduling of mobile robots in production-transportation supply chains," *IFAC Proceedings Volumes*, vol. 42, no. 4, pp. 786 – 791, 2009.

[2] S. Giordani, M. Lujak, and F. Martinelli, "A distributed multi-agent production planning and scheduling framework for mobile robots," *Computers & Industrial Engineering*, vol. 64, no. 1, pp. 19 – 30, 2013.

[3] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transportation Science*, vol. 40, no. 4, pp. 455–472, 2006.

[4] S. Ropke, J.-F. Cordeau, and G. Laporte, "Models and branch-and-cut algorithms for pickup and delivery problems with time windows," *Networks*, vol. 49, no. 4, pp. 258–272, 2007.

[5] H. Aytug, M. A. Lawley, K. McKay, S. Mohan, and R. Uzsoy, "Executing production schedules in the face of uncertainties: A review and some future directions," *European Journal of Operational Research*, vol. 161, no. 1, pp. 86 – 110, 2005, iEPM: Focus on Scheduling.

[6] N. Bianchessi and G. Righini, "Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery," *Computers & Operations Research*, vol. 34, no. 2, pp. 578 – 594, 2007, reverse Logistics.

[7] M. G. S. Furtado, P. Munari, and R. Morabito, "Pickup and delivery problem with time windows: A new compact two-index formulation," *Operations Research Letters*, vol. 45, no. 4, pp. 334 – 341, 2017.

[8] Q. Lu and M. Dessouky, "An exact algorithm for the multiple vehicle pickup and delivery problem," *Transportation Science*, vol. 38, no. 4, pp. 503–514, 2004.

[9] M. W. Savelsbergh and M. Sol, "The general pickup and delivery problem," *Transportation science*, vol. 29, no. 1, pp. 17–29, 1995.

[10] S. Parragh, K. Doerner, and R. Hartl, "A survey on pickup and delivery problems: Part ii: Transportation between pickup and delivery locations," vol. 58, pp. 81–117, 06 2008.

[11] G. Laporte, "Fifty years of vehicle routing," *Transportation Science*, vol. 43, no. 4, pp. 408–416, 2009.

[12] K. Braekers, K. Ramaekers, and I. V. Nieuwenhuyse, "The vehicle routing problem: State of the art classification and review," *Computers & Industrial Engineering*, vol. 99, pp. 300 – 313, 2016.

[13] P. Toth, D. Vigo, P. Toth, and D. Vigo, *Vehicle Routing: Problems, Methods, and Applications, Second Edition*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2014.

[14] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems," *J. of Scheduling*, vol. 12, no. 4, pp. 417–431, Aug. 2009.